Design of VGA monitor control using Altera FPGA based system

M.BHARATHI1, A.YOGANANTH2

1PG Scholar, 1,2 Department of ECE, Sembodi Rukumani Varadharajan Engineering College, Sembodai, Tamilnadu, India

k.bharathi05@gmail.com, yogananth1986@gmail.com

ABSTRACT

In this paper, we present the design and implementation of efficient hardware architecture for (Video Graphics Array) VGA monitor controllers based on (Field Programmable Gate Arrays) FPGA technology. The ability to provide multiple display resolutions (up to VGA 640×800) and a customizable internal (First in First Out) FIFO. The main purpose of this project is to design and implement VGA Controller on FPGA with on-chip distributed RAM for reducing memory size, time & cost. In this, the input text has given from mobile module and enabled through GSM Modules. The Altera FPGA has received the text from the mobile module and display in the monitor through VGA. At the same time the VGA Color synchronization has synchronized the color of specified image (flag) in the background of the display. Thus, in order to design and implement VGA Controller on FPGA using Very High Speed Integrated Circuit Hardware Description Language (VHSC HDL) on Altera Quartus II software, implement the VGA Controller program and enable text into Altera FPGA Board to generate images and text on LCD screen, and establish an interconnection between a LCD screen and Altera FPGA Board.

Keywords: VGA- Video Graphics Array, FPGA - Field Programmable Gate Arrays, GSM.

[1] INTRODUCTION

The design is more compact than the others, hence, it only works in a fixed display standard; This paper introduces a VGA monitor controller to work in both graphical mode and text mode. The efficiency of this design provides many choices for different FPGA devices, where system designers can select a proper display mode or configure the internal pixel buffer to be suitable with the application requirements. The design is intently integrated with Processor Local Bus (PLB) interfaces to be used in Altera Cyclone II FPGA-based systems. The purpose of this paper is to design a VGA Controller and enable text in the On-chip memory using GSM Module, VHSC HDL and implement it on FPGA. 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 VHSC HDL using Altera Quartus II compiler software, which will compile, run and simulate the written program. Once the simulation is succeeded, the program will be burnt into Altera Cyclone II Board, which will process the VGA Controller program and display the image on LCD screen. FPGA’s parallel computational advantage is well utilized in this paper. Where timing signals for various VGA screen resolutions can be synced using a single clock and display data’s are loaded simultaneously, thus rendering any text smooth. It is capable of displaying data’s under resolutions ranging from 640x480 to 1280x800 monitors. For implementation purposes our project will use 640x480 resolutions to display the data and for displaying the Font size of the character is 7x15.

Notes: Without using any external memory, By FPGA on chip distributed RAM memory itself used for displaying upto the maximum of 10 characters

2. METHODOLOGY

In this paper the most required things will be VGA, Cyclone II Board on which the FPGA, GSM Module and mobile modules. “Video Graphics Array” is the standard monitor or display interface used in most PCs. and for a display resolution of 640x480 pixels. The VGA supports both All Points Addressable graphics modes, and alphanumeric text modes. There are two kinds of VGA interface signals to display. One is data signal, and the other is control signal. The three data signals are red, green and blue and two control signals are horizontal synchronization and vertical synchronization signals. There are different frequencies of the horizontal and vertical synchronization signal for the changeable output resolution. Thus, if someone wants to implement any application on any higher graphics arrays, the try could be given on the VGA first. So, we did the same thing by implementing the application on the VGA.

In our case we have used the, VHSC HDL to specify the design. It is possible to make use of even both the HDL languages together. We know that the memories associated with the FPGA will have sufficient memory space for small information’s like limited number of characters and some image requiring very small memory to be displayed on screen, but would be insufficient to display an animated characters or images. For this kind of
situation we will require some additional processor that can reduce the length of code and other information while developing some real-time applications.

![VGA Display Using Altera FPGA](image)

**Figure-1 VGA Display Using Altera FPGA**

### 2.1 Main Blocks in the VGA Controller using Altera FPGA:
- Mobile Module
- GSM Module
- Altera Cyclone II Board
- VGA Cable
- LCD or CRT Screen

### 3. BASIC IDEAS AND THE NEEDS OF AN EFFICIENT ARCHITECTURE

In order to display an image on screen, VGA monitor controller has to read every pixel data of the image while driving the color signals and synchronization signals of the VGA interface. All pixels are scanned in raster order at a frequency called pixel frequency. To ensure the visual quality, whole image will be re-drawn at a rate determined by refresh rate. The pixel frequency certainly depends on the display resolution and the refresh rate, better resolution or higher refresh rate will require higher pixel frequency. Some examples can be seen in Table -1.

