The Design and Implementation of VGA Controller on FPGA

Radi H.R., Caleb W.W.K., M.N.Shah Zainudin., M.Muzafar Ismail.
Faculty of Electronic and Computer Engineering
Universiti Teknikal Malaysia Melaka
Hang Tuah Jaya
76100 Durian Tunggal, Melaka, Malaysia

***Abstract*— Industrial production machines of today must be highly flexible in order to competitively account for dynamic and unforeseen changes in the product demands. Field-programmable gate arrays (FPGAs) are especially suited to fulfill these requirements; FPGAs are very powerful, relatively inexpensive, and adaptable, since their configuration is specified in an abstract hardware description language. Thus, in order to design and implement VGA Controller on FPGA, Verilog Hardware Description Language (Verilog HDL) is used. Verilog HDL is used to describe and program the gates and counters in FPGA blocks in order to construct an internal logic circuit in FPGA. The main purpose of this project is to design and implement VGA Controller on FPGA. Therefore, the block diagram for VGA Controller is designed and the VGA Controller program is written based on the block diagram using Verilog HDL. Also, functions required for VGA Controller are included in the Verilog code and test bench is created to test the functions written to ensure the FPGA VGA Controller works correctly and accurately without errors. Finally, the completed program is implemented on FPGAs chip of Altera DE2-115 Development and Educational Board.**

***Index Term*— Altera DE2-115 Development and Educational Board, block diagram, Field-programmable gate arrays (FPGAs), Verilog Hardware Description Language (Verilog HDL), VGA Controller**

I. INTRODUCTION

Field-Programmable Gate Arrays (FPGAs) are digital integrated circuits (ICs) that contain configurable blocks of logic along with configurable interconnects between these blocks [1]. Specifically, an FPGA contains programmable logic components called logic elements (LEs) and a hierarchy of reconfigurable interconnects that allow the LEs to be physically connected. LEs can be configured to perform complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory [2].

VGA (video graphics array) is a video display standard. It provides a simple method to connect a system with a monitor for showing information or images. As a standard display interface, VGA has been widely used. There is more and more need in displaying the result of the process in real time as the fast development of embedded system, especially the development of high speed image processing [3]. Apart from that, display will be replacing paper for future. Words of wisdom; seeing is believing and picture telling thousand words, display can give correct information about something. Display is used when people present something. Pictures or texts at display catch more attention than verbal voice when people are doing presentation. When people do that kind of presentation, there must be some device involved in control the display.

Verilog Hardware Description Language (Verilog HDL) is a popular and standard hardware description language which is now extensively used by engineers and scientists on digital hardware designs. Verilog HDL offers many useful features for digital hardware design, that is, Verilog HDL is a general-purpose hardware description language that is easy to learn and easy to use. It is similar in syntax to the C programming language. Verilog HDL allows different levels of abstraction to mix in the same model [4]. Thus, a hardware model can be defined in terms of switches, gates, RTL, or behavioral code. Also, most popular logic synthesis tools support Verilog HDL. This makes it the language of choice for designers [4].

The purpose of this project is to design a VGA Controller using Verilog HDL and implement it on FPGA. First and foremost, RGB data are abstracted from an image file in bitmap format using MATLAB and rearranged using Microsoft Excel. The arranged data are then stored in a MIF file created by using Altera Quartus II compiler software. After that, a VGA Controller program is written in Verilog HDL using Altera Quartus II compilar software, which will compile, run and simulate the written program. Once the simulation is succeeded, the program will be burnt into Altera DE2-115 Board (Fig. 1), which will process the VGA Controller program and display the image on LCD screen.

II. BASIC KNOWLEDGE

A few basic knowledge need to be understand before starting the design process.

