Advanced Microelectronics Lab

(ELCT 1005)

Lab Report 1

Mohamed Tarek 49-7879

Martel Megalaa 49-3843

Abdelrahman Ayman 52-15037

Introduction:

Over the years, as the demand for enhanced graphics performance and flexibility has grown, the integration of VGA with Field-Programmable Gate Arrays (FPGAs) has emerged as a powerful solution.

It is required in this lab to control the LCD screen (640x480) to display a fixed color using RGB combinations and an image of your choice to be displayed in place on the screen.

Video Graphics Array (VGA), is used as an interface between the FPGA and the LCD screen. It controls the signals timing on the monitor using the Horizontal-sync for the scan line process and Vertical-sync for the frame process.

Task 1:

A. Methodology and Implementation

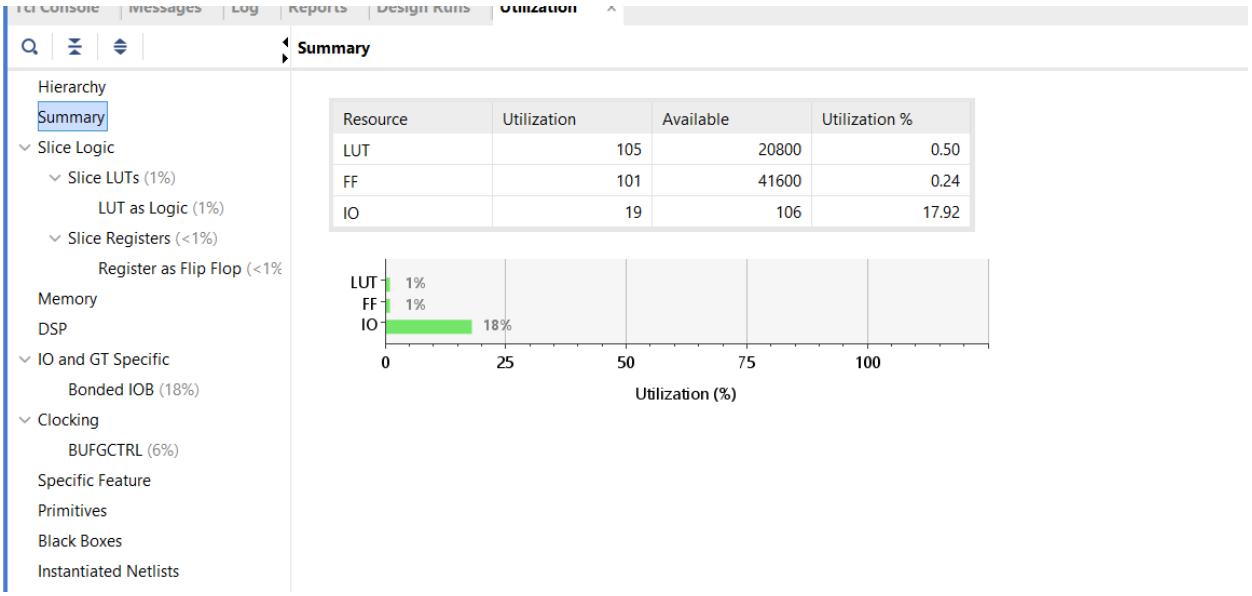
First, a VGA controller module is implemented to control the signals and their timings. 5 inputs (clk, reset, Rin, Gin, Bin) and 5 outputs (R, G, B, hsync, vsync) were used. Because the VGA operates only at 25MHz while the FPGA (basys3) operates at 100MHz, a clock divider has been used in such way that every 2 clock cycles (rising edge) correspond to half a cycle in the 25MHz clock (every 2 clock cycles of 100MHz the 25MHz clock signal changes). The three switches Rin, Gin, and Bin were responsible for the switching of the 8 different color combinations. For the outputs R, G, and B each represent 4 bits. Hsync and vsync outputs are used as signals for writing each pixel on the screen. To understand how this works, pixels start getting written from top left moving horizontally. After each row ends, hsync signal is used to increment the line row and starting writing the pixels. Once all the rows are done, the vsync signal is used to repeat the whole cycle with a new clear frame (simulations were run to ensure timings are working properly and clock cycles are right).

