EIE330 FPGA Based System Design

Report of Laboratory 4

FPGA Lab4: VGA Display

Source Code of this report can be found at: EIE330-FPGA-Laboratory

by

PENGRUI K. TANG
Student No: 1220031811
from EIE330 D1,
Faculty of Innovation Engineering,
Universidade de Ciência e Tecnologia de Macau

XIAOYANG S. ZHEN
Student No: 1220034170
from EIE330 D1,
Faculty of Innovation Engineering,
Universidade de Ciência e Tecnologia de Macau

Nov, 2024
(Coloane, Macao SAR)

This report is a part of the result for EIE330 FPGA Based System Design in U.C.T.M

This page is intentionally left blank.

Conteúdo

I Abstract **iii**
I Result location . . . iii
II Contribution . . . iii

II VGA Display **iv**
I Introduction . . . iv
I: i Background . . . iv
I: ii Purpose . . . v
I: iii Design and Key-results . . . v
II Materials and Methods . . . v
II: i Structure Block Diagram . . . v
II: ii Designed Signal Waveforms . . . vi
III Results . . . vii
III: i Testbench . . . vii
III: ii Simulated Result . . . vii
IV Discussions and Conclusion . . . ix
IV: i Discussions . . . ix
IV: ii Conclusion . . . ix
V Appendices . . . ix

Capítulo I

Abstract

I Result location

The Source Code of this report can be found in the git-hub repository: EIE330-FPGA-Laboratory. And the demonstration video of certain codes can be found at the Github release Page or the Youtube video.

II Contribution

All relevant output in this report is shown in the following directory.

- **FPGA Lab4**
  1. 4-0. Report [T][Z]
     (a) Text [T][Z]
  2. 4-3. Simple VGA Display
     (a) Design [T][Z]
     (b) Test [T]

And the contribution of each member is labeled with their last name:

1. Pengrui Tang: [T]
2. Xiaoyang Zhen: [Z]

Capítulo II

VGA Display

I Introduction

I: i Background

- **What is the VGA:**
  VGA, the abbreviation of Video Graphics Array, is a computer display hardware standard introduced by IBM in 1987. It is a standard for computer display hardware. While VGA was widely used, newer interface standards like HDMI (High-Definition Multimedia Interface) have become more common in modern applications, especially for domestic use. VGA supports a maximum native resolution of 640×480 pixels with 4-bit color depth, allowing for 16 colors. Its connection uses a 15-pin D-sub connector, which will be explained in detail below.
- **VGA interface and pins:**
  VGA connector uses 15 pins in 3×5 layout to transmit analog video signals to the monitor. Here is the figure (fig.1) of the interface:

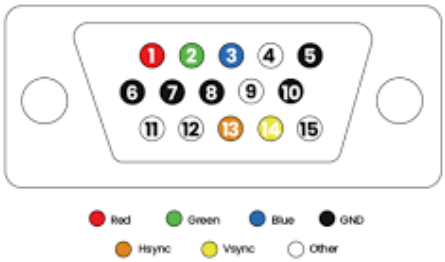

Figura 1: VGA Interface

fig.1 shows the configuration of the pins on the interface, the function of these pins is (number in front represents the pin-no.):

1. **Red**. It transfers the red colour information signal.
2. **Green**. It transfers the green colour information signal.
3. **Blue**. It transfers the blue colour information signal.
4. **Ground**. It is the ground of the **Red Signal**.
5. **Ground**. It is the general ground.

6. **Ground**. It is the ground of the **Green Signal**.
7. **Ground**. It is the ground of the **Blue Signal**.
8. **Ground**. It is the general ground.
9. **Reserved**.
10. **Sync Ground**. It is the ground for synchronization signals.
11. **ID0/Monitor ID**. Monitor identification.
12. **ID1/SDA**. Monitor identification/Display Data Channel data line.
13. **Horizontal Sync (HSync)**. It carries the horizontal synchronization signal.
14. **Vertical Sync (VSync)**. It carries the vertical synchronization signal.
15. **ID3/SCL**. Monitor identification/Display Data Channel clock line.

