ECE 272 Lab #6

Video Graphics Array (VGA)

Marshall Saltz

12/03/2022

1. Introduction

Lab 6 focuses on displaying colors on a monitor using Video Graphics Array (VGA). VGA uses the outputs hsync and vsync to draw lines controlled by red, green, and blue. The frame is refreshed when vsync goes to 0, and a new display is drawn. The hardware used for this lab is a monitor, a USB to USB-B cable, a VGA cable, my computer, and a DE10-Lite FPGA. The software used is Quartus Prime Lite 18. The goal of this lab is to use 6 switches to control different levels of red, green, and blue on a monitor.

2. Design

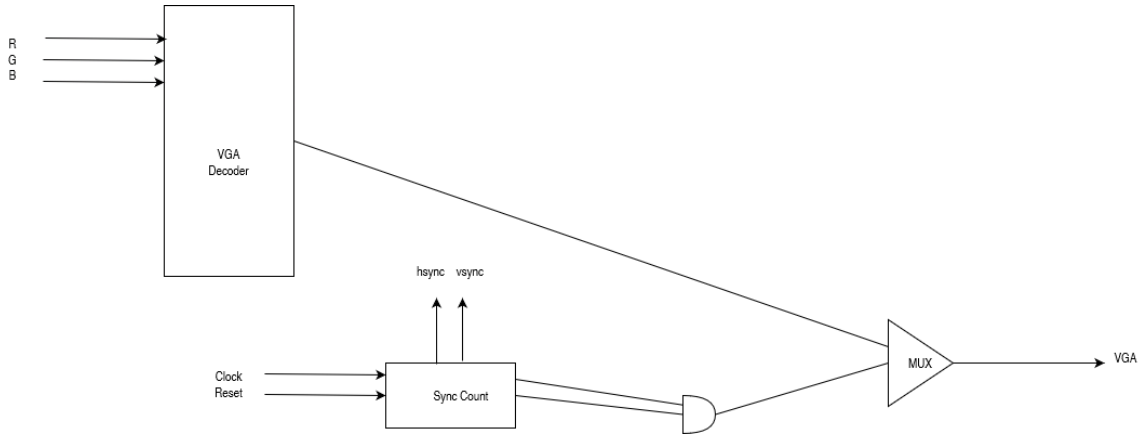

Figure 1: Top Level Block Diagram

The top level block diagram details the logic at the top level. VGA Decoder reads the switches and turns them into VGA signals, which it sends to the multiplexer. Sync count accepts the clock and reset signals and syncs the entire design, outputting hsync, vsync, and to the multiplexer. The multiplexer ultimately chooses which colors to display.

Interface Name	Interface Purpose	Input or Output	Number of Pins	Active High or Low	Board Label	FPGA Pin
clk	Gate Clock	Input	1	High	50 MHz Clock	P11
reset	Gate input	Input	1	Low	Push Button	B8
R[0]	Gate input	Input	1	High	Slide Switch	C10
R[1]	Gate input	Input	1	High	Slide Switch	C11
G[0]	Gate input	Input	1	High	Slide Switch	D12
G[1]	Gate input	Input	1	High	Slide Switch	C12
B[0]	Gate input	Input	1	High	Slide Switch	A12
B[1]	Gate input	Input	1	High	Slide Switch	B12
hsync	Gate Output	Output	1	Low	VGA Horizontal Sync	N3
vsync	Gate Output	Output	1	Low	VGA Vertical Sync	N1
vga[0]	Gate Output	Output	1	High	VGA Red[0]	AA1
vga[1]	Gate Output	Output	1	High	VGA Red[1]	V1
vga[2]	Gate Output	Output	1	High	VGA Red[2]	Y2
vga[3]	Gate Output	Output	1	High	VGA Red[3]	Y1
vga[4]	Gate Output	Output	1	High	VGA Green[0]	W1
vga[5]	Gate Output	Output	1	High	VGA Green[1]	T2
vga[6]	Gate Output	Output	1	High	VGA Green[2]	R2
vga[7]	Gate Output	Output	1	High	VGA Green[3]	R1

vga[8]	Gate Output	Output	1	High	VGA Blue[0]	P1
vga[9]	Gate Output	Output	1	High	VGA Blue[1]	T1
vga[10]	Gate Output	Output	1	High	VGA Blue[2]	P4
vga[11]	Gate Output	Output	1	High	VGA Blue[3]	N2

Figure 2: Interface Definition

The interface definition defines pins for each input and output.

