Práctica 10.

DISEÑO DE UN GENERADOR DE VIDEO VGA

OBJETIVO:

El alumno aprenderá como es la señalización para generar video VGA con el fin de comprender la funcionalidad del código que controla el mencionado tipo de señal de video.

ESPECIFICACIONES:

Utilizando un FPGA, un cable y pantalla VGA, se programará el controlador de video VGA, con la finalidad de proyectar una imagen estática.

Como se observa en el diagrama de bloques de la figura 10.1, el sistema tiene una entrada de reloj, y cinco salidas HS, VS, R, G y B.

DIAGRAMA DE BLOQUES:

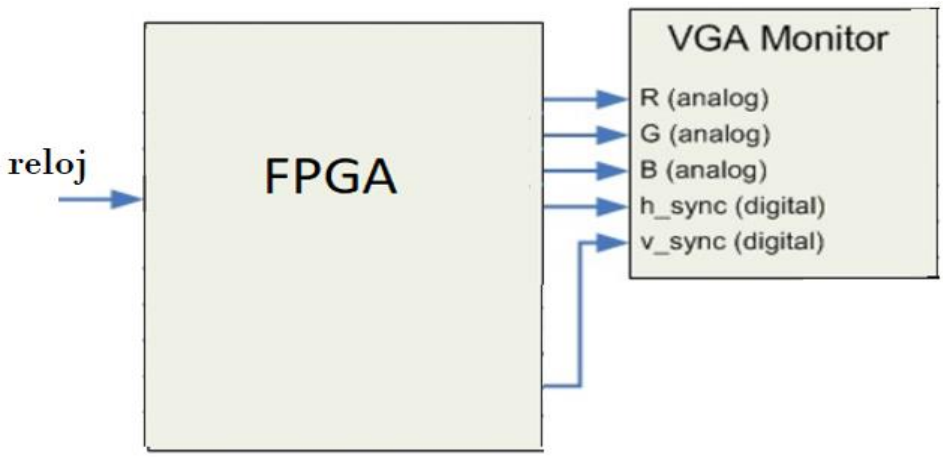

Figura 10.1. Diagrama de bloques del sistema Adaptador de Video VGA

Como se observa en el diagrama de bloques funcionales de la figura 10.2, el sistema cuenta con tres bloques funcionales.

DIAGRAMA DE BLOQUES FUNCIONALES:

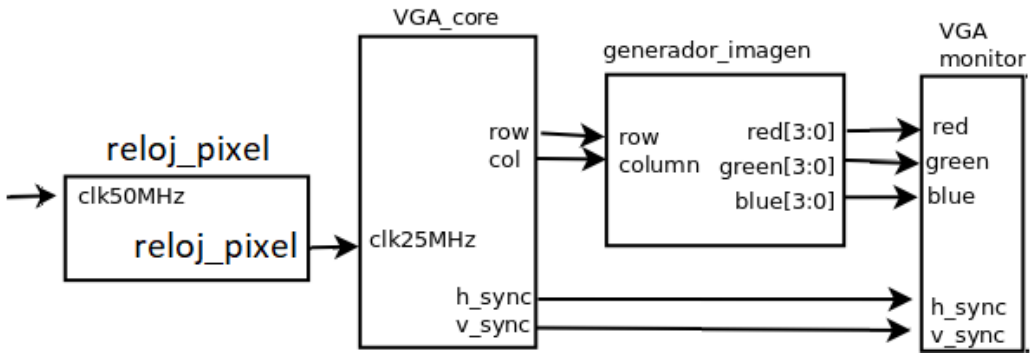

Figura 10.2. Diagrama de bloques funcionales del sistema Adaptador de Video VGA

INTRODUCCIÓN.

El VGA es un estándar de gráficos hecho por IBM en la década de los 80s. VGA significa "Video Graphics Array" y empezó como un adaptador gráfico de video, con una resolución de 640x480.

Una señal de vídeo VGA contiene 5 señales activas:

- Dos para sincronizar video: Sincronización horizontal y Sincronización vertical.
- Tres para asignar color: Rojo (R), Verde (G), Azul (B).

El formato VGA permite que se vean imágenes y video en un monitor, el video desplegará la emulación de movimiento con imágenes mostradas a una determinada velocidad. Las imágenes deben estar contenidas en un cuadro visible de 640x480 pixeles, que a su vez debe estar dentro de otro cuadro más grande (un margen invisible de derecha a izquierda y de arriba hacia abajo) de 800 pixeles x 525 líneas.

En la figura 10.3 se muestra gráficamente cómo se compone un cuadro de imagen VGA.

