Universidad Nacional Autónoma de México

Facultad de Ingeniería

Diseño Digital VLSI

Semestre 2020-1

Practica 4 “FPGA y VGA”

Grupo 05

Chavez Troncoso Hector Manuel

Garciarebollo Rojas Diego Iñaki

Moreno Osuna Isaac

Rubio Carmona Alonso Rafael

Cd. Universitaria a 30 de septiembre de 2019

Planteamiento del Problema:
Utilizando una tarjeta FPGA y un monitor con entrada estándar de VGA, realizar estados diferentes para mostrar diferentes cosas en pantalla, pasando de un cuadro, una línea recta, una línea a 45° y una circunferencia utilizando como base el código proporcionado por el profesor al inicio de la sesión.

Objetivo:
Comprender el funcionamiento del estándar VGA y enviar una señal de video digital a través del puerto VGA.

Metodologia:

Para la elaboración de las distintas pantallas que se mostraban en el monitor, se implementó el uso de ecuaciones matemáticas básicas cuando era necesario; tal es el caso de cuando se muestra una circunferencia o un círculo, en la cual se implementa la ecuación general de la circunferencia, pero con diferentes restricciones o en el caso de cuando se muestra una recta a 45 grados, donde también utilizamos su ecuación general. Para cuando en pantalla se tenía que mostrar con un color diferente, solo modificamos los valores de la señal que lleva el color hacia el monitor.

En cuanto a cuando se tenía que mostrar un cuadrado, lo que hicimos fue hacer una restricción para que en cierta posición de la pantalla solo se coloreara de un diferente color con respecto al fondo. Al momento de asignar movimiento al cuadrado, lo que hicimos fue poner un contador, el cual va a ir modificando la posición donde se muestra el cuadrado. Con respecto a la implementación de la recta que gira, se muestran varias rectas la cuales se intercalan con respecto a la posición de la anterior recta y se muestran durante un cierto de rango de cuadros por segundo, esto con ayuda de un reloj.

Arquitectura del sistema:

Código Fuente:

```
library ieee;
use ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;
use ieee.numeric_std.all;

entity CONTROLADOR_VGA is
port( clk, reset, jump : in std_logic;
           sel : in STD_LOGIC_VECTOR(3 downto 0);
           hsync, vsync : out std_logic;
           VGA_R, VGA_G, VGA_B: out std_logic_vector(3 downto 0));
```

```
end CONTROLADOR_VGA;

architecture COMP_CONTROLADOR_VGA of CONTROLADOR_VGA is

signal div2 : std_logic := '0';
signal pos_x : integer range 0 to 640 := 640;
signal pos_y : integer range 0 to 480 := 480;
signal habilitado : std_logic;

signal clk_40Hz : std_logic := '0';
signal clk_20Hz : std_logic := '0';

signal div20 : integer range 0 to 2500000 := 0;
signal div40 : integer range 0 to 1250000 := 0;
signal tempo : integer range 0 to 540 := 0;
signal grado : integer range 0 to 35 := 0;

component SINC_VGA
port (clk : in std_logic;
          hsync, vsync, habilitado : out std_logic;
          pos_x : out integer range 0 to 800 := 0;
          pos_y : out integer range 0 to 525 := 0);
end component;

begin
     sincronizador: SINC_VGA port map (div2, hsync, vsync,
habilitado, pos_x, pos_y);

     process(clk)
     begin
          if rising_edge(clk) then
               div2 <= not div2;
               if (div40 < 1250000) then
                    div40 <= div40 + 1;
               else
                    div40 <= 0;
                    clk_40Hz <= NOT clk_40Hz;
               end if;
               if (div20 < 2500000) then
                    div20 <= div20 + 1;
               else
                    div20 <= 0;
                    clk_20Hz <= NOT clk_20Hz;
```

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

    process(clk_40Hz)
    begin
        If clk_40Hz'event AND clk_40Hz = '1' then
            if (tempo < 541) then
                tempo <= tempo + 1;
            else
                tempo <= 0;
            end if;
        end if;
    end process;

    process(clk_20Hz)
    begin
        If clk_20Hz'event AND clk_20Hz = '1' then
            if (grado < 4) then
                grado <= grado + 1;
            else
                grado <= 0;
            end if;
        end if;
    end process;

    senialVGA: process(div2)

    variable colorRGB : std_logic_vector(11 downto 0) := (others =>
'0');

    begin
            if rising_edge(div2) then
                if habilitado = '1' then
                    Case sel is
                    when "0000" => -- Pantalla roja
                    colorRGB := "1111"&"0000"&"0000"; -- Rojo

                    when "0001" => -- Pantalla azul cielo
                    colorRGB := "0101"&"1101"&"1111"; -- Azul
cielo
                    when "0011" => -- Pantalla rosa
                    colorRGB := "1111"&"0000"&"1000"; -- Rosa
```

