UNIVERSIDAD NACIONAL DE SAN MARTÍN

ESCUELA DE CIENCIA Y TECNOLOGÍA

# Informe Proyecto Final Electrónica Digital I:

# “Desarrollo de voltímetro digital sobre placa FPGA”

Fecha: 07 de *marzo de 2019*

**Autores:** Joaquin Gonzalez joagonzalez@gmail.com
David, Wolovelsky dwolovelsky@gmail.com

**Carrera:** Ing. en Telecomunicaciones

# Contenido

# Objetivo

El objetivo de este trabajo es implementar, en lenguaje VHDL, un voltímetro digital conformado por un conversor analógico-digital Sigma-Delta a la entrada para realizar el muestreo y la conversión de la señal a medir. A la salida del sistema, la medición realizada será enviada al módulo VGA de la placa Spartan-3E.

Los tres módulos principales del voltímetro se identifican de la siguiente manera:

- Bloque que se encarga de digitalizar el voltaje de entrada que se desea medir
- Bloque controlador VGA que se encarga de producir la señal de sincronismo e indicar en qué posición (horizontal y vertical) de la pantalla se encuentra en ese momento.
- Bloque que se encarga de encender los píxeles correspondientes a los caracteres que se desea proyectar en la pantalla

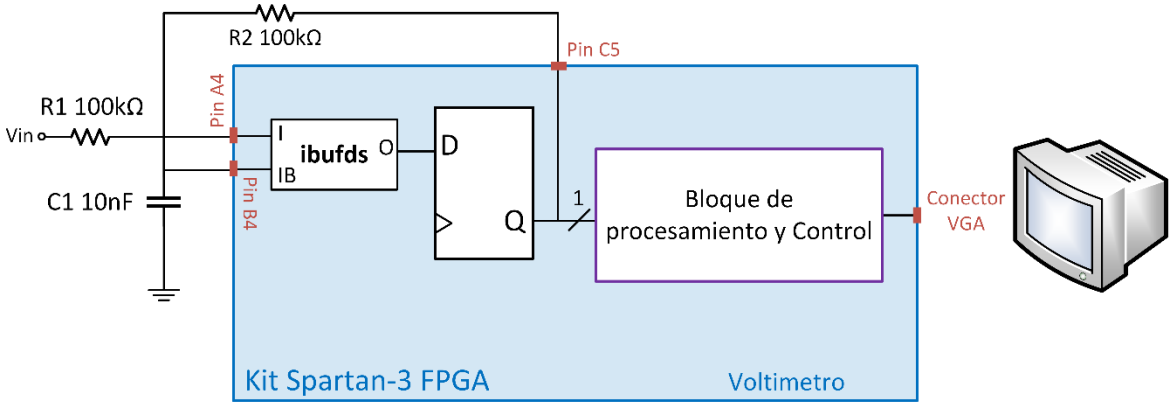



**Figura 1. Conexión de placa Spartan-3E con voltímetro y periféricos**

## Conversor ADC Sigma-Delta

La señal de voltaje que se desea medir es de naturaleza analógica, por lo tanto, debe digitalizarse de manera que pueda ser procesada por el voltímetro desarrollado. Esta tarea la realizaremos con un conversor ADC Sigma-Delta.

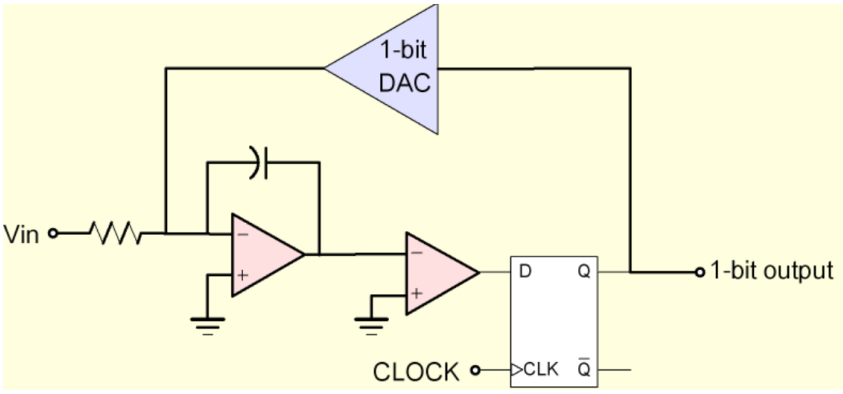



Figura 2. Conversor ADC Sigma-Delta

1. Al voltaje de entrada Vin se le resta el voltaje a la salida del DAC de 1 bit a través del loop de realimentación negativa

