FAMU - FSU College of Engineering

Department of Electrical and Computer Engineering
Fall 2024 Semester

EEL 4710L – Intro to VHDL Lab Report

Section No: 02

Lab Instructor: Dr. Uwe Meyer-Baese

Lab No: 10

Lab Title: Project

Name: Ruth Massock
Partner's Name: Keila Souriac

Date Performed: 11/08/2024

Date Delivered: 12/04/2024

Contents

1 Introduction

This lab aimed to implement a VGA controller using the DE-1 FPGA board to explore VGA interface concepts. Specifically, the project involved configuring the VGA display to show red, green, or blue colors based on switch inputs, along with a 7-segment HEX display to indicate the selected color. This work provided hands-on experience with VHDL design, synchronization protocols, and interfacing digital and analog signals via the DE-1 FPGA board.

2 Requirements

The project required the following design elements inputs and outputs specified below:

Signal	Direction	Width	description
SW[2:0]	Input	3 bits	Switches used to select the displayed color (red, green, or blue). Active high.
VGA_R[7:0]	Output	8 bits	Represents the red color intensity for the VGA display.
VGA_G[7:0]	Output	8 bits	Represents the green color intensity for the VGA display.
VGA_B[7:0]	Output	8 bits	Represents the blue color intensity for the VGA display.
VGA_CLK	Output	1 bit	Pixel clock for driving the VGA display.
VGA_SYNC_N	Output	1 bit	Syncs with the DAC to switch between green and RGB DAC. Active low.
VGA_BLANK_N	Output	1 bit	Active low during retrace periods.
HEX0-5[6:0]	Output	7 bits	Displays “R”, “G”, or “B” based on the selected color. And also or initials

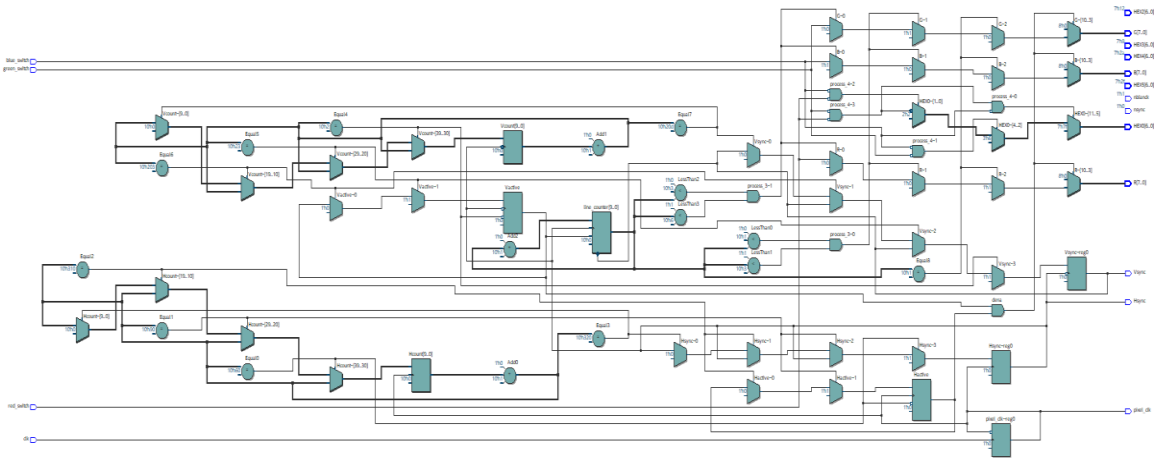
3 Theoretical Design

3.1 Design Narrative

The VGA controller generates color signals (VGA_R, VGA_G, VGA_B) for display, synchronized with horizontal and vertical sync signals. The Pixel Clock (VGA_CLK) drives the timing. The switches (SW[2:0]) determine the active color, while the 7-segment display (HEX0) indicates the selected color. The design utilizes VHDL to describe behavior and connect signals.

3.2 Top-level design

Below is a block diagram depicting the top-level design



Components Description

VGA Controller

Inputs: SW[2:0], VGA_CLK

Outputs: VGA_R[7:0], VGA_G[7:0], VGA_B[7:0], VGA_SYNC_N, VGA_BLANK_N

Function: Decodes switch inputs to set the appropriate color signals and generates synchronization signals.

HEX Display Controller

Inputs: SW[2:0]

Outputs: HEX0[6:0]

Function: Maps switch values to display “R”, “G”, or “B” for selected color or turns off for invalid inputs.

4 Synthesized Design

Below is the VHDL code used for the VGA controller and HEX display modules.