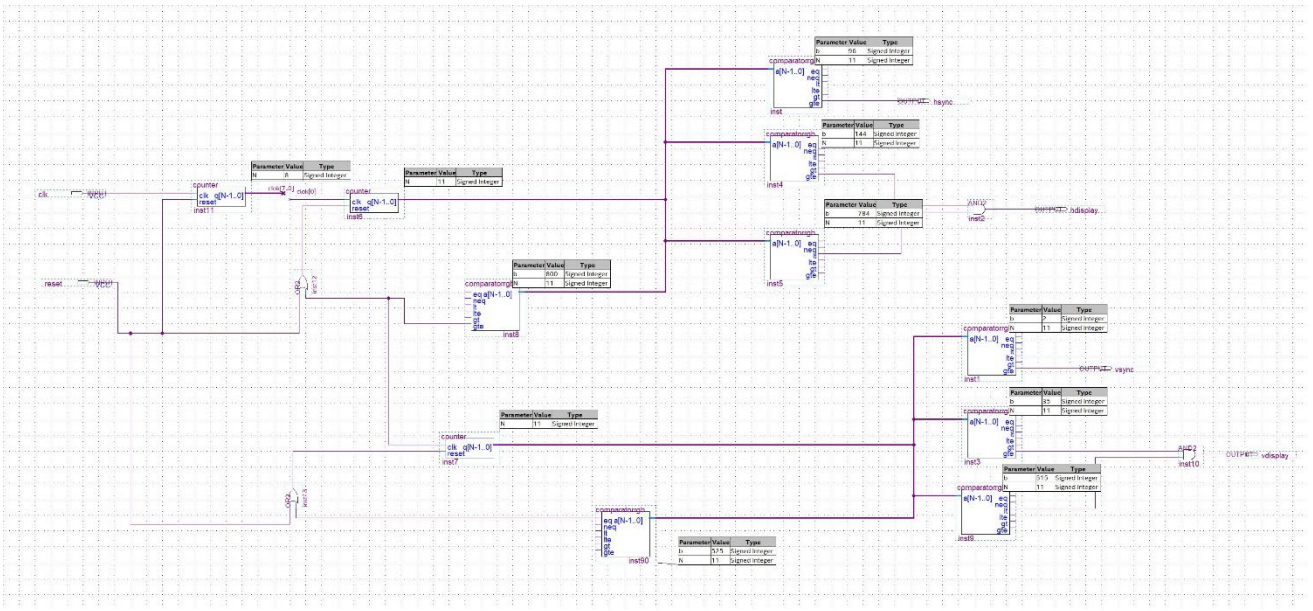

Figure 3: Sync Count Schematic

Sync Count takes in clock and reset and uses a series of comparators, counters, or gates, and an and gate to determine the vertical sync, horizontal sync, and vertical and horizontal displays.