2 processes were used one for the vsync while the other for the hsync. Constants of the waveforms, sync pulse time, display time, pulse width, front porch, and back porch were all used following their respective clock cycles. To conclude, whenever both the vsync and the hsync are in their display mode, RGB outputs can be displayed depending on the input switches.

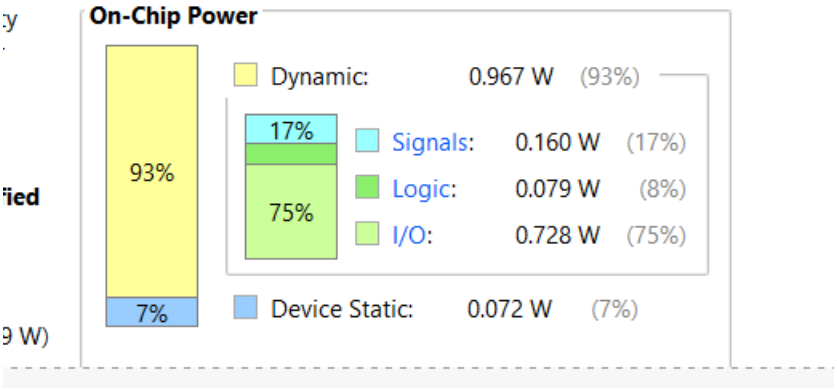
B. Output

Depending on the switch inputs, the color on the screen changes.
(File attached in the drive)

C. Area Utilization



D. Power Utilization



Task 2:

A. Methodology and Implementation

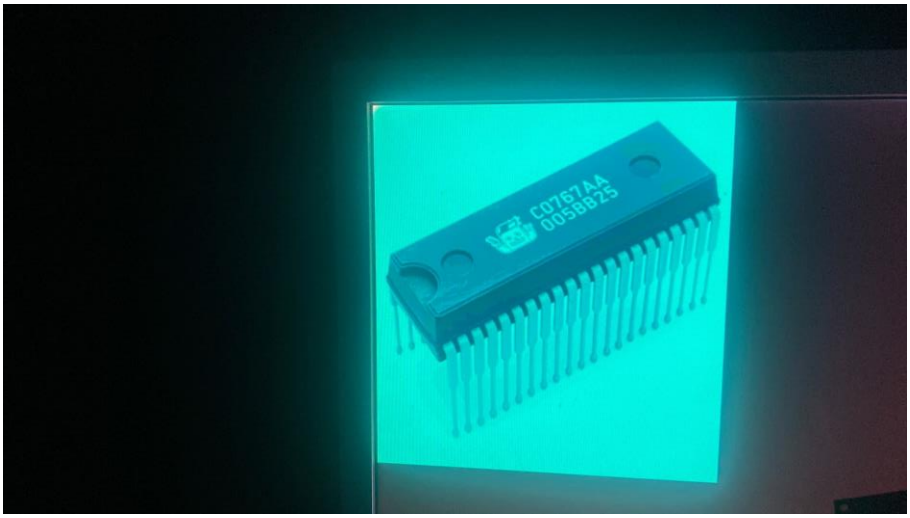
Primary, an image is converted to a coe file with each pixel representing a 12-bit RGB representation either using MATLAB or Python code (MATLAB was used).

Due to the FPGA's limited memory, a restricted 300x300 image could be used.