VGA Controller Code

```
1  ----------------------------------------------------------------------------
2  -- Title       : Project - VGA Controller
3  -- Author      : Keila Souriac & Ruth Massock
4  -- Date        : 11/08/2024
5  -- Description : This program is the driver for the VGA interface on the
6  --               DE-1 FPGA board.
7  ----------------------------------------------------------------------------
8    LIBRARY ieee;
9    USE ieee.std_logic_1164.all;
10   ------------------------------------------------------------
11 ENTITY Project IS
12 GENERIC (
13   Ha: INTEGER := 96;   --Hpulse
14   Hb: INTEGER := 144;  --Hpulse+HBP
15   Hc: INTEGER := 784;  --Hpulse+HBP+Hactive
16   Hd: INTEGER := 800;  --Hpulse+HBP+Hactive+HFP
17   Va: INTEGER := 2;    --Vpulse
18   Vb: INTEGER := 35;   --Vpulse+VBP
19   Vc: INTEGER := 515;  --Vpulse+VBP+Vactive
20   Vd: INTEGER := 525); --Vpulse+VBP+Vactive+VFP
21 PORT (
22   clk: IN STD_LOGIC; --50MHz in our board
23   red_switch, green_switch, blue_switch: IN STD_LOGIC;
24   pixel_clk: BUFFER STD_LOGIC;
25   Hsync, Vsync: BUFFER STD_LOGIC;
26   R, G, B: OUT STD_LOGIC_VECTOR(7 DOWNTO 0);
27   HEX0,HEX2,HEX3,HEX4,HEX5: OUT STD_LOGIC_VECTOR(6 DOWNTO 0);
28   nblanck, nsync : OUT STD_LOGIC);
29 END Project;
30   ------------------------------------------------------------
31 ARCHITECTURE vga OF Project IS
32 SIGNAL Hactive, Vactive, dena: STD_LOGIC;
33 ------HEX display signals------------
34   signal M: STD_LOGIC_VECTOR(6 DOWNTO 0):="0101010";
35   signal K: STD_LOGIC_VECTOR(6 DOWNTO 0):="0001001"; -- Letter H as a K
36   signal S: STD_LOGIC_VECTOR(6 DOWNTO 0):="0010010"; -- letter S
37   signal O: STD_LOGIC_VECTOR(6 DOWNTO 0):="1111111"; -- all segments off
38   signal Red  : STD_LOGIC_VECTOR(6 DOWNTO 0):="0101111";   --Lowercase r (RUTH & RED)
39   signal Green: STD_LOGIC_VECTOR(6 DOWNTO 0):="0000010"; -- Letter G (GREEN)
40   signal Blue : STD_LOGIC_VECTOR(6 DOWNTO 0):="0000000"; -- Letter B (BLUE)
41 BEGIN
```