2. Esta diferencia de tensión se integra y la salida se compara con una tensión de referencia Vref. El integrador se implementa de manera externa a la placa como se muestra en la figura 1

3. La salida del comparador se inyecta a la entrada de un Flip-flop D

4. La salida Q del Flip-Flop D se realimenta a través del DAC de 1 bit a la entrada del sistema

5. La salida será Q=1 (Vref) o Q=0 (0V) dependiendo el valor de D

6. El contador BCD contará cada 1 que salga del ADC (Q=1)

7. El proceso se repite durante N ciclos del clock (33000 limitados por el bloque contador que resetea el módulo BCD)

8. El voltaje medido por el voltímetro será (#Cantidad de 1's a la salida de ADC/N) x Vref donde N=33000 y Vref=3.3V, dando un valor de 0.0001V por cada uno ingresado al BCD

Debido a lo observado en el punto 8, se utilizará un contador BCD de 5 décadas para poder almacenar una resolución de 5 valores. Los últimos 3 dígitos que utiliza el contador (Q4, Q3, Q2) serán los dígitos ***D1.D2D3*** que se desean medir y mostrar en pantalla.

Se utiliza un reloj de sincronismo interno que provee el kit de desarrollo Spartan-3E que trabaja a una frecuencia de 50MHz (t=1/50MHz=20ns). En donde t*33000=660us. Se contempla que la señal a medir no tendrá cambios bruscos o componentes de muy alta frecuencia, por lo que el tiempo de medición es aceptable bajo estas consideraciones.

## Bloque controlador VGA

El bloque controlador tiene las siguientes características de funcionamiento:

- Imagen de 640 pixeles * 480 líneas x 60 Hz (800 píxeles * 525 líneas en total)
- Sincronismo Horizontal
  - 96 píxeles de sincronismo (hs)
  - 48 píxeles de porch trasero (hbp)
  - 16 píxeles de porch delantero (hfp)
- Sincronismo Vertical
  - 2 píxeles de sincronismo (vs)
  - 33 píxeles de porch trasero (vbp)
  - 10 píxeles de porch delantero (vfp)

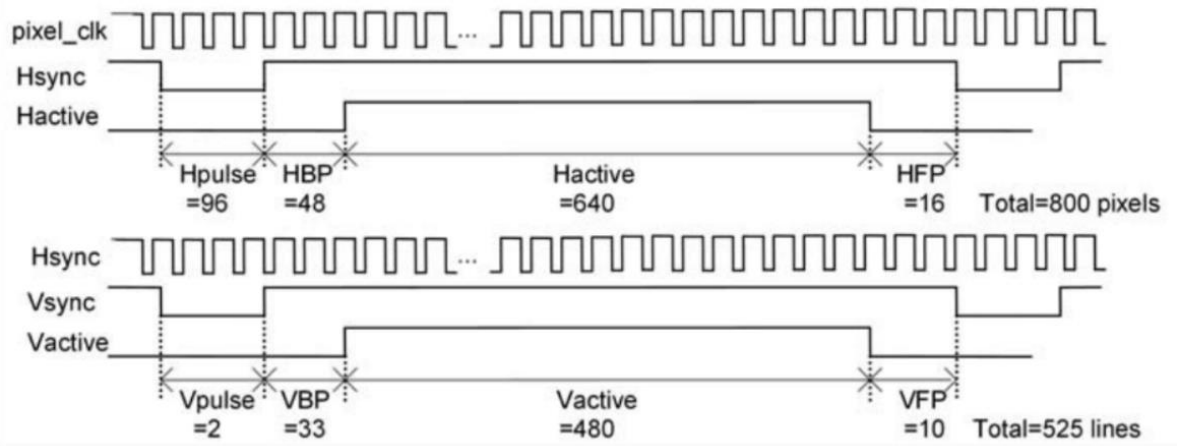



Figura 3. Sincronismo señal VGA

El controlador VGA está conformado por dos contadores, uno para el barrido horizontal (v_cont_h) que contará hasta 800, al llegar a este número enviará una señal de Enable al contador vertical (v_cont_v) el cual bajará una línea en el barrido. El contador horizontal tiene una condición de reset al contar 801 y el vertical al contar 522, esto permitirá que las coordenadas (x,y) que se envíen por pantalla tomen los valores requeridos para la resolución elegida.