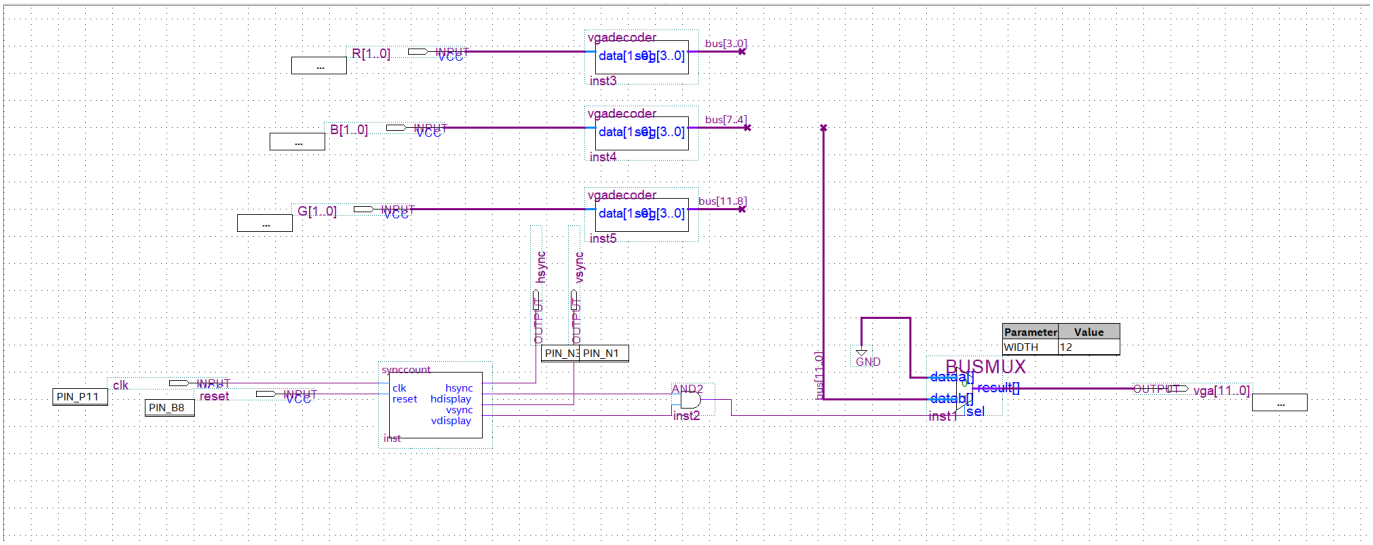

Figure 4: Top Level Schematic
This is the block diagram put into action, and follows the same logic.

3. Results

The monitor was able to display varying levels of red, green, and blue depending on which switches on the FPGA were pushed. I do not have photos of this, as I forgot to take them when I had access to a monitor and VGA cable. The code and schematics worked as intended.

4. Experiment Notes

Another long lab; who would have thought that displaying visuals from bits and bytes could be so difficult. Ultimately what gave me the most trouble was figuring out in which order to place the comparators in sync count. Most of the lab relied heavily on the previous labs, which made it easier.

5. Study Questions

1. 307200 pixels are displayed. I got this number by multiplying the width of the display by the height, so 640 pixels times 480 pixels.
2. The video driver is 25 MHz, which is from the divided input of the 50 MHz clock input.
3. There are 6 switches used, so the display would be able to show 720 colors. This is found by calculating 6 factorial.

6. Appendix

```
module countersix#(parameter N=8)(input logic clk, input logic
reset, output logic [N-1:0] q);

      always_ff @(posedge clk, posedge reset)
            if(reset) q <=0;
            else q <= q+1;
endmodule
```

Figure 5: Counter

This is the counter SystemVerilog code.

```
module comparatorrgb #(parameter b = 96, N = 10)(input logic [N-
1:0] a, output logic eq, neq, lt, lte, gt, gte);

      assign eq = (a == b);
      assign neq = (a != b);
      assign lt = (a < b);
      assign lte = (a <= b);
      assign gt = (a > b);
      assign gte = (a >= b);
endmodule
```

Figure 6: Comparator

This is the comparator SystemVerilog code.

```
module vgadecoder(input logic [1:0] data, output logic [3:0] seg);

      always_comb
            case(data)
                  0: seg=4'b0000;
                  1: seg=4'b0101;
                  2: seg=4'b1010;
                  3: seg=4'b1111;
            endcase
endmodule
```

Figure 7: Decoder
This is the decoder SystemVerilog code.