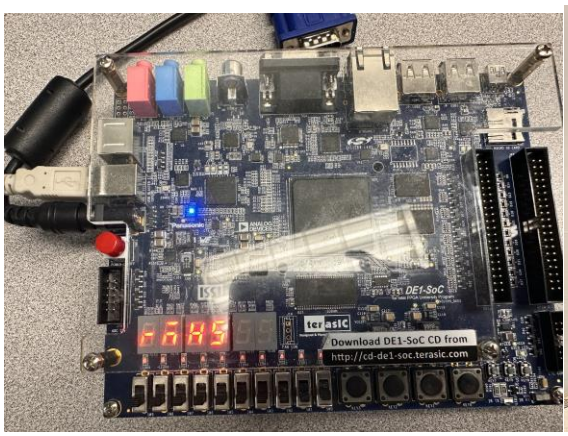
```
42  ----------------------------------------------------------
43  --Part 1: CONTROL GENERATOR
44  ----------------------------------------------------------
45  --Static signals for DACs:
46  nblanck <= '1'; --no direct blanking
47  nsync <= '0'; --no sync on green
48  --Create pixel clock (50MHz->25MHz):
49  PROCESS (clk)
50  BEGIN
51  IF (clk'EVENT AND clk='1') THEN
52  pixel_clk <= NOT pixel_clk;
53  END IF;
54  END PROCESS;
55  --Horizontal signals generation:
56  PROCESS (pixel_clk)
57  VARIABLE Hcount: INTEGER RANGE 0 TO Hd;
58  BEGIN
59  IF (pixel_clk'EVENT AND pixel_clk='1') THEN
60  Hcount := Hcount + 1;
61  IF (Hcount=Ha) THEN
62  Hsync <= '1';
63  ELSIF (Hcount=Hb) THEN
64  Hactive <= '1';
65  ELSIF (Hcount=Hc) THEN
66  Hactive <= '0';
67  ELSIF (Hcount=Hd) THEN
68  Hsync <= '0';
69  Hcount := 0;
70  END IF;
71  END IF;
72  END PROCESS;
73  --Vertical signals generation:
74  PROCESS (Hsync)
75  VARIABLE Vcount: INTEGER RANGE 0 TO Vd;
76  BEGIN
77  IF (Hsync'EVENT AND Hsync='0') THEN
78  Vcount := Vcount + 1;
79  IF (Vcount=Va) THEN
80  Vsync <= '1';
81  ELSIF (Vcount=Vb) THEN
82  Vactive <= '1';
83  ELSIF (Vcount=Vc) THEN
84  Vactive <= '0';
85  ELSIF (Vcount=Vd) THEN
86  Vsync <= '0';
87  Vcount := 0;
88  END IF;
89  END IF;
```

```
90    END PROCESS;
91    ---Display enable generation:
92    dena <= Hactive AND Vactive;
93    -------------------------------------------------------
94    --Part 2: IMAGE GENERATOR
95    -------------------------------------------------------
96    PROCESS (Hsync, Vsync, Vactive, dena, red_switch,
97    green_switch, blue_switch)
98    VARIABLE line_counter: INTEGER RANGE 0 TO Vc;
99    BEGIN
100   IF (Vsync='0') THEN
101   line_counter := 0;
102   ELSIF (Hsync'EVENT AND Hsync='1') THEN
103   IF (Vactive='1') THEN
104   line_counter := line_counter + 1;
105   END IF;
106   END IF;
107   IF (dena='1') THEN
108   IF (line_counter=1) THEN
109   R <= (OTHERS => '1');
110   G <= (OTHERS => '0');
111   B <= (OTHERS => '0');
112   ELSIF (line_counter>1 AND line_counter<=3) THEN
113   R <= (OTHERS => '0');
114   G <= (OTHERS => '1');
115   B <= (OTHERS => '0');
116   ELSIF (line_counter>3 AND line_counter<=6) THEN
117   R <= (OTHERS => '0');
118   G <= (OTHERS => '0');
119   B <= (OTHERS => '1');
120   ELSE
121   R <= (OTHERS => red_switch);
122   G <= (OTHERS => green_switch);
123   B <= (OTHERS => blue_switch);
124   END IF;
125   ELSE
126   R <= (OTHERS => '0');
127   G <= (OTHERS => '0');
128   B <= (OTHERS => '0');
129   END IF;
130   END PROCESS;
131   -------- case for hex display---------------
132   PROCESS (red_switch,
133   green_switch, blue_switch)
134   BEGIN
135   ----Display our initials-------
136   HEX5 <= Red;
137   HEX4 <= M;
```

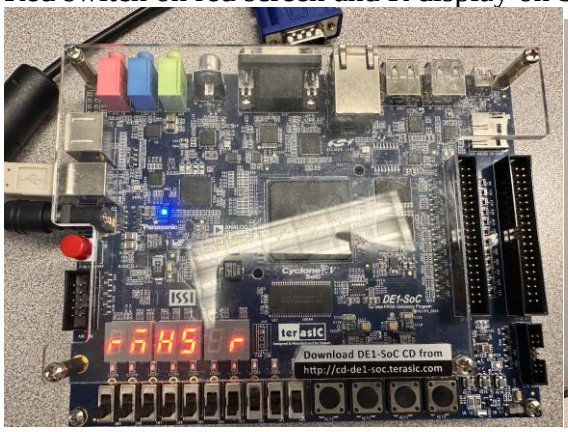
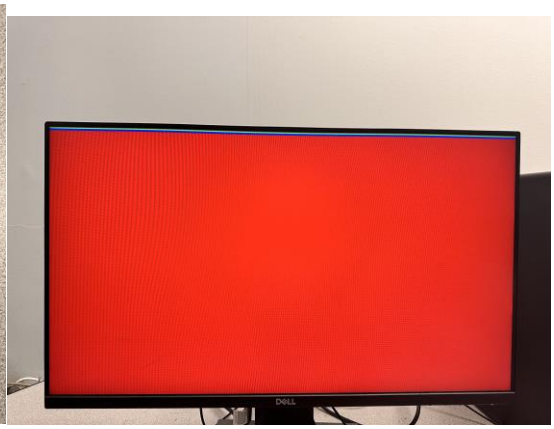
5 Results

On the FPGA board, the monitor displayed red, green, and blue colors successfully. The 7-segment HEX display correctly indicated "R", "G", or "B" based on the selected color, it also displays our initials on hex display “rmks”.