<table>
<thead>
<tr>
<th>Display Standard</th>
<th>Refresh rate(Hz)</th>
<th>Pixel frequency (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGA 640*480</td>
<td>60</td>
<td>25.175</td>
</tr>
<tr>
<td>SVGA 800*600</td>
<td>60</td>
<td>40.000</td>
</tr>
<tr>
<td>XGA 1280*768</td>
<td>60</td>
<td>65.000</td>
</tr>
<tr>
<td>WXGA 1280*800</td>
<td>60</td>
<td>83.460</td>
</tr>
<tr>
<td>WXGA+ 1440*900</td>
<td>60</td>
<td>106.47</td>
</tr>
</tbody>
</table>

Table -1 Display Standard

The size of an image is usually larger than memory resources available on FPGA devices. The image therefore cannot be stored totally inside the design of VGA monitor controller. It should be held on an off-chip memory (e.g. SDRAM) and transferred into this VGA unit by small data blocks during the display time. For that reason, most of VGA monitor controllers have an internal FIFO memory to temporarily store these data blocks, e.g. [1], [3] or [4]. In the Spartan 3E family using the External memory (e.g. SDRAM). But in Cyclone II family have the following features.

Cyclone® II devices feature embedded memory structures to address the on-chip memory needs of FPGA designs. The embedded memory structure consists of columns of M4K memory blocks that can be configured to provide various memory functions such as RAM, first-in first-out (FIFO) buffers, and ROM. M4K memory blocks provide over 1 Mbit of RAM at up to 250-MHz operation.

The M4K blocks support the following features:
- Over 1 Mbit of RAM available without reducing available logic 4,096 memory bits per block (4,608 bits per block including parity)
- Variable port configurations
- True dual-port (one read and one write, two reads, or two writes) operation
- Byte enables for data input masking during writes
- Initialization file to pre-load content of memory in RAM and ROM modes
- Up to 250-MHz operation

#### 3.1 VGA Timing Control
- Timing of VGA signals are ruled by VESA. Here is a short introduction about how FPGA drive the VGA display with 640×480@60Hz.
In the standard of VGA industry, the output frequency of pixel is 25.175MHz, and the frequencies of horizontal scan and vertical scan are 31.496 KHz and 59.940 Hz. If display receives this standard frequency, then the resolution will be 640×480, and refresh rate is 60Hz.

3.2 Basic pixel timing:
If there is a space of 25.17 μs to handle all of the required pixels, then some basic calculations needs to be carried out to make sure that the Field Programmable Gate Array (FPGA) can display the correct data in the time available.

- For example, if we have a 640 × 480 VGA system, then that means that 640 pixels must be sent to the monitor in 25.17 μs.
- A simple calculation shows that for each pixel we need 25.17 μs/640 = 39.328 ns per pixel.
- If our clock frequency is 100 MHz on the FPGA, then that gives a minimum clock period of 10 ns, which can be achieved using a relatively standard FPGA.

3.3 VGA Data Aligner
From the overall architecture of VGA unit, the FIFO can output 4 or 8 bytes data per clock cycle. However, the VGA Interface module can only push 3 bytes data per clock cycle. The VGA Data Aligner therefore will aim to adjust the data flow between FIFO and VGA Interface.

Basically, VGA Data Aligner is formed as the second FIFO of VGA unit. It has two different architectures for the 32 bits mode and the 64 bits mode. For the 32 bits mode, the FIFO has the size of 12 bytes and 24 bytes for the 64 bits mode. The read and write pointer controllers are controlled by the VGA Controller module. The read address and write address specified in every square of these figures, where the read address is on the right and the write address is on the left. The output data width is always 3 bytes, which is equivalent to three color components RGB.

3.4 VGA Controller
For a selected display standard, VGA Controller will determine horizontal and vertical timing parameters to generate corresponding video timing and to drive synchronization signals. During the display time, PLB Master Burst requests the bus to transfer data from external memory to the FIFO module. The addresses to access the memory and the burst length in every transfer session are calculated by VGA Controller. It also makes a handshake with the bus master interface to control read/write operations of the FIFO and VGA Data Aligner modules. To handle these tasks, VGA Controller includes the following functional modules: Video Timing Generator, Video Address Calculator, FIFO Read/Write Pointer Controller, and VDA Read/Write Pointer Controller. These modules are driven by a finite state machine (FSM).