```
                        when "0010" => -- Pantalla negra
                        colorRGB := "0000"&"0000"&"0000"; -- Negro

                        when "0110" => -- cuadro rojo en el centro
                        if (pos_x > 280 AND pos_x < 360 AND pos_y
> 200 AND pos_y < 280  ) then
                              colorRGB := "1111"&"0000"&"0000";
                        else
                              colorRGB := "0000"&"0000"&"0000";
                        end if;

                        when "0111" => -- linea horizontal de 5
pixeles de ancho
                        if (pos_x > 120 AND pos_x < 400 AND pos_y
> 200 AND pos_y < 206  ) then
                              colorRGB := "0000"&"0101"&"0001";
                        else
                              colorRGB := "1110"&"1110"&"1110";
                        end if;

                        when "0101" => -- linea a 45 grados de 5
pixeles de ancho
                        if (pos_y <= (480-pos_x) AND pos_y >=
(480-(pos_x+5))) then
                        colorRGB := "1111"&"0000"&"0000";
                        else
                        colorRGB := "1110"&"1110"&"1110";
                        end if;

                        when "0100" => -- Circulo en el centro
--                      mitad x => 640/2 = 320
--                      mitad y => 480/2 = 240

                        if((pos_x-320)*(pos_x-320) +
(pos_y-240)*(pos_y-240) <= 10000) then
                              colorRGB := "1111"&"1100"&"0000";
                        else
                              colorRGB := "1110"&"1110"&"1110";
                        end if;

                        when "1100" => -- Circunferencia de 5
pixeles de radio.
```

```
--                          mitad x => 640/2 = 320
--                          mitad y => 480/2 = 240

                            if(((pos_x-320)*(pos_x-320) +
(pos_y-240)*(pos_y-240) <= 10000)
                            AND((pos_x-320)*(pos_x-320) +
(pos_y-240)*(pos_y-240) >= 9025 )) then
                                 colorRGB := "1111"&"1100"&"0000";
                            else
                                 colorRGB := "1110"&"1110"&"1110";
                            end if;

                            when "1101" => -- Cuadro en movimiento de
izquierda a derecha
                            if((pos_x > (0 + tempo) AND pos_x <
(100+tempo))AND(pos_y > 190 AND pos_y < 290)) then
                                 colorRGB := "1111"&"0000"&"0000";
                            else
                                 colorRGB := "1111"&"1111"&"1111";
                            end if;

                            when "1111" => -- Recta que gira en el
centro
--                          mitad x => 640/2 = 320
--                          mitad y => 480/2 = 240
                       case grado is
                            when 0 =>
                            if (pos_y > 236 AND pos_y < 244) then
                                 colorRGB := "1111"&"0000"&"0000";
                            else
                                 colorRGB := "1111"&"1111"&"1111";
                            end if;

                            when 1 =>
                            if ((pos_y <= (480-pos_x)+80) AND (pos_y
>= (480-(pos_x+5)+80))) then
                                 colorRGB := "1111"&"0000"&"0000";
                            else
                                 colorRGB := "1110"&"1110"&"1110";
                            end if;

                            when 2 =>
                            if (pos_x > 316 AND pos_x < 324) then
```

```
                              colorRGB := "1111"&"0000"&"0000";
                         else
                              colorRGB := "1111"&"1111"&"1111";
                         end if;

                         when others =>
                         if ((pos_y <= pos_x - 80) AND (pos_y >=
(pos_x - 85))) then
                              colorRGB := "1111"&"0000"&"0000";
                         else
                              colorRGB := "1110"&"1110"&"1110";
                         end if;
                         end case;

                    when others =>
                         colorRGB := "0000"&"0000"&"1111"; --
Pantallazo azul
                         END CASE;

                         VGA_R <= colorRGB(11 downto 8);
                         VGA_G <= colorRGB(7 downto 4);
                         VGA_B <= colorRGB(3 downto 0);
                    else
                         VGA_R <= "0000";
                         VGA_G <= "0000";
                         VGA_B <= "0000";
                    end if;
               end if;
     end process;
end COMP_CONTROLADOR_VGA;
```

Resultados:

https://youtu.be/zl6e_ZmFdp8

Conclusiones

En esta práctica se lograron todos los objetivos y encontramos la lógica básica para poder comunicarnos a un dispositivo de visualización (monitor) de forma sencilla y manteniendo una resolución aceptable. Además de mostrar el “color depth” que maneja nuestra FPGA.