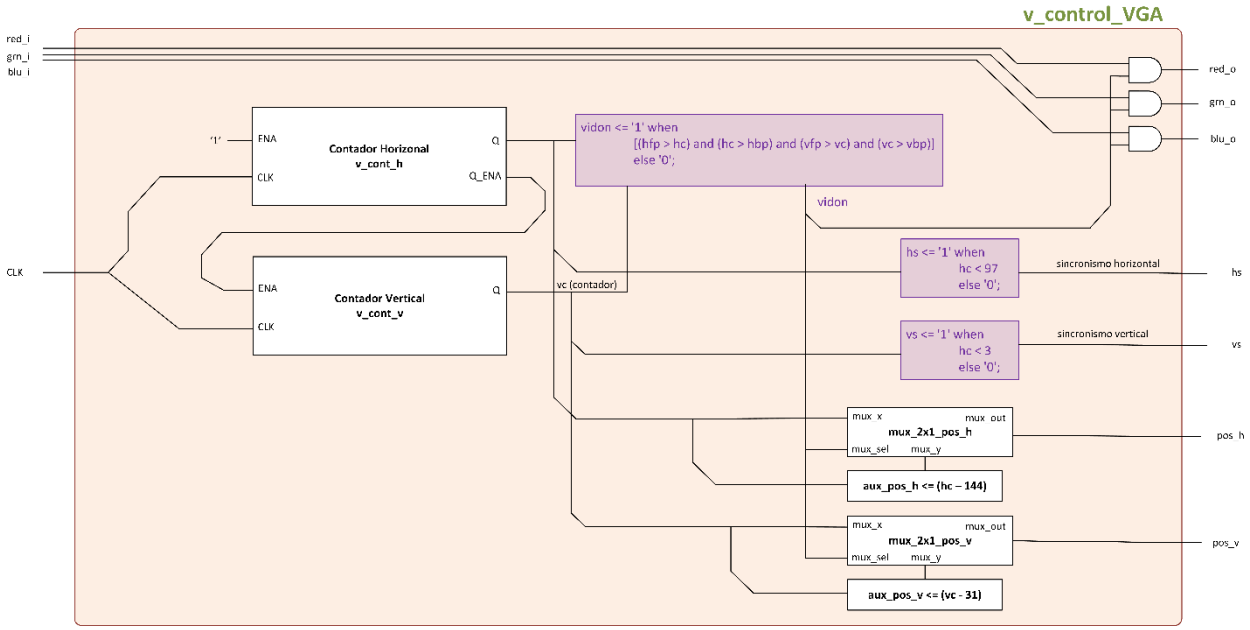



Figura 4. Controlador VGA

El controlador VGA requiere una frecuencia de reloj de 25.175MHz, prácticamente la mitad de la frecuencia utilizada para sincronizar el resto de los componentes del voltímetro. Se implementa un divisor de frecuencia por dos para alimentar el reloj del controlador VGA como se muestra en la figura 4.

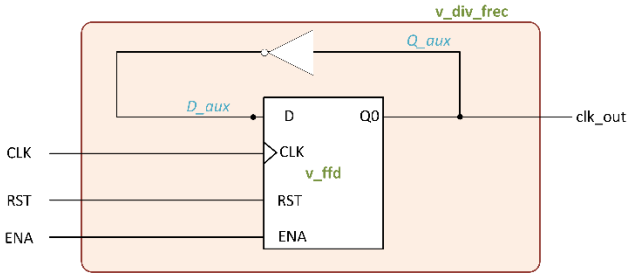



Figura 5. Bloque divisor de frecuencia por 2 utilizando flip-flop D

Además, se utilizan comparadores a la salida de los contadores horizontal y vertical con el objetivo de generar un flag llamado vidon. Este será '1' cuando el contador se encuentre en la zona visible, es decir, cuando sea mayor que los back porch y cuando sea menor que los front proch. Este flag, se utilizará para habilitar las salidas RGB y, también, será utilizado de selector en un MUX 2x1 que sacará las señales pos_h y pos_v que alimentarán el bloque v_MUX y v_CGA.

## Generador de píxel y ROM

Con el objetivo de poder graficar los dígitos que mide el contador BCD en pantalla, se realizará una distribución matricial de la zona visible para poder ubicarlos en bloques específicos de la misma. Para ello se divide la pantalla (zona visible) en 5 columnas de 128 píxeles de ancho (5*128=640), y 3 filas de 128 píxeles de alto más una de 96 (128*3+96=480). Además, cada uno de estos bloques de 128*128 píxeles se dividen en 8*8 sub-bloques de 16 píxeles de lado. Estos sub-bloques de 16*16 bloques (256 píxeles en total) serán tratados como una unidad básica de encendido/apagado por los controladores RGB.

