12.4 Video Graphics Array

The video graphics array (VGA) is a display standard used in the CRT and LCD monitors. The Basys3 board has a VGA port as mentioned in Chap. 3. We will use it to develop projects in Verilog and VHDL in this section. Let's start with the working principles of the VGA.

12.4.1 Working Principles of VGA

In the VGA, the display is formed of pixels (picture elements). These are grouped into horizontal lines. Horizontal lines placed on the screen form a frame. Therefore, a pixel location has both horizontal and vertical coordinates. One standard VGA display size is 640×480 pixels. This should be read as follows. The display is formed of 480 horizontal lines each holding 640 pixels.

The time needed to display a single pixel is determined by a pixel clock. Hence, pixels in a horizontal line are displayed by the successive clock signals. When end of the line is reached, the display should continue with a new line. This is set by the horizontal synchronization signal. When all lines in a frame are displayed, a new frame should be formed. This is set by the vertical synchronization signal which also defines the refresh rate of display. The horizontal and vertical synchronization signals depend on pixel clock by definition. Moreover, the monitor needs some time before applying horizontal and vertical synchronization signals. This is called front porch. Similarly, we should wait for a certain amount of time after displaying pixels in a horizontal line and frame. This is called back porch. More information on the VGA timing can be found in [52].

Every pixel has red, green, and blue (RGB) values in the VGA. As mentioned in Chap. 3, the VGA connector on the Basys3 board allows these RGB values to be represented by at most 12 bits. In this scenario, the RGB values get four bits each. Hence, a pixel can have one of $2^4 \times 2^4 \times 2^4 = 4096$ different colors. One can also use eight bits to represent the RGB values. Then, the RGB values get three, three, and two bits, respectively. Hence, a pixel can have one of $2^3 \times 2^3 \times 2^2 = 256$ different colors.

12.4.2 VGA in Verilog

We can display an image using the VGA connection of the Basys3 board. To do so, we first provide the Verilog description of the VGA module in Listing 12.25. This module works in connection with the distributed ROM and clock divider modules. We will introduce such a complete application in Sec. 12.4.4.

The inputs to the VGA module are `clk25`, `pixel_data`, `sx`, and `sy`. `clk25` represents the clock signal fed to the VGA module. For our case, it will be 25 MHz. `pixel_data` represents the vector holding RGB pixel values to be displayed. The VGA module is set to work with eight-bit data. Hence, the RGB values get three, three, and two bits, respectively, as mentioned before. `sx` and `sy` represent the image size to be displayed.

Although the default display size in VGA module is 640 × 480 pixels, it is not possible to keep such an image in the Artix-7 FPGA block or distributed ROM. Therefore, we can set `sx` and `sy` to 80 and 87 pixels at most. The outputs of the VGA module are `red`, `green`, `blue`, `Hsync`, `Vsync`, and `pixel_address`. The outputs `red`, `green`, and `blue` represent the pixel color values each being three, three, and two bits, respectively. `Hsync` and `Vsync` represent horizontal and vertical synchronization signals. Finally, `pixel_address` represents the address of the pixel to be fed to the accompanying ROM module.

Listing 12.25 Verilog Description of the VGA Module

```
module VGA_module (clk25,pixel_data,sx,sy,red,green,blue,Hsync,Vsync,
    pixel_addr);

input clk25;
input [7:0] pixel_data;
input [7:0] sx, sy;
output reg [2:0] red, green, blue;
output reg Hsync, Vsync;
output reg [12:0] pixel_addr;

localparam HDISP=640;
localparam HFP=16;
localparam HPW=96;
localparam HLIM=800;

localparam VDISP=480;
localparam VFP=10;
localparam VPW=2;
localparam VLIM=525;

reg [10:0] hcount=0;
reg [10:0] vcount=0;
reg enable=0;

always@(posedge clk25)
begin

if (hcount < HLIM-1)
    hcount <= hcount+1;
    else
    begin
    hcount<=0;
    if (vcount < VLIM-1)
    vcount <= vcount+1;
    else
    vcount <= 0;
    end

if (vcount > sy)
    begin
    pixel_addr<=-1;
    enable <= 0;
    end
    else
    begin
    if (hcount < sx)
    begin
    enable <= 1;
    pixel_addr<=pixel_addr+1;
    end
    else
    enable <= 0;
    end
```

```
if (enable==1)
    begin
    red <= pixel_data[2:0];
    green <= pixel_data[5:3];
    blue  <= pixel_data[7:6];
    end
    else
    begin
    red   <= 3'b000;
    green <= 2'b00;
    blue  <= 2'b00;
    end

if (hcount > (HDISP+HFP) && hcount <=(HDISP+HFP+HPW))
    Hsync <= 0;
    else
    Hsync <= 1;

if (vcount >= (VDISP+VFP) && vcount < (VDISP+VFP+VPW))
    Vsync <= 0;
    else
    Vsync <= 1;

end
endmodule
```