A. Field-programmable Gate Arrays

FPGAs are a semiconductor device containing programmable logic components called "logic blocks", and programmable interconnects [9]. Logic blocks can be programmed to perform the function of basic logic gates such as AND, and XOR, or more complex combinational functions such as decoders or simple mathematical functions. FPGAs are also known as reconfigurable devices. These reconfigurable FPGAs are generally favored in prototype building because the device does not need to be thrown away every time a change is made. This allows one piece of hardware to perform several different functions. Of course, those functions cannot be performed at the same time. Besides that, FPGAs are standard parts, they are not designed for any particular function but are programmed by the customer for a particular purpose [5].

FPGAs have compensating advantages, largely due to the fact that they are standard parts. There is no wait from completing the design to obtaining a working chip. The design can be programmed into the FPGA and tested immediately. Apart from that, FPGAs are excellent prototyping vehicles. When the FPGA is used in the final design, the jump from prototype to product is much smaller and easier to negotiate. Also, the same FPGA can be used in several different designs, reducing inventory costs [5].

B. Altera DE2-115 Development and Education Board

The Altera DE2-115 Development and Education board was designed by professors, for professors. It is an ideal vehicle for learning about digital logic, computer organization, and FPGAs. Featuring an Altera Cyclone® IV 4CE115 FPGA, the DE2-115 board is designed for university and college laboratory use. It is suitable for a wide range of exercises in courses on digital logic and computer organization, from simple tasks that illustrate fundamental concepts to advanced designs [6].

C. VGA Controller

The monitor screen for a standard VGA format contains 640 columns by 480 rows of picture elements called pixel. An image is displayed on the screen by turning on and off individually pixels. Turning on one pixel does not represent much, but combining numerous pixels generates an image. The monitor continuously scans through the entire screen, rapidly turning individual pixels on and off. Although pixels are turned on one at a time, we get the impression that all the pixels are on because the monitor scans so quickly. This is why old monitors with slow scan rates flicker.

Referred to Appendix (Fig. 2), the scanning process starts from row 0, column 0 in the top left corner of the screen and moves to the right until it reaches the last column. When the scan reaches the end of a row, it retraces to the beginning of the next row. When it reaches the last pixel in the bottom right corner of the screen, it retraces back to the top-left corner and repeats the scanning process. In order to reduce flicker on the screen, the entire screen must be scanned 60 times per second. This period is called the refresh rate. The human eye can detect flicker at refresh rates less than 30 Hz. To reduce flicker from interference from fluorescent lighting sources, refresh rates higher than 60 Hz are sometimes used in PC monitors. During the horizontal and the vertical retraces, all the pixels are turned off [7].

The VGA monitor is controlled by 5 signals: red, green, blue, horizontal synchronization, and vertical synchronization. The three color signals, collectively referred to as the RGB signal, control the color of a pixel at a given location on the screen. They are analog signals with voltages ranging from 0.7 to 1.0 volt. Different color intensities are obtained by varying the voltage. For simplicity, these three-color signals are treated as digital signals, so we can just turn each one on or off [7].

The horizontal and vertical synchronization signals are used to control the timing of the scan rates. Unlike the three analog RGB signals, these two sync signals are digital signals. In other words, they take on either logic 0 or logic 1 value. The horizontal synchronization signal is used to control the horizontal deflection circuit in the VGA monitor, so that the start and end of a line of pixels is correctly displayed across the visible display area of the screen. Meanwhile, the vertical synchronization signal is used to control the vertical deflection circuit in the VGA monitor, so that the start and end of a frame (of lines) is correctly displayed between the top and bottom edges of the visible display area of the screen. In other words, the horizontal synchronization signal determines the time it takes to scan a row, while the vertical synchronization signal determines the time it takes to scan the entire screen. By manipulating these two sync signals and the three RGB signals, images are formed on the monitor screen [7].