Para controlar la posición en pantalla se utilizará la señal de barrido en combinación con multiplexores que permitirán establecer la condición de posición para identificar el bloque que se está recorriendo en un momento dado y, de esta manera, poder decidir si se desea enviar señal de encendido o no. Además, al tomar la decisión de encender/apagar en base a los bloques donde queremos dibujar los dígitos medidos, utilizaremos una memoria ROM donde almacenaremos las “formas” que deseamos dibujar en pantalla. Esta memoria contendrá los píxeles que deben encenderse dentro de un bloque de 128*128 para dibujar un carácter.

La memoria ROM de caracteres y el MUX que verifica la condición de encendido/apagado vertical se implementan en el módulo v_CGA como se observa en la figura 5.

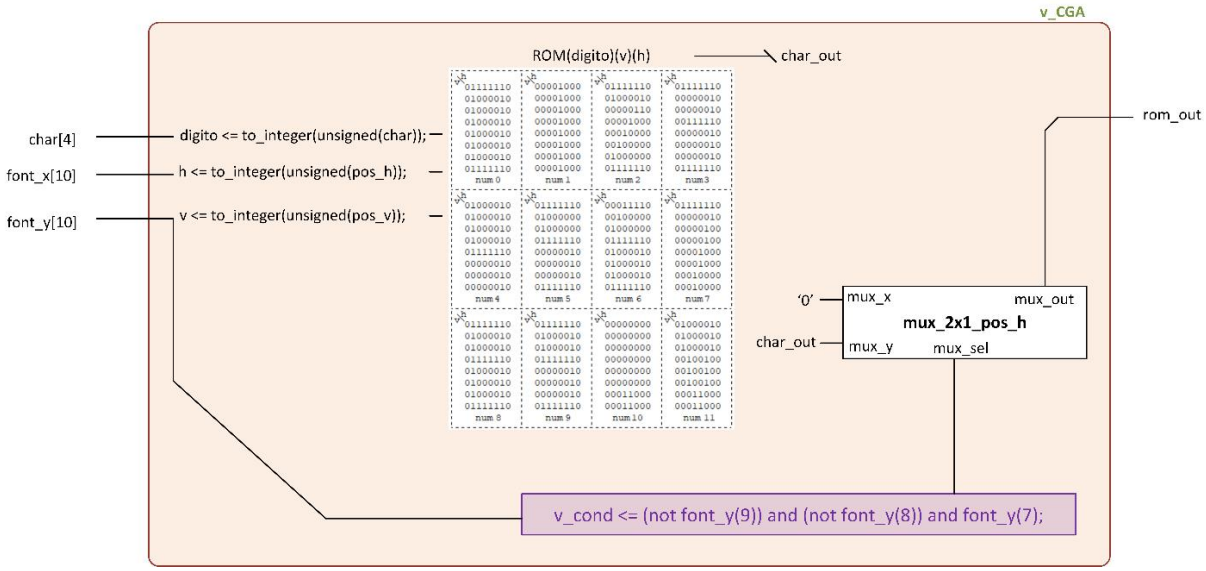



Figura 6. Bloque v_CGA

La salida rom_out del bloque v_CGA está conectada a las entradas RGB del controlador VGA, que son los que envían las señales de encendido/apagado. Las condiciones de encendido/apagado horizontal serán controladas con el bloque v_MUX, que es un multiplexor de 5 entradas (D1,’.’,D2,D3,’V’) y que, dependiendo la columna donde se encuentre el contador de posición x, será el dígito que le indique dibujar al módulo v_CGA.