The working principles of the VGA module in Listing 12.25 are as follows. The module is set to work with 640 × 480 pixels. These values are represented as local parameters in the module. Similarly, front and back porch values for horizontal and vertical synchronization signals are set as local parameters. Based on these values, the maximum horizontal and vertical display limits are set as 800 and 525 pixels as local parameters within the module. We assume that the input clock to the module (`clk25`) has 25-MHz frequency. Based on these values, the refresh time of a frame is 800 × 600/25 × 10^{-6} s. Hence, the refresh rate of the display becomes 52 Hz, which is a suitable value. The VGA module calculates the pixel address to be displayed, whether to generate horizontal and vertical synchronization signals, and RGB values to be used in display. All these operations depend on accurate and synchronous timing calculations. Besides, no detailed operation is performed.

12.4.3 VGA in VHDL

The VHDL version of the VGA module introduced in the previous section is presented in Listing 12.26. Within this module, we set all input, output, and parameter names the same as presented in Listing 12.25. Besides, the working principles of both modules are also the same. Hence, the reader can follow the explanation in the previous section for the VHDL version of the VGA module as well.

12.4.4 VGA Application

We can use the VGA module (in Verilog or VHDL) in a simple application to show its working principles. This application displays an RGB image on the display connected to VGA port of the Basys3 board. We first provide the Verilog version of the top module for the application in Listing 12.27.

Listing 12.26 VHDL Description of the VGA Module

```
    enable <= '0';
    else

if hcount < unsigned(sx) then
    enable <= '1';
    pixcount<=pixcount+1;
    else enable <= '0';
    end if;
end if;

if enable = '1' then
    red <= pixel_data(2 downto 0);
    green <= pixel_data(5 downto 3);
    blue <= pixel_data(7 downto 6);
    else
    red <= (others => '0');
    green <= (others => '0');
    blue  <= (others => '0');
end if;

if hcount> (HDISP+HFP) and  hcount <=(HDISP+HFP+HPW) then
    Hsync <= '0';
    else
    Hsync <= '1';
end if;

if vcount >= (VDISP+VFP) and vcount < (VDISP+VFP+VPW) then
    Vsync <= '0';
    else
    Vsync <= '1';
end if;

end if;
end process;
end behavioral;
```

The top module in Listing 12.27 has input and output values directly set for the Basys3 board. Hence, XDC file namings are used there. Besides, the top module has three submodules: `clock`, `memory`, and `VGA`. The `VGA` submodule is the one in Listing 12.25. The `clock` submodule is for dividing 100-MHz clock of the Basys3 board to 25 MHz to be used in the `VGA` submodule. To do so, we used "Clocking Wizard IP" which can be found in IP Catalog → FPGA Features and Design → Clocking. With its simple interface, this IP block allows generating a frequency divider. The image to be displayed is kept in the distributed ROM as explained in Sec. 9.5. Here, the image is saved in ROM as an initialization file in `coe` format. Here, we suggest a two-step operation. We provide the MATLAB file to generate a text file from a given TIFF image on this book's companion website, www.mhprofessional.com/1259837904.

First, the reader can convert the image file to suitable text format via this file. Second, this file should be converted to `coe` format as explained in Sec. 9.5. This way, the image of interest can be included to the ROM module. As the project is built and the VGA port is connected to the display, the image should be seen on it.

We also provide the VHDL version of the top module for the VGA application in Listing 12.28. This module has the same working principles as presented in Listing 12.27. Therefore, the explanations there directly apply to it as well.

Listing 12.27 Top Module of the VGA Application in Verilog

```
module VGA_top_module(clk,vgaRed,vgaGreen,vgaBlue,Hsync,Vsync);

input clk;
output [2:0] vgaRed;
output [2:0] vgaGreen;
output [1:0] vgaBlue;
output Hsync;
output Vsync;

//image size
parameter sx=80;
parameter sy=80;

wire clk_25;
wire [12:0] pixel_addr;
wire [7:0] pixel_data;

clk_wiz_0 clock (clk25, clk);

dist_mem_gen_0 memory(.a(pixel_addr),.spo(pixel_data));

VGA_module VGA(
.clk25(clk25),
.pixel_data(pixel_data),
.sx(sx),
.sy(sy),
.red(vgaRed),
.green(vgaGreen),
.blue(vgaBlue),
.Hsync(Hsync),
.Vsync(Vsync),
.pixel_addr(pixel_addr));

endmodule
```

Listing 12.28 Top Module of the VGA Application in VHDL