To obtain the 640 × 480 screen resolution, a clock with a 25.175 MHz frequency is used. A higher clock frequency is needed for a higher screen resolution. For the 25.175 MHz clock, the period is as below:

$$\frac{1}{25.175\ MHz} \approx 0.0397 \mu s \text{ per clock cycle} \quad (1)$$

Referred to Appendix (Fig. 3), for section B of the horizontal synchronization signal, 3.77 µs is needed, which is approximately 95 clock cycles (3.77/0.0397). For section C, 1.79 µs is needed, which is approximately 45 clock cycles. Similarly, 640 clock cycles (section D) for the 640 columns of pixels and 20 clock cycles for section E [7].

The total number of clock cycles needed for each row scan is 800 clock cycles (95 + 45 + 640 + 20). Notice that with a 25.175 MHz clock, section D requires exactly 640 cycles, generating the 640 columns per row. If a different clock speed is used, a different screen resolution will be obtained.

Because the vertical sync signal is analogous to the horizontal sync signal, the same calculations can be performed as with the horizontal sync regions to obtain the number of cycles needed for each vertical region. However, instead of using the number of periods of a 25.175 MHz clock, the times for each vertical region are multiples of the horizontal cycles. For example, the time for a horizontal cycle is 31.77 µs, and section P requires 64 µs, which is approximately two horizontal

cycles (2 × 31.77). Section Q requires 1,020 µs, which equals to 32 horizontal cycles (1,020/31.77). The calculation for section R is 480 horizontal cycles (15,250/31.77).

D. Verilog Hardware Description Language

Verilog HDL is a description language that can be used to model a digital system at many levels of abstraction ranging from the algorithmic-level to the switch-level. The complexity of the digital system being modeled could vary from that of a simple gate to a complete electronic digital system, or anything in between. The digital system can be described hierarchically and timing can be explicitly modeled within the same description.

The Verilog HDL language includes capabilities to describe the behavioral nature of a design, the dataflow nature of a design, a design's structural composition, delays and a waveform generation mechanism including aspects of response monitoring and verification, all modeled using one single language. In addition, the language provides a programming language interface through which the internals of a design can be accessed during simulation including the control of a simulation run.

The language not only defines the syntax but also defines very clear simulation semantics for each language construct. Therefore, models written in this language can be verified using a Verilog simulation. The language inherits many of its operator symbols and constructs from the C programming language. Verilog HDL provides an extensive range of modeling capabilities, some of which are quite difficult to comprehend initially. However, a core subset of the language is quite easy to learn and use. This is sufficient to model most application. The complete language, however, has sufficient capabilities to capture the descriptions from the most complex chips to a complete electronic system.

E. Altera Quartus II Compiler Software

Altera Quartus is programmable logic device design software from Altera. Its features include an implementation of VHDL and Verilog for hardware description, visual edition of logic circuits, and vector waveform simulation [8].

Quartus II is a software tool produced by Altera for analysis and synthesis of HDL designs, which enables the developer to compile their designs, perform timing analysis, examine RTL diagrams, simulate a design's reaction to different stimuli, and configure the target device with the programmer [8].

III. Design and Implementation

A. Block Diagram

Referred to Appendix (Fig. 4), the function of "clock generator" block is to reduce the frequency of input clock from 50 MHz to 25 MHz. Meanwhile, "vga_sync" block is used to generate timing and synchronization signals. The "h_count" and "v_count" indicate the relative positions of the scans and essentially specify the location of the current pixel while the "h_sync" signal specifies the required time to scan a row, and the "v_sync" signal specifies the required time to scan the entire screen. "vga_sync" block also generates the "video_on" signal which indicates whether to enable or disable the display. Besides that, "address generator" block is used to generate address for the "img_data" block by using the "h_sync" and "v_sync" signal. "img_data" block will get the index data (q) from the MIF file according to the address generated. Note that the index data are connected to the "img_index" block to use as the address. The "img_index" block will get the RGB data (q) from MIF file according to the address generated (index data). The RGB data consist of 24-bits, whereas "q [23:16] ", "q [15:8]" and "q [7:0]" indicate the "R_data", "G_data" and "B_data" respectively.