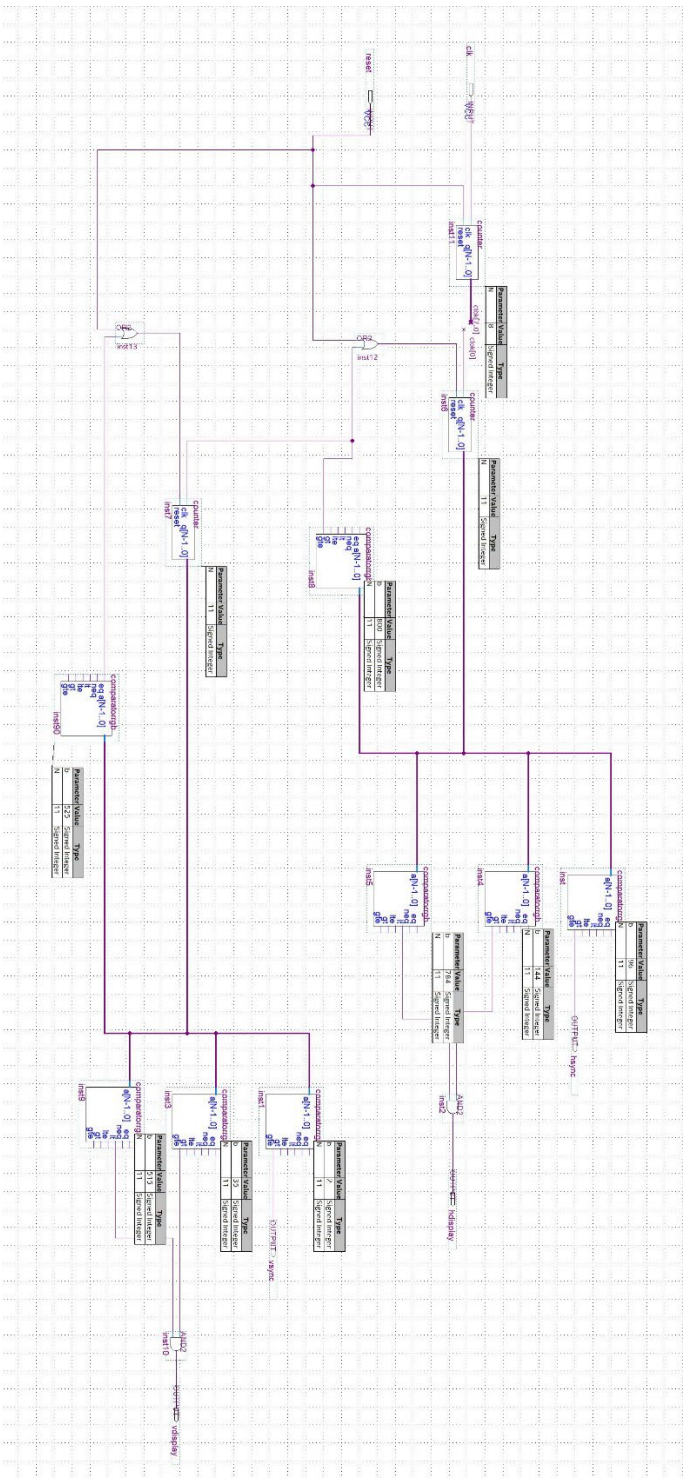

Figure 8: Sync Count Blown Up
This is Figure 3 blown up.

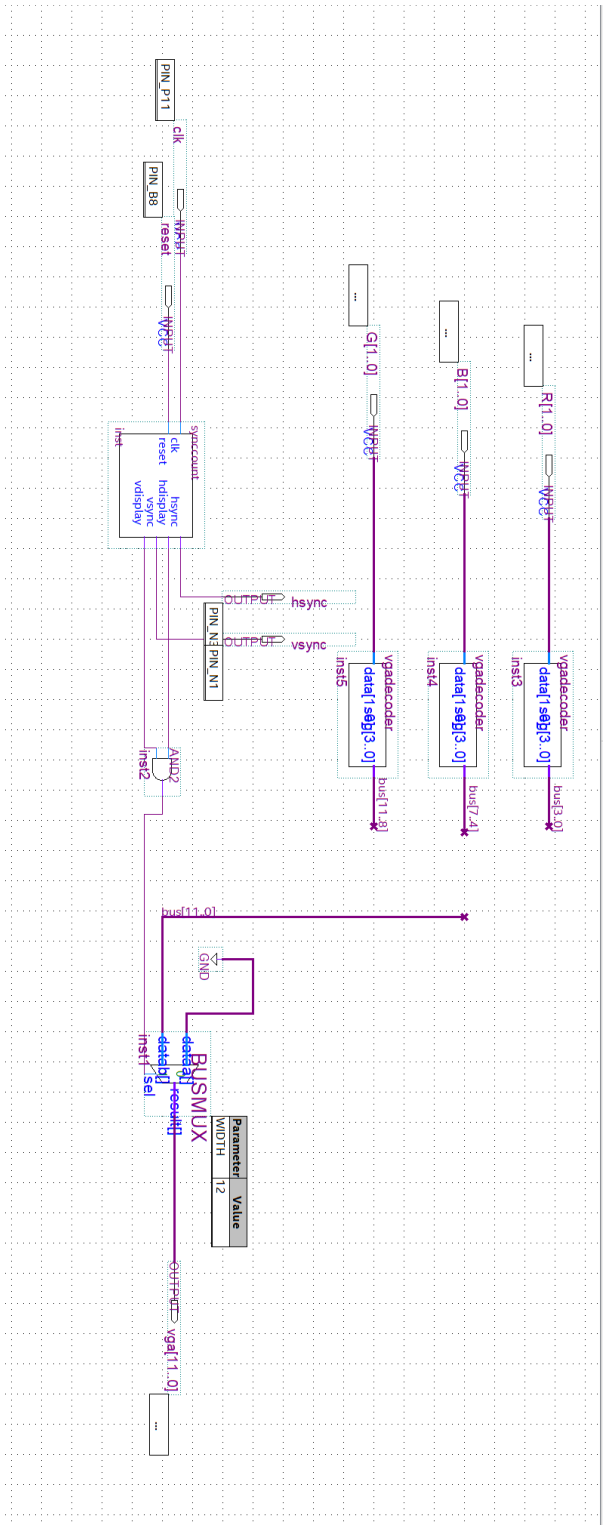

Figure 9: Top Level Blown Up
This is Figure 4 blown up.