- **VGA display principle:**
  VGA transmits analog signals to the monitor, which the signals sent consist signal of Red, Green, and Blue information in separate channels. With the help of synchronization signals to coordinate the pixel, the pixel information will be displayed on the screen correctly.
- **VGA timing standards:**
  VGA contains several timing components, which are the pixel clock, sync pulses, horizontal and vertical active video, and refresh rates. They ensure whether the image is displayed by the monitor correctly or not. Furthermore, these components ensure synchronization between the graphics unit and the display by controlling the number of pixels displayed per line, the number of lines per frame, and how often the screen refreshes.

I: ii Purpose

In this experiment, we are required to build an FPGA project that lets "MUST" show in the middle of the screen. We are going to learn how the VGA signal works in Verilog codes.

I: iii Design and Key-results

The designed circuit should be able to display the required text in the middle of the screen. Only one line of text should be displayed. The rest pixels should be off-state.

II Materials and Methods

II: i Structure Block Diagram

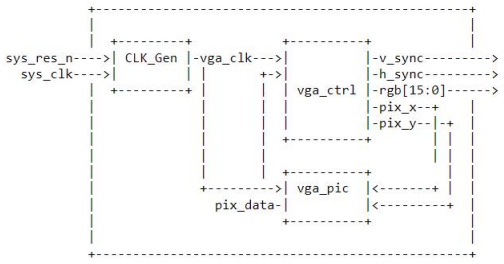

Figura 2: VGA Block Diagram

As the figure 2 shows, the VGA block diagram consists of three blocks:

- **clk_gen**
  This block generates the clock signal for VGA scan from the system clock. The pixel clock generated should have a frequency of 25.175MHz according to the VGA standard for 640x480@60Hz resolution. [2].
- **vga_ctrl**
  This block will generate the control signals for the VGA port according to the VGA standard. When set to work, it'll continuously scan the screen by sending the vertical and horizontal synchronization signals to the display. When it scans to the valid area, it will request the pixel value by outputting the pixel coordinate in the valid area.
- **vga_pic**
  This block generates the pixel's color value based on the pixel position input. If it receives a valid pixel coordinate, it will generate the pixel value to display the MUST item.

  This module has an internal register to store the bitmap of the text MUST. By default, it will output the color black. When the pixel coordinate fells on the area we want to display the bitmap, it will enquire about the bitmap's value. If the value in the bitmap is 1, the pixel value will be set to represent gold; else, it will be set to represent blue.

II: ii Designed Signal Waveforms

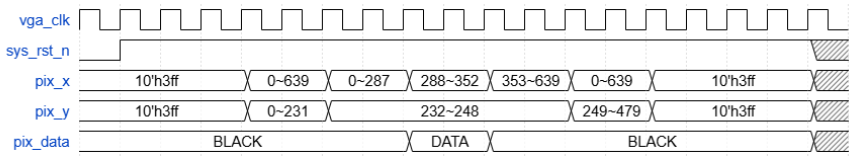

Figura 3: Designed Waveform of VGA Display

As figure 3 shows, there are five different signal channels.

sys_rst_n represents the state of the $RESET$ button. Low means the key is pressed, and high means the key is released.

vga_clk controls the clock of the system, it synchronizes the clock of vertical pixel scan and horizontal pixel scan. It ensures the synchronization of vertical and horizontal signals.

pix_x and pix_y represent the pixel number in the row and the pixel number in the column respectively. Be reminded that there is an empty period for synchronizing vertical signals and horizontal signals before the project starts.

pix_data contains the contents information, the corresponding range of pix_data to the pix_x and pix_y is calculated as equations below:

$$pix\_x_{min} = \frac{640}{2} - \frac{64}{2} = 288 \tag{1}$$

$$pix\_x_{max} = \frac{640}{2} + \frac{64}{2} = 352 \tag{2}$$

$$pix\_y_{min} = \frac{480}{2} - \frac{16}{2} = 232 \tag{3}$$

$$pix\_y_{max} = \frac{480}{2} + \frac{16}{2} = 248 \tag{4}$$

So the valid range of pix_data is pix_x from 288 to 352, pix_y from 232 to 248. Due to the scan logic, each column should be fully scanned when the row number is changing, so pix_y has more clocks than pix_x, it is scanned from pix_x equals 0 to pix_x equals 639.