B. Design Flow of VGA Synchronization

First and foremost, reset button is sensed. If reset is equal to 1, "h_count" and "v_count" will be reset to 0. If reset is equal to 0, it will check whether the value of "h_count" is equal to 799 or not. If the value of "h_count" is not equal to 799, it will be increased by 1. The step after "h_count" increased by 1 will be discussed later in part (ii) and part (iii) of the flow chart for "vga_sync" module. Meanwhile, if the value of "h_count" is equal to 799, it will be reset to 0. This is due to one complete horizontal scan is start from 0 to 799. Then, it will check whether the value of "v_count" is equal to 524 or not. If the value of "v_count" is not equal to 524, it will be increased by 1. The step after "v_count" increased by 1 will be discussed later in part (iv) and part (v) of the flow chart for "vga_sync" module. If the value of "v_count" is equal to 799, it will be reset to 0. This is due to one complete vertical scan is start from 0 to 524.

While "h_count" is increased by 1, it will check whether the value of "h_count" is less than 96 or not. The value of "h_count" less than 96 indicates horizontal retrace. If the value of "h_count" is less than 96, "cH_display" will be set to 0. The value of "cH_display" will then store into "h_sync". If the value of "h_count" is not less than 96, "cH_display" will be set to 1 and then stored into "h_sync".

Also, while "h_count" is increased by 1, it will check whether the value of "h_count" is less than 784 and greater or equal to 144 or not. The value of "h_count" less than 784 and greater or equal to 144 indicates the display area for horizontal scan. If the value of "h_count" is less than 784 and greater or equal to 144, "h_valid" will be set to 1. If the value of "h_count" is not less than 784 and greater or equal to 144, "h_valid" will be set to 0. The step after "h_valid" is set will be discussed later in part (vi) of the flow chart for "vga_sync" module.

While "v_count" is increased by 1, it will check whether the value of "v_count" is less than 2 or not. The value of "v_count" less than 2 indicates vertical retrace. If the value of "v_count" is less than 2, "cV_display" will be set to 0. The value of "cV_display" will then store into "v_sync". If the value of "v_count" is not less than 2, "cV_display" will be set to 1 and then stored into "v_sync".

Also, while "v_count" is increased by 1, it will check whether the value of "v_count" is less than 514 and greater or equal to 34 or not. The value of "h_count" less than 514 and greater or equal to 34 indicates the display area for vertical scan. If the value of "v_count" is less than 514 and greater or equal to 34, "v_valid" will be set to 1. If the value of "v_count"

is not less than 514 and greater or equal to 34, "v_valid" will be set to 0. The step after "v_valid" is set will be discussed later in part (vi) of the flow chart for "vga_sync" module.

The value of "h_valid" and "v_valid" obtained from part (iii) and part (v) of the flow chart for "vga_sync" module are used to multiple each other and the resultant value is stored in "cDisplay_EN". After that, the value in "cDisplay_EN" is stored into "video_on".

C. Design Flow of VGA Controller

First and foremost, a 25 MHz "vga_clk" is generated from a 50 MHz input clock. Then, "reset" is generated for "vga_sync" module. The timing diagram for horizontal and vertical scan is generated also. After that, address is generated for the "img_data" module by using the "h_sync" and "v_sync" signal from "vga_sync" module. "img_data" module will get the index data (q) from the MIF file according to the address generated. Note that the index data are connected to the "img_index" module to use as the address. Thus, "img_index" module will get the RGB_data_raw (q) from MIF file according to the address generated (index data). Since the RGB_data_raw consist of 24-bits, it is separated into "q [23:16] ", "q [15:8]" and "q [7:0]", which indicate the "R_data", "G_data" and "B_data" respectively. Next, the connections to the output port are made and image is displayed on LCD screen.