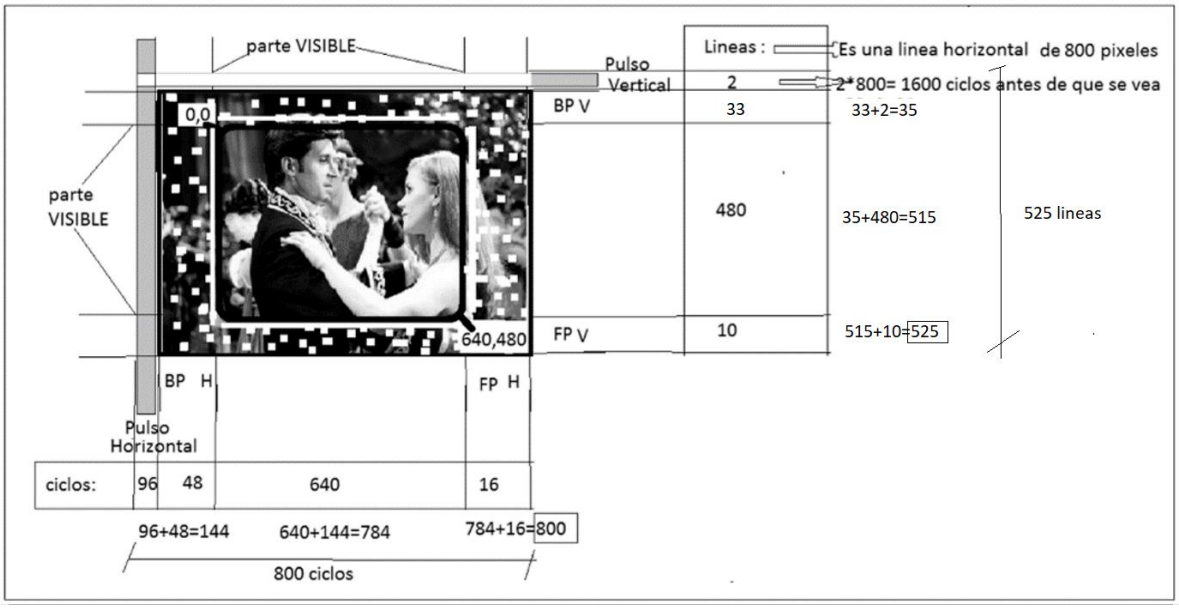

Figura 10.3. Ciclos y líneas de formato VGA 640x480

Los datos que se requieren para programar el controlador son:

- Frecuencia de actualización; 60hz. Resolución: 640x480 pixeles. Reloj: 25MHz.
- Parámetros Horizontales=> PulsoH=96, BPH=48, PH=640, FPH=16 (pixeles)
- Parámetros Verticales=> PulsoV=2, BPV=33, PV=480, FPV=10 (líneas)

DESARROLLO.

La figura 10.4 muestra la entidad del sistema de señalización VGA.

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

entity vga is
   port ( clk50MHz:  in std_logic;
                red: out std_logic_vector (3 downto 0);
              green: out std_logic_vector (3 downto 0);
               blue: out std_logic_vector (3 downto 0);
             h_sync: out std_logic;
             v_sync: out std_logic );
end entity vga;
```

Figura 10.4. Entidad del sistema de Adaptador de Video VGA

Se requiere generar una frecuencia de 25 MHz, la cual se obtendrá a partir del reloj principal de la tarjeta de desarrollo. En la figura 10.5 se observa el código de un divisor para obtener la frecuencia anteriormente mencionada a partir de un reloj de 50 MHz.

```
reloj_pixel: process (clk50MHz) is
begin
   if rising_edge(clk50MHz) then
      reloj_pixel <= not reloj_pixel;
   end if;
end process reloj_pixel;  -- 25mhz
```

Figura 10.5. Divisor de frecuencia del sistema Adaptador de Video VGA

Para controlar los tiempos horizontales y las líneas verticales, se requiere de dos contadores, uno horizontal y el otro vertical. El primero va de 0 a 800 y el segundo de 0 a 525. La figura 10.6 muestra el código correspondiente a los procesos requeridos.

```
contadores : process (reloj_pixel)  -- H_periodo=800, V_periodo=525
begin
   if rising_edge(reloj_pixel) then
      if h_count<(h_period-1) then
         h_count<=h_count+1;
      else
         h_count<=0;
         if v_count<(v_period-1) then
            v_count<=v_count+1;
         else
            v_count<=0;
         end if;
      end if;
   end if;
end process contadores;
```

Figura 10.6. Procesos de contadores del sistema Adaptador de Video VGA