3.5 Enabling text mode:
Displaying information as text is very useful in many applications. On the basic of the hardware design of VGA unit, we develop a set of methods in its software driver to enable text mode. However, it is obvious that the implementing a character generator by software is much easier and more flexible than by hardware. To generate a character, a standard font is converted to bitmap format. Every character is described as a matrix of pixels. Bit
‘0’ presents a pixel in the background and bit ‘1’ presents a pixel in the foreground of character. Figure 6 shows an example of character “A” with the size of $8 \times 12$ pixels. The image of character “A” is converted to the binary format and represented as an array in C language.

\[
\begin{array}{cccccccc}
XXX & 00000000 & 0x00, \\
XX & 00110000 & 0x38, \\
XX & 01101000 & 0x66, \\
XX & 11000110 & 0xC6, \\
XX & 11000110 & 0xC6, \\
XXXXXXX & 11111111 & 0xFE, \\
XX & 11000110 & 0xC6, \\
XX & 11000110 & 0xC6, \\
XX & 00000000 & 0x00, \\
XX & 00000000 & 0x00
\end{array}
\]

Figure 4. The representation of character “A”.

4. DESIGN OF VGA BACKGROUND COLOUR SYNCHRONIZATION (FLAG)

To the major point in the design of my VGA Timing component gives the number of columns and rows in the frame to other components to help them output the desired RGB signals. However, my VGA timing component is required to give the map addresses to the map RAM and the pixel addresses to the tile RAM to implement the tile idea to the system. This work can be done by separating the column and the row counters and doing the recombination. This component was designed using the VHDL. First, its I/O ports were assigned. This component takes only the clock signal as the input and output is the map address, pixel address, HS, VS and the RGB control.

Its behaviour may be divided into three parts: the vertical control, the horizontal control and the synchronized pulse generating. The horizontal control process the clock input. The counter will count every clock periods. This clock period is the reference time for single pixel to output. The counter is designed to count 800 times at most. After reaching the value 799, it will go back to zero to re-start counting because the extra clock periods that count for the front porch, back porch.

Next step is to decode this horizontal counter in the process of hc signal in order to output the horizontal switch signal. This horizontal switch signal cooperates with the vertical switch signal to decide the value of the RGB control signal which can control when the pixels should be output. The hon signal is switched on for 789-142+1=648 pixels. This means that there are 648 pixels are displayed in one line. This is because that the number of the pixels in one line is different from one screen to another. The method to make the picture fill the screen is to add the number of the pixels drawn in lines one by one and adjust the design based on the result every time. After several modifications, the best number of pixels in one line was decided as 648. So this is why the code is designed to switch on the hon for 648 pixels.

The vertical counter works almost the same as the horizontal counter. But it will count once after the horizontal counter reaching 799, which means the finish of drawing one line. The last part is the synchronized pulse control. The Synchronized pulses are generated with the reference of the counters.

During the display time, the switches will be the logic ‘1’. So the RGB control signal should be the result of the logic AND of the horizontal switch and the vertical switch. The pixels will be output one by one from left to right. This cannot be changed because the VGA screen is fixed. But what we can do to make the component tile-based is to create the map address and the pixel address using the counter.

The next work to do is to design an output logic component so that the work of the VGA timing can be tested. Without the help of the output logic component, the VGA timing component will not control the RGB to output at the display periods itself. It takes the RGB signals as input, do the logic “AND” with the RGB_control signal and output the result. After finishing these two components, I used a test_bench to simulate the work of them. The test_bench simply generates the clock signal for the component as input. The period of this clock is 40ns. Following Figure 5 give the simulation result:

4.1 Design of Character Enabling in Altera Cyclone II Board:

Character enabled in on chip ALTERA FPGA using simulator Quartus II complier. Example binary form of the character to be simulator in 7X15 sizes.
CONCLUSION
In conclusion, Field-Programmable Gate Array (FPGA) is a great invention to be used in developing a VGA Controller. By using VHDL (VHSIC Hardware Description Language) on FPGA, we have presented a design of hardware architecture for VGA monitor controller which has a high potential to be used in Altera FPGA-based systems. The highlighted features such as multiple display resolutions supporting capability, customizable internal FIFO memory, 32/64-bit data bus width, independence to a single clock... make the design suitable for several FPGA devices and able to meet different requirements of targeted applications. This system plays a role as a real-time specific display application. 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
[6]. "IBM VGA Technical Reference Manual". This is the original IBM reference. The document provides a good overview of VGA functionality and is fairly complete, including a detailed description of standard BIOS modes and some programming techniques.