III Results

III: i Testbench

For the convenience of testing, we wire up vga_control and vga_pic to the testbench, and we also monitor the v_sync and h_sync signals to verify the timing is correct. The simulation result is in the following subsection.

III: ii Simulated Result

1. **Sync**

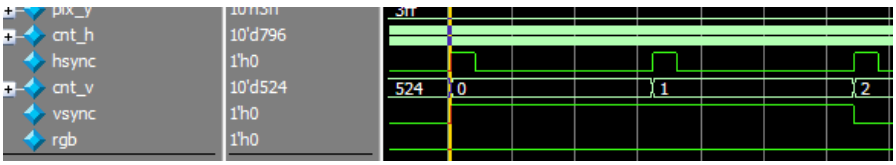

Figura 4: Testbench of Sync

Figure 4 gives an overview of the timing. In this figure, we are mainly focusing on the fourth row (cnt_v). The signal label and its label are on the left.

Field-synchronizing signal is constructed by Sync, Back Porch, Top Border, Addressable Video, Bottom Border, and Front Porch. In this case, addressable video is the real signal that will be shown on the monitor, and other invisible signals will require:

- **Sync** 2 clock cycles
- **Back Porch** 25 clock cycles
- **Top Border** 8 clock cycles
- **Bottom Border** 8 clock cycles
- **Front Porch** 2 clock cycles

While Addressable Video requires 480 pixels (clock cycles), adding up with other elements will give out 525 clock cycles. So when the vertical pixel counter (cnt_v) reaches 524, it turns to 0 and sends a signal to pix_y. Then pix_y will be set to 0 and start scanning from the top to the bottom of the screen again.

2. **Rgb output**

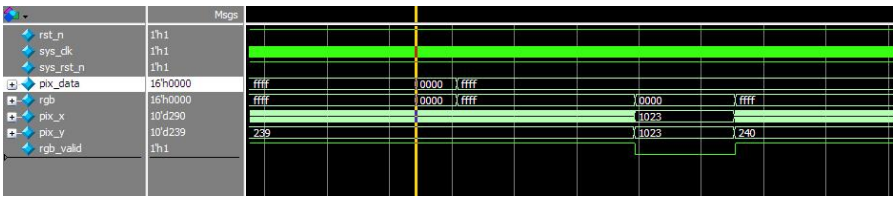

Figura 5: Testbench of RGB

Figure 5 gives an overview of the timing. The signal label and its label are on the left. We implemented this experiment with slight changes, the background is white, and the font is yellow with a black background.

As figure 5 shows, at the beginning of pixel scanning, the pixel is all white (pix_data/rgb 16'hffff). The pix_x keeps scanning until reaches the range of pixels of font, the colour turns to all black (pix_data/rgb 16'h0000), then when the scanned pixel is out of range of font, pixels turn back to all white again until it reaches the edge of the screen. When pix_x

reaches the edge of the screen, rgb_valid transfers from 1 to 0, which means the pixel is turned offs until it starts a new row (pix_y turns from 239 to 240).

IV Discussions and Conclusion

IV: i Discussions

During this experiment, we met multiple issues while we were fulfilling the requirements.

Firstly, we met the problem of converting the text to pixel information.

Then we met the problem of content not showing up on the screen, we found out that is caused by incorrect pixel information for the fonts. Again, we met the issue of text showing up in the wrong direction, it was showing in a column and 90 degrees rotated to the left, instead of showing in a row in the middle of the screen and paralleled.

After we fixed the problem of improper direction, we encountered the problem of multiple fonts displayed in the middle of the monitor. This time we found out that it is because we used miscalculated pixel information logic for the contents.

IV: ii Conclusion

In this lab, we learned about how the VGA signal is configured in Verilog codes. We now understand the relationship between the pixels to the monitor and the relationship between the period of vertical and horizontal scanning. Furthermore, we now know how a font is converted to pixels with colours.

V Appendices

The code in this experiment referenced the following sources (IEEE Style):

Referências

[1] pikipity, Aug, 2024, "FPGA-Laboratory/../3_VGA_Display," distributed on Github, https://github.com/pikipity/FPGA-Laboratory/tree/main/Lab4/2_VGA_Display/RTL

[2] SECONS Ltd, 2008, "VGA Signal Timing" distributed on tinyvga.com, http://www.tinyvga.com/vga-timing