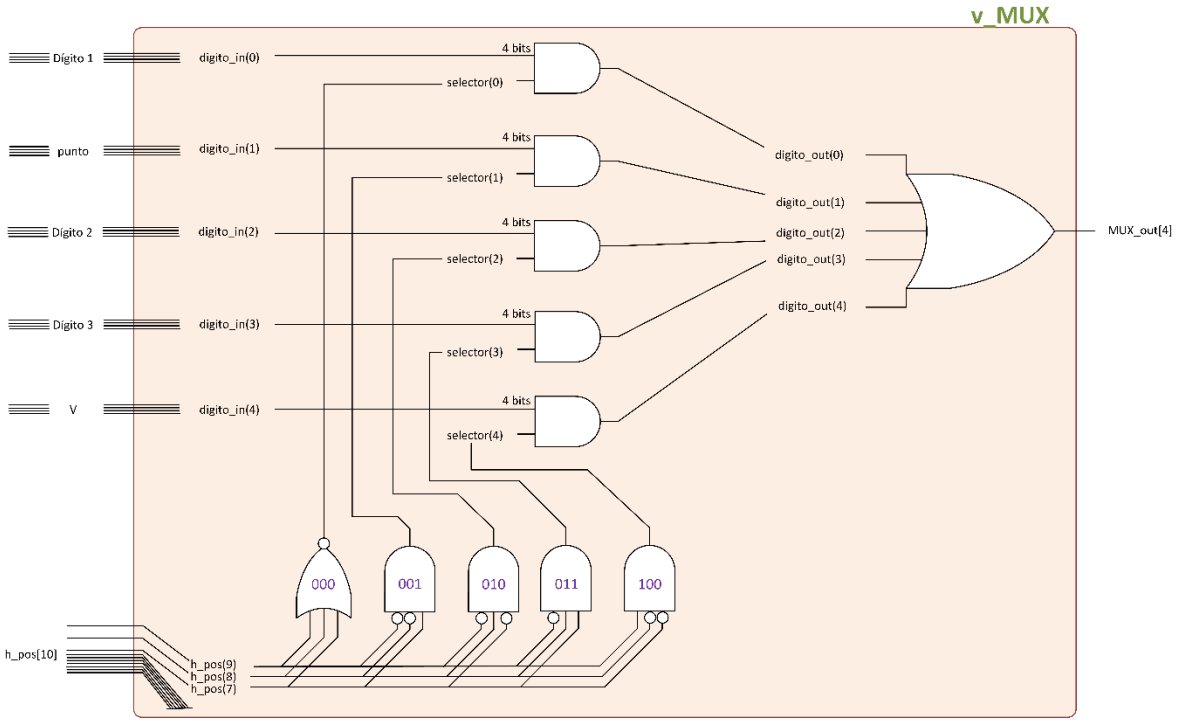



Figura 7. Bloque v_MUX. Selección de caracter a dibujar en base a sync horizontal

Al bloque v_MUX se le agrego una modificación extra, para validar también la posición vertical y, en esos casos, enviar todos ceros en los 4 bits de la salida MUX_out.

```
-- 1     =     000    Franja de pantalla 1/5 y fijando franja vertical 001
selector(0) <= (not (h_pos(9) or h_pos(8) or h_pos(7))) and ((not v_pos(9)) and (not v_pos(8)) and
v_pos(7));
-- 1     =     001    Franja de pantalla 2/5 y fijando franja vertical 001
selector(1) <= ((not h_pos(9)) and (not h_pos(8)) and h_pos(7)) and ((not v_pos(9)) and (not v_pos(8)) and
v_pos(7));

-- 1     =     010    Franja de pantalla 3/5 y fijando franja vertical 001
selector(2) <= ((not h_pos(9)) and h_pos(8) and (not h_pos(7))) and ((not v_pos(9)) and (not v_pos(8)) and
v_pos(7));

-- 1     =     011    Franja de pantalla 4/5 y fijando franja vertical 001
selector(3) <= ((not h_pos(9)) and h_pos(8) and h_pos(7)) and ((not v_pos(9)) and (not v_pos(8)) and
v_pos(7));

-- 1     =     100      Franja de pantalla 5/5 y fijando franja vertical 001
selector(4) <= (h_pos(9) and (not h_pos(8)) and (not h_pos(7))) and ((not v_pos(9)) and (not v_pos(8)) and
v_pos(7));

   digito_in(0) <= D1;
   digito_in(1) <= punto;
   digito_in(2) <= D2;
   digito_in(3) <= D3;
   digito_in(4) <= V;

   digito_out_block : for i in 0 to 4 generate
      digito_out(i) <= (digito_in(i)(3)and selector(i))&(digito_in(i)(2)and selector(i))&(dig-
ito_in(i)(1)and selector(i))&(digito_in(i)(0)and selector(i));
   end generate digito_out_block;

   MUX_out <= digito_out(4) or digito_out(3) or digito_out(2) or digito_out(1) or digito_out(0);
```

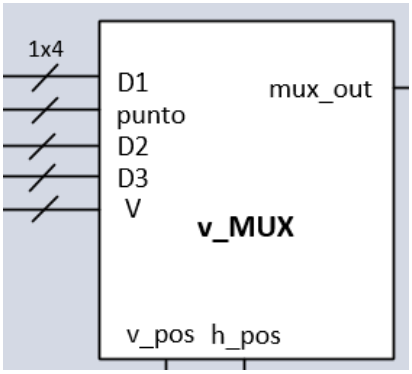



Figura 8. v_MUX con verificación de posición vertical

## Resumen y funcionamiento voltímetro

El diseño del voltímetro, junto con todos sus bloques, se describe en la figura 8.