```
senial_hsync : process (reloj_pixel)  --h_pixel+h_fp+h_pulse= 784
begin
   if rising_edge(reloj_pixel) then
      if h_count>(h_pixels + h_fp) or
         h_count>(h_pixels + h_fp + h_pulse) then
         h_sync<='0';
      else
         h_sync<='1';
      end if;
   end if;
end process senial_hsync;

senial_vsync : process (reloj_pixel)  --vpixels+v_fp+v_pulse=525
begin                    --checar si se en parte visible es 1 o 0
   if rising_edge(reloj_pixel) then
      if v_count>(v_pixels + v_fp) or
         v_count>(v_pixels + v_fp + v_pulse) then
         v_sync<='0';
      else
         v_sync<='1';
      end if;
   end if;
end process senial_vsync;

coords_pixel: process(reloj_pixel)
begin                    --asignar una coordenada en parte visible
   if rising_edge(reloj_pixel) then
      if (h_count < h_pixels) then
         column <= h_count;
      end if;
      if (v_count < v_pixels) then
         row <= v_count;
      end if;
   end if;
end process coords_pixel;
```

Figura 10.6. (continuación) Procesos de contadores del sistema Adaptador de Video VGA

Para visualizar el cuadro de imagen en el monitor VGA, solo queda programar en que renglón y columna inicial y final se empieza y se termina de pintar un color.

La figura 10.7, muestra las dos condiciones que se requieren para el despliegue de la imagen:

1.- Que el habilitador de pintura esté en el espacio visible.

2.- Colocar en la coordenada, el color asignado.

Si la cuenta horizontal (h_count) es menor que 640 y si al mismo tiempo el contador horizontal (v_count) es menor que 480, significa que estamos en el espacio visual y activamos la bandera de habilitación de despliegue (display_ena=1).

```
generador_imagen: PROCESS(display_ena, row, column)
BEGIN
   IF(display_ena = '1') THEN
      if ((row > 300 and row <350) and
         (column>350 and column<400)) THEN
         red <= (OTHERS => '1');
         green<=(OTHERS => '0');
         blue<=(OTHERS => '0');
      elsif ((row > 300 and row <350) and
            (column>450 and column<500)) THEN
         red <= (OTHERS => '0');
         green<=(OTHERS => '1');
         blue<=(OTHERS => '0');
      elsif ((row > 300 and row <350) and
            (column>550 and column<600)) THEN
         red <= (OTHERS => '0');
         green<=(OTHERS => '0');
         blue<=(OTHERS => '1');
      else
         red <= (OTHERS => '0');
         green  <= (OTHERS => '0');
         blue <= (OTHERS => '0');
      end if;
   ELSE
      red<= (OTHERS => '0');
      green <= (OTHERS => '0');
      blue<= (OTHERS => '0');
   END IF;
END PROCESS generador_imagen;
```

Figura 10.7. Proceso de generación de imagen del sistema Adaptador de Video VGA

Para pintar se tiene 3 colores rojo, verde y azul (en inglés Red, Green, Blue). Cuando: RGB= 1,0,0 el color es Rojo, si RGB= 0,1,0 el color que desplegará será verde, si RGB=0,0,1 será Azul, si RGB=1,1,1 el color es blanco y si RGB es 0,0,0 el color será negro.

La figura 10.8 muestra el proceso habilitador de visualización del sistema Adaptador de Video VGA.

```
display_enable: process(reloj_pixel) --- h_pixels=640; y_pixeles=480
begin
   if rising_edge(reloj_pixel) then
      if (h_count < h_pixels AND v_count < v_pixels) THEN
         display_ena <= '1';
      else
         display_ena <= '0';
      end if;
   end if;
end process display_enable;
```

Figura 10.8. Proceso habilitador de visualización del sistema Adaptador de Video VGA

Se requiere dar valores a las constantes para el manejo del formato VGA, que se declaran dentro de un "GENERIC", la figura 10.8, muestra un ejemplo de estos valores.

```
GENERIC(     --Constantes para monitor VGA en 640x480
   CONSTANT h_pulse  : INTEGER := 96;
   CONSTANT h_bp     : INTEGER := 48;
   CONSTANT h_pixels : INTEGER := 640;
   CONSTANT h_fp     : INTEGER := 16;
   CONSTANT v_pulse  : INTEGER := 2;
   CONSTANT v_bp     : INTEGER := 33;
   CONSTANT v_pixels : INTEGER := 480;
   CONSTANT v_fp     : INTEGER := 10
   );
```

Figura 10.9. Constantes dentro de un *GENERIC* del sistema Adaptador de Video VGA

Las declaraciones y operaciones de las constantes tipo señal adicionales, se muestra en figura 10.10.

```
--Contadores
signal h_period : INTEGER := h_pulse + h_bp + h_pixels + h_fp;
signal v_period : INTEGER := v_pulse + v_bp + v_pixels + v_fp;
signal h_count  : INTEGER RANGE 0 TO h_period - 1 := 0;
signal v_count  : INTEGER RANGE 0 TO v_period - 1 := 0;
```

Figura 10.10. Declaraciones y operaciones de las constantes tipo señal

ACTIVIDAD COMPLEMENTARIA:

El alumno unirá los distintos procesos en uno solo y mostrará sus resultados en un monitor.