D. Implementation

In this project, Altera DE2-115 Development and Education board is used as a prototype of VGA controller which the Verilog HDL coding written was downloaded into Altera Cyclone IV E - EP4CE115F29C7, the FPGA chip. Thus, Altera Quartus II compiler software was used to be the main synthesis tool to synthesize the written Verilog HDL coding and program it onto Cyclone IV E, the FPGA chip.

Some of the information is important to be checked before burning the code into FPGA chip. The Family and Device are the important part to be checked before downloading the code into FPGA; because of it does not work on the FPGA chosen if the Family and Device set in Altera Quartus II compiler software are different with the type and version of FPGA which to be used as prototype of project.

Besides, design goal has to be made sure it is balanced and there is no error and warning after synthesis processes, and all signals and constraints are completely routed and met respectively [10]. If there is any imbalance design, error or warning, the coding sometimes might still can be downloaded into FPGA but it might certainly have unnecessary problem in prototype functionality or using extra gates in FPGA (used space wasted).

The errors happen normally occurred when there is/are not synthesizable function code being used in Verilog HDL coding written. To solve this problem, different method of coding which synthesizable should be used to replace the method of coding which is not synthesizable. Moreover, the warnings may also happen when there is/are "don't care" or "high impedance" ports which may be useless or not connected after downloading into FPGA. Thus, by referring this summary it is easier for us to remove or connect those missing, extra or miss to add connection variables and avoid wasting space or having tiny problem while testing the Verilog HDL codes on FPGA chip. If there is any warning as mention above, the signals and constraints may not state completely routed and met in the design summary.

After finish the whole synthesis process without any error and warning, device was generated and previewed by using "programmer" in Altera Quartus II compiler software. The device was created and it was really to be programmed into Altera Cyclone IV E – EP4CE115F29C7, the FPGA chip.

IV. Conclusion

In conclusion, Field-Programmable Gate Array (FPGA) is great invention to be used in developing a VGA Controller. By using Verilog Hardware Description Language (Verilog HDL) on FPGA, VGA Controller could be constructed easily without constructing the circuit manually; just to write a behavioral model or few behavioral models based on its logic flows, then simulate it with test benches, synthesize it with netlist, and finally program it onto FPGA. It is very effective as this VGA Controller only needs new data to change to other design display. Thus, FPGA-based VGA controller might be a good choice as it is easy to be designed and tiny to be used.

References

[1] C. Maxfield, "*FPGAs: Instant Access*", Elsevier Ltd., USA, 2008.

[2] Altera Corporation, "*What is an FPGA?"* (26 October 2011), [Online]. Available: http://www.altera.com/products/fpga.html

[3] D. Bing, Z. Qi, and W. Rui, "*The design and implementation of VGA image controller based on FPGA*", College of Automation, Harbin Engineering University, China, 2006.

[4] S. Palnitkar, "*Verilog HDL - A guide to Digital Design and Synthesis*", Sun Microsystems, Inc, USA, 1996.

[5] W. Wolf, "*FPGA-Based System Design*", Pearson Education, Inc., USA, 2004.

[6] Altera Corporation, *DE2-115 Development and Education Board* (28 February 2012), [Online]. Available: http://www.altera.com/education/univ/materials/boards/de2-115/unv-de2-115-board.html

[7] E. Hwang, "*Build A VGA Monitor Controller*", Circuit Cellar, USA, 2004.

[8] Altera Corporation, "*Introduction to Quartus II*" (28 February 2012), [Online]. Available: http://users.ece.gatech.edu/~hamblen/UP3/intro_to_quartus2.pdf

[9] M. Khalil-Hani, "Starter's Guide to Digital Systems VHDL & Verilog Design", Prentice Hall, Malaysia, 2007.

[10] M. D. Ciletti, *Modeling, Synthesis, and Rapid Prototyping with the VerilogHDL. verilog HDL - A guide to Digital Design and Synthesis*: SunSoft Press. 8, 580-592; 1999.MIMOS History (14 April 2012).