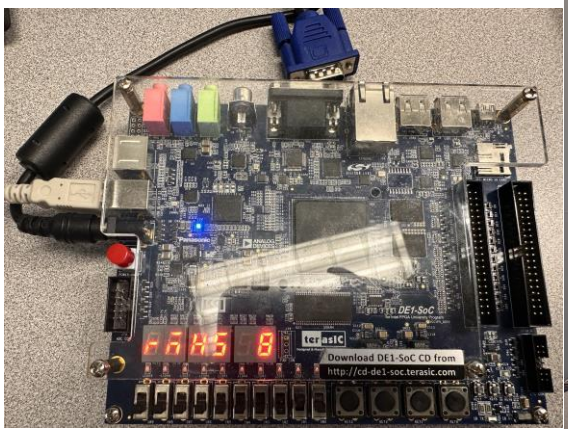
Switches are off and so black screen



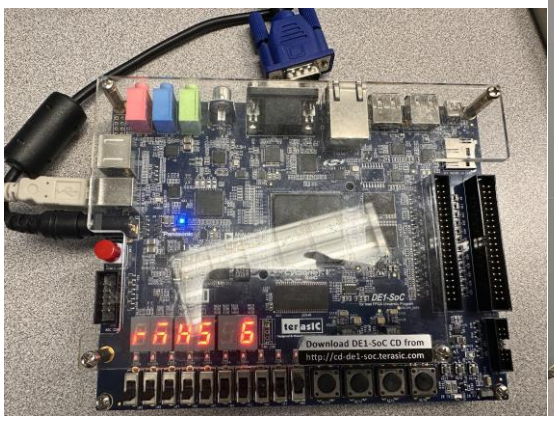

Red switch on red screen and R display on board

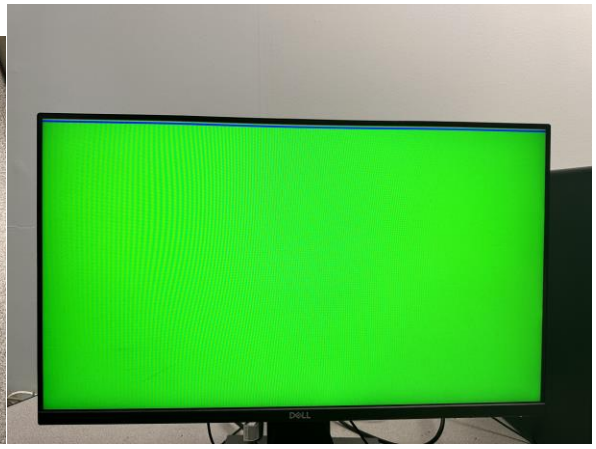



Blue Switch on blue screen and B display on board

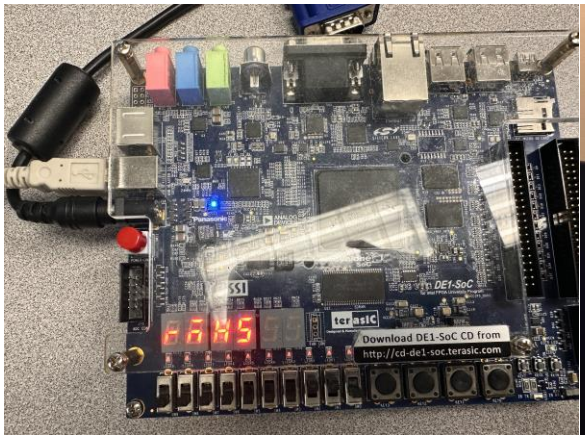



Green switch on Green screen and G display on board

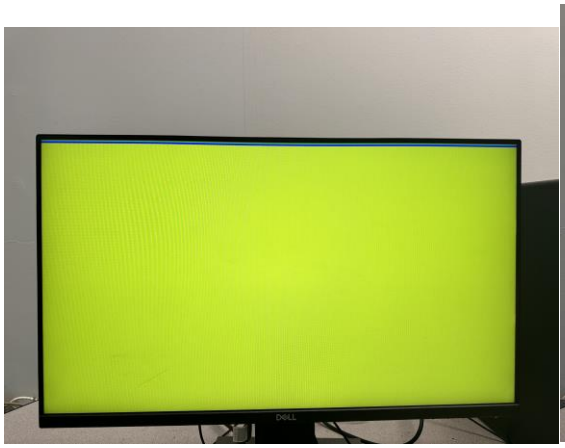




All switches on White



Green and Red switch ON



Blue and Green Switch ON

Red and Blue Switch ON

6 Summary and Lessons Learned

The project successfully implemented a VGA controller and HEX display interface. Observations include:

- Correct synchronization between the DE-1 FPGA and the VGA display.
- Challenges in aligning timing constraints between horizontal and vertical sync signals were resolved by adjusting the Pixel Clock frequency.

Key lessons:

- Proper pin assignments and clock signal configuration are critical in FPGA designs.
- Understanding VGA timing diagrams is essential for display interfacing.

Future recommendations:

- Implement advanced VGA patterns, such as gradients or animations, to further test VGA capabilities.
- Utilize additional debugging tools to streamline the design process.

END OF DOCUMENT