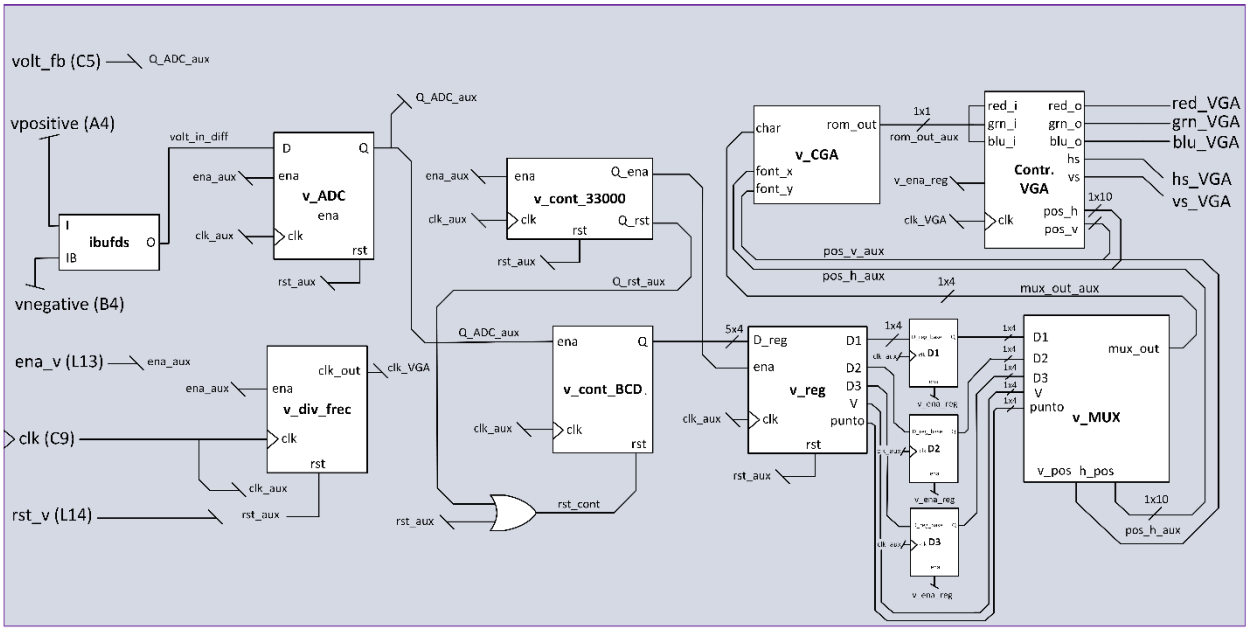



Figura 9. voltímetro digital

La tensión a medir ingresará al bloque ADC a través de vpositive. La salida del ADC (Q) ingresará a la entrada del contador BCD de 5 décadas. La resolución del contador será de 0.0001V como se indicó en la introducción de este trabajo. El bloque v_cont_33000 contará de forma independiente 33001 ciclos de reloj y, al llegar al final de cada ciclo de conteo enviará una señal de reset al bloque contador BCD. Además, al contar 33000 enviará una señal de enable al bloque v_reg que capturará la salida de las tres cifras más significativas del bloque v_cont_BCD que se desactivará al siguiente ciclo de reloj. Como v_reg actúa de memoria, guardará estos valores y los enviará, junto con las referencias en ROM de los caracteres “V” y “.” Al bloque v_MUX. Este multiplexor de 5 entradas enviará, a través de su salida mux_out, el dígito que debe dibujar el controlador VGA en base a la posición en pantalla que indiquen las señales de barrido v_pos y h_pos. Los primeros 3 bits de estas señales (9, 8, 7) indican en cuál de los

5 bloques horizontales se encuentra el barrido y en cual de los 4 verticales como se puede ver en la figura 9. Los bits 6, 5 y 4 indican en cual de los 64 sub bloques (8*8) de 16*16 píxeles se encuentra el barrido, de esta manera se podrá referenciar la ROM de manera específica como se muestra a continuación: ROM(digito)(v)(h), donde v y h son variables que contienen la codificación a entero de los 3 bits indicados.