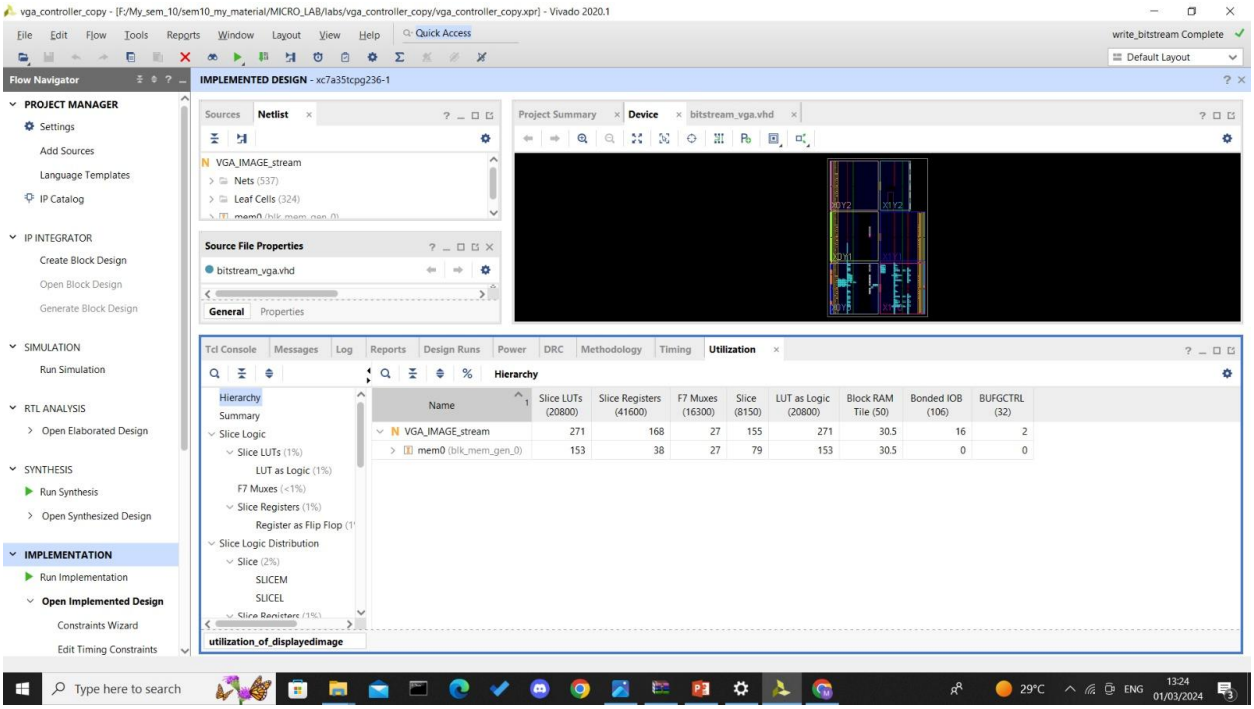
(300×300) × 12 bits = 90000 × 12 bits

Nothing much changed from the first task except that display is on for only the first 300 pixels while the rest are set as 0's (black). Once all the 90000 pixels are read and the normal full vsync and hsync cycles complete (640x480) the image is displayed on the screen.

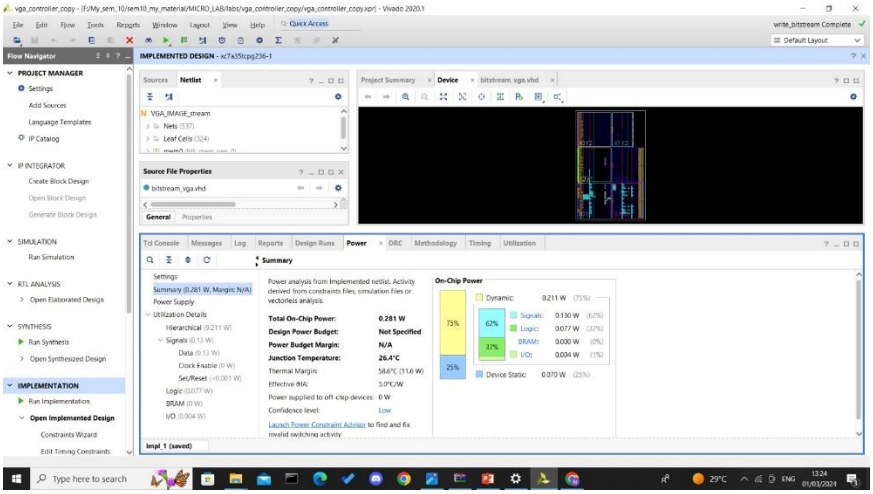
B. Output



C. Area Utilization



D. Power Utilization



CONCLUSION:

A fixed image was successfully displayed on a VGA 640x480 display thanks to the successful implementation of VHDL code. The image was successfully displayed on the VGA screen by assigning the correct RGB values to each pixel. Working on the VGA can be a bit confusing but at the end it is quite rewarding.

Facing certain difficulties such as the FPGA's limited memory and assigning the correct vsync and hsync signals clock cycles.

Expecting that the image display would use more area and power than displaying a single color on the screen even though the image is on a smaller scale.