APPENDIX

Fig. 1. Altera DE2-115 Development and Education Board

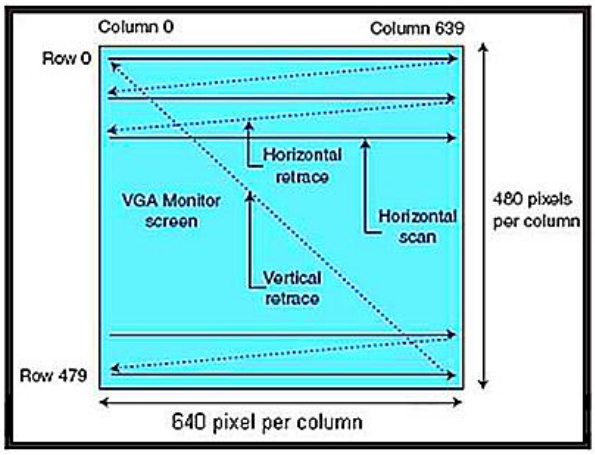

Fig. 2. Scanning pattern of VGA Controller.

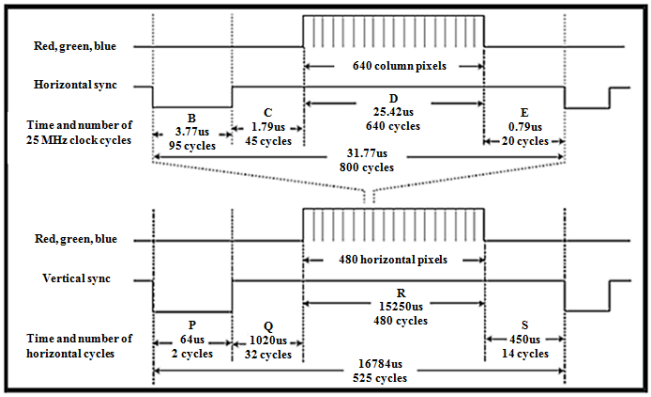

Fig. 3. The horizontal and vertical synchronization signal- timing diagram.

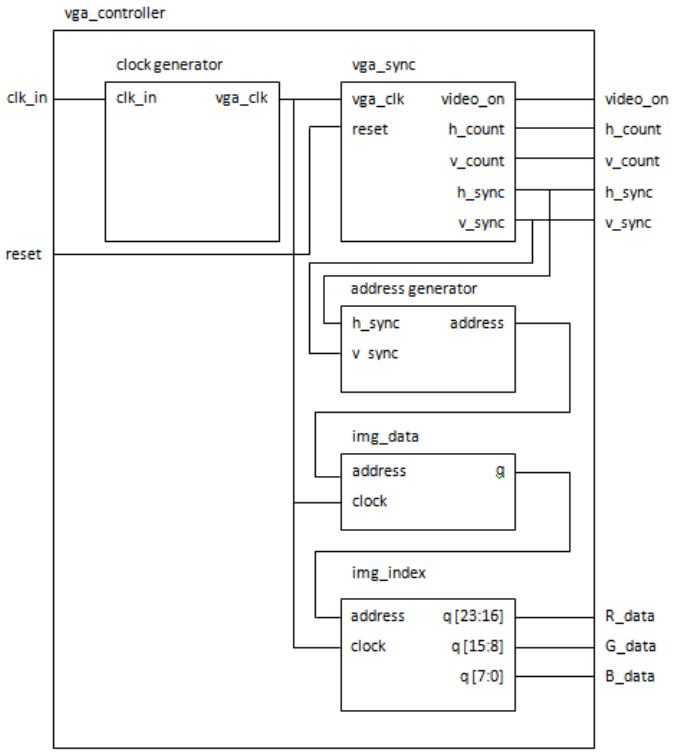

Fig. 4. Block Diagram of VGA Controller.