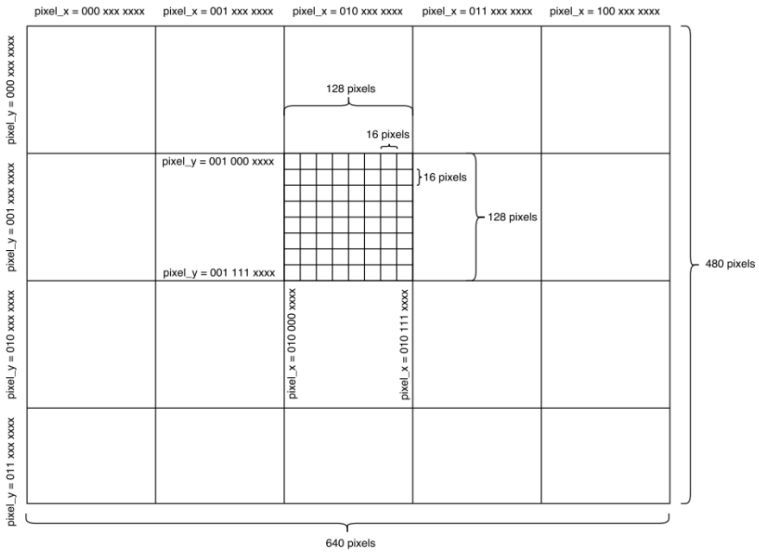



Figura 10. División matricial de pantalla y señales de barrido pixel_x y pixel_y

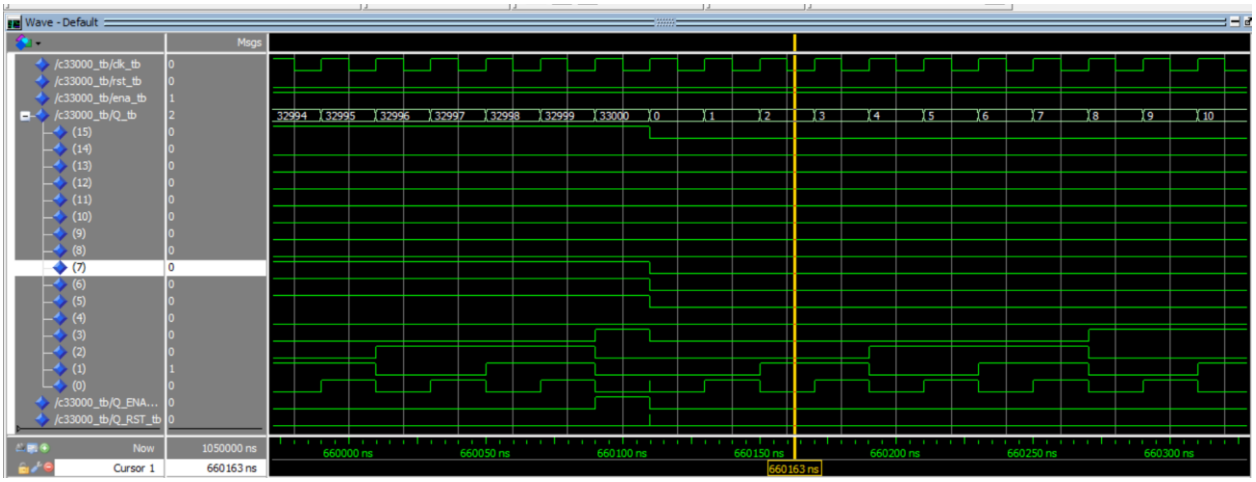
La salida de 4 bits del bloque v_MUX, que indica el dígito a dibujar en pantalla, será enviada al módulo v_CGA, este bloque buscará en la matriz ROM como debe dibujarse ese dígito en pantalla, es decir, que puntos (16*16 píxeles) debe encender/apagar dentro de la celda referenciada. Además, este módulo tiene una validación adicional, la cual enviará “0” a través de rom_out en caso de estar en la fila vertical no deseada. De no existir esta validación, se repetirían los dígitos en todas las filas de la pantalla.

```
    pos_h <= font_x(6)&font_x(5)&font_x(4); -- Determinacion del pixel horizontal
    pos_v <= font_y(6)&font_y(5)&font_y(4); -- Determinacion del pixel vertical

-- Determinacion del subondice para el caracter seleccionado
    digito <= to_integer(unsigned(char));

-- Determinacion del subindice para el pixel horizontal
    h <= to_integer(unsigned(pos_h));
-- Determinacion del subindice para el pixel vertical
    v <= to_integer(unsigned(pos_v));

    -- Condicion para habilitar salida (001)
    v_cond <= (not font_y(9)) and (not font_y(8)) and font_y(7);
    -- Caracter seleccionado
    char_out <= ROM(digito)(v)(h);

    mux_selector: v_mux_2x1
        port map(
            mux_x => '0',
            mux_y => char_out,
            mux_sel => v_cond,
            mux_out => rom_out
        );
```