APPENDIX:

Task1:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;
use ieee.numeric_std.all;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity VGA_Module is
    Port ( clk : in STD_LOGIC;
    reset : in std_logic;
    Rin,Gin,Bin: in std_logic;
    R,G,B: out std_logic_vector(3 downto 0) ;
    hsync,vsync:out std_logic);

end VGA_Module;

architecture Behavioral of VGA_Module is


signal clk_out1: std_logic:='1';
signal hsyn,vsyn:std_logic;
signal vdisplay,hdisplay:std_logic:='1';

begin

hsync <= hsyn;
vsync<=vsyn;


clocking:process (clk)
variable count:integer := 0;
begin
if rising_edge(clk) then
    count := count+1;
    if count = 2 then
        clk_out1 <= not clk_out1;
        count := 0;
    end if;
end if;
end process;



scan_line:process(clk_out1,reset)
    constant TFP: integer:= 16;
    constant TDISP : integer:= 640;
    constant TPW:  integer:=96;
    constant TBP: integer:=48;
    variable Hcount : integer:=0;
```

```
begin
    if (reset ='1') then
        Hcount := 0;
        hsyn <= '1';
        hdisplay <= '1';
    elsif rising_edge(clk_out1) then

        Hcount := Hcount+1;
        if (Hcount = TDISP ) then
            hdisplay <= '0';
        end if;
        if (Hcount = TDISP+TFP ) then
            hsyn <= '0';
        end if;
        if (Hcount = TDISP+TFP+TPW) then
            hsyn <= '1';
        end if;

        if (Hcount = TFP+TPW+TBP+TDISP) then
            Hcount := 0;
            hsyn <='1';
            hdisplay <= '1';

        END IF;

    end if;
end process;


process(clk_out1,reset)
    constant TFP: integer:= 8000;
```

```
    constant TVDISP300 : integer:= 240000;
    constant TDISP: integer:= 384000;
    constant TPW:  integer:= 1600;
    constant TBP: integer:= 26400;
    variable Vcount : integer:= 0;
begin
    if (reset ='1') then
        Vcount := 0;
        vsyn <= '1';
        vdisplay <= '1';
    elsif rising_edge(clk_out1) then
        Vcount := Vcount+1;
        if (Vcount = TVDISP300 ) then
            vdisplay <= '0';
        end if;
        if (Vcount = TDISP+TFP ) then
            vsyn <= '0';
        end if;
        if (Vcount = TDISP+TFP+TPW) then
            vsyn <= '1';
        end if;


        if (Vcount = TFP+TPW+TBP+TDISP) then
            Vdisplay <= '1';
            Vcount := 0;
            vsyn <='1';
```

```
        END IF;

    end if;
end process;
process(vdisplay,hdisplay,reset,clk_out1)


begin
if (vdisplay='1' and hdisplay='1' and reset ='0' )then --and rising_edge(clk_out1)) then
--




                    if (Rin = '1') then
                        R <= "1111";
                    elsif (Rin = '0') then
                        R <= "0000";
                    end if;
----------------------GREEN-----------------------------------------------------------
                    if (Gin = '1') then
                        G <= "1111";
                    elsif (Gin = '0') then
                        G <= "0000";
                    end if;
----------------------BLUE------------------------------------------------------------
                    if (Bin = '1') then
                        B <= "1111";
                    elsif (Bin = '0') then
                        B <= "0000";
                    end if;
```

```
ELSE
                    r <= (others => '0');
                    g <= (others => '0');
                    b <= (others => '0');
end if ;
end process;
end Behavioral;
```

Task2:

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24  use ieee.std_logic_unsigned.all;
25  use ieee.numeric_std.all;
26
27  -- Uncomment the following library declaration if using
28  -- arithmetic functions with Signed or Unsigned values
29  --use IEEE.NUMERIC_STD.ALL;
30
31  -- Uncomment the following library declaration if instantia
32  -- any Xilinx leaf cells in this code.
33  --library UNISIM;
34  --use UNISIM.VComponents.all;
35
36  entity VGA_IMAGE_stream is
37      Port ( clk : in STD_LOGIC;
38      reset : in std_logic;
39      Rin,Gin,Bin: in std_logic;
40      R,G,B: out std_logic_vector(3 downto 0) ;
41      hsync,vsync:out std_logic);
42      --clk_out,
43      --flag:out std_logic;
44      --dout1: out std_logic_vector(0 downto 0);
45      --addr:out std_logic_vector(16 downto 0));
46  end VGA_IMAGE_stream;
47
48  architecture Behavioral of VGA_IMAGE_stream is
49
50
51  component blk_mem_gen_0 IS
52    PORT (
```

```
    clka : IN STD_LOGIC;
    wea : IN STD_LOGIC_VECTOR(0 DOWNTO 0);
    addra : IN STD_LOGIC_VECTOR (16 DOWNTO 0);
    dina : IN STD_LOGIC_VECTOR(11 DOWNTO 0);
    douta : OUT STD_LOGIC_VECTOR(11 DOWNTO 0)
  );
END component;


signal clk_out1: std_logic:='1';
signal hsyn,vsyn:std_logic;
signal vdisplay,hdisplay:std_logic:='1';
signal address:std_logic_vector(16 downto 0):=(others => '0');
signal dout:std_logic_vector(11 downto 0);

begin

--addr <= address;
--vdisp <= vdisplay;
--hdisp <= hdisplay;
--clk_out<=clk_out1;

hsync <= hsyn;
vsync<=vsyn;
mem0: blk_mem_gen_0 port map(clka => clk_out1,wea =>(others => '0'),addra => address,dina => (others => '0'),douta =>dout);

--flag<=vdisplay and hdisplay;

clocking:process (clk)
```

```
81  clocking:process (clk)
82  variable count:integer := 0;
83  begin
84  if rising_edge(clk) then
85      count := count+1;
86      if count = 2 then
87          clk_out1 <= not clk_out1;
88          count := 0;
89      end if;
90  end if;
91  end process;
92
93
94
95  scan_line:process(clk_out1,reset)
96      constant TFP: integer:= 16;
97      constant TDISP300 : integer:= 300;
98      constant TDISP : integer:= 640;
99      constant TPW:  integer:=96;
00      constant TBP: integer:=48;
01      variable Hcount : integer:=0;
02  begin
03      if (reset ='1') then
04          Hcount := 0;
05          hsyn <= '1';
06          hdisplay <= '1';
07      elsif rising_edge(clk_out1) then
08
09          Hcount := Hcount+1;
10          if (Hcount = TDISP300 ) then
11              hdisplay <= '0';
```

```
            hdisplay <= '0';
        end if;
        if (Hcount = TDISP+TFP ) then
            hsyn <= '0';
        end if;
        if (Hcount = TDISP+TFP+TPW) then
            hsyn <= '1';
        end if;

        if (Hcount = TFP+TPW+TBP+TDISP) then
            Hcount := 0;
            hsyn <='1';
            hdisplay <= '1';

        END IF;

    end if;
end process;


process(clk_out1,reset)
    constant TFP: integer:= 8000;
    constant TVDISP300 : integer:= 240000;
    constant TDISP: integer:= 384000;
    constant TPW:  integer:= 1600;
    constant TBP: integer:= 26400;
    variable Vcount : integer:= 0;
begin
    if (reset ='1') then
        Vcount := 0;
        vsyn <= '1';
```

```
            vdisplay <= '1';
        elsif rising_edge(clk_out1) then
            Vcount := Vcount+1;
            if (Vcount = TVDISP300 ) then
                vdisplay <= '0';
            end if;
            if (Vcount = TDISP+TFP ) then
                vsyn <= '0';
            end if;
            if (Vcount = TDISP+TFP+TPW) then
                vsyn <= '1';
            end if;


            if (Vcount = TFP+TPW+TBP+TDISP) then
                Vdisplay <= '1';
                Vcount := 0;
                vsyn <='1';

            END IF;

        end if;
    end process;
    process(vdisplay,hdisplay,reset,clk_out1)
    --const int ivVal[] = {33, 44, 55, 66};
    variable hcount: integer:=0;
    variable vcount: integer:=0;
    variable upper: integer:=90000;
    begin
    if (vdisplay='1' and hdisplay='1' and reset ='0' )then --and rising_edge(clk_out1)) then
    --
```

```
--

    if rising_edge(clk_out1) then

                    address <= address+1;
----
                    if (unsigned(address) >= "10101111110001111") or (vcount = 90000) then
----                    report("weselt");
                         address <= (others => '0');

                    end if;
                     R <= dout(11 downto 8);
                     G <= dout(7 downto 4);
                     B <= dout(3 downto 0);
--                  address <= (others => '0');
   end if;

ELSE
                r <= (others => '0');
                g <= (others => '0');
                b <= (others => '0');
end if ;
end process;
end Behavioral;
```