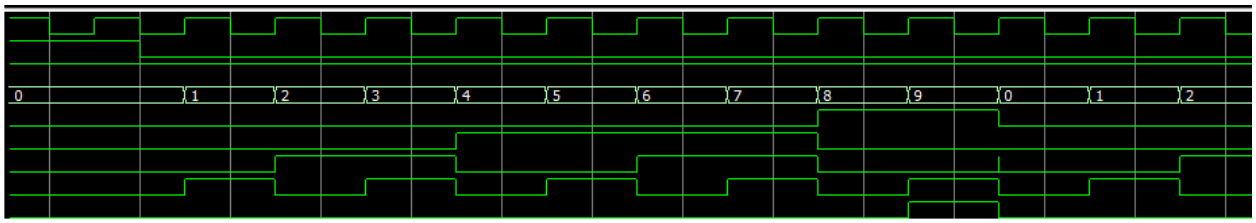
La salida del bloque v_CGA está conectada a las entradas red_i, grn_i y blu_i del bloque v_control_VGA. Dependiendo de la combinación de colores deseada para mostrar en pantalla, serán las señales que se envíen a cada color. Estas señales indicarán cuando deberá encenderse un bloque de 16*16 píxeles de la pantalla y serán reenviadas a través de las salidas red_o, grn_o y blu_o respectivamente. El bloque controlador VGA se encargará de enviar las señales de sincronismo al módulo de la FPGA asi como identificar las posiciones de la zona activa (posiciones visibles dentro de la pantalla) que habilitaran que pasen señales por las salidas red_o, grn_o y blu_o. Logra estas funciones a través de contadores de posición horizontal y vertical. Las salidas de los contadores (10 bits) serán realimentadas como entradas de v_MUX y v_CGA como fue explicado anteriormente. Es importante destacar que el reloj del controlador VGA tiene una frecuencia de 25MHz, la cual se logra a través de un bloque divisor de frecuencia.

## Testing de módulos

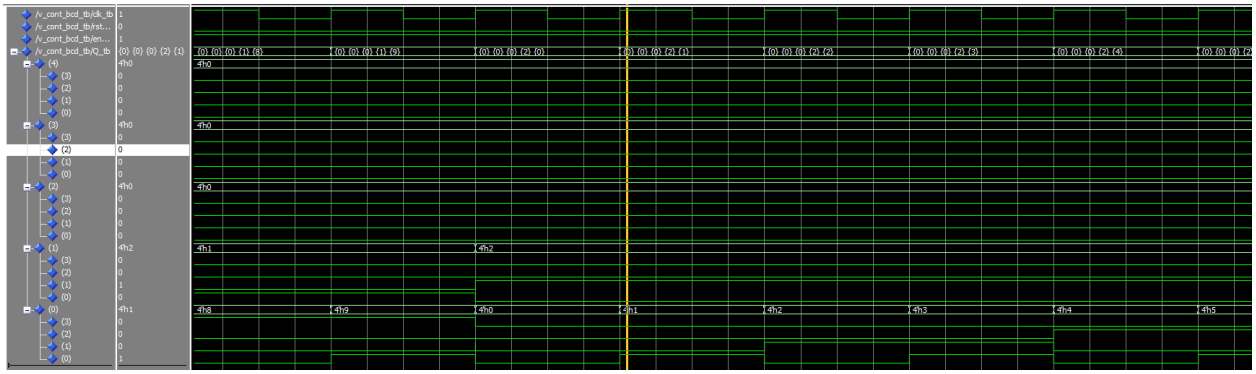
**v_cont_33000_tb.vhd**



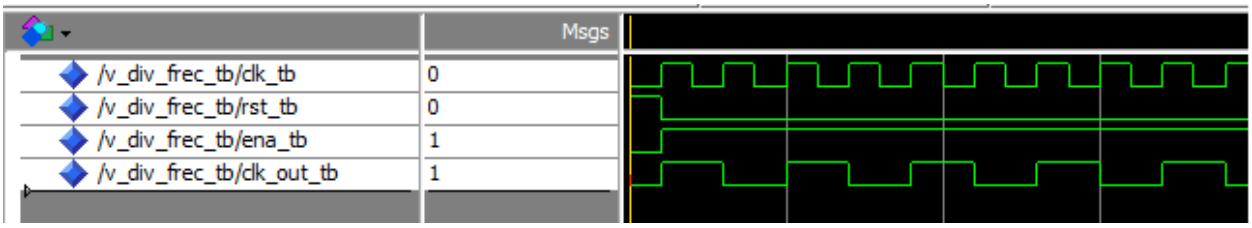

**v_cont_BCD_base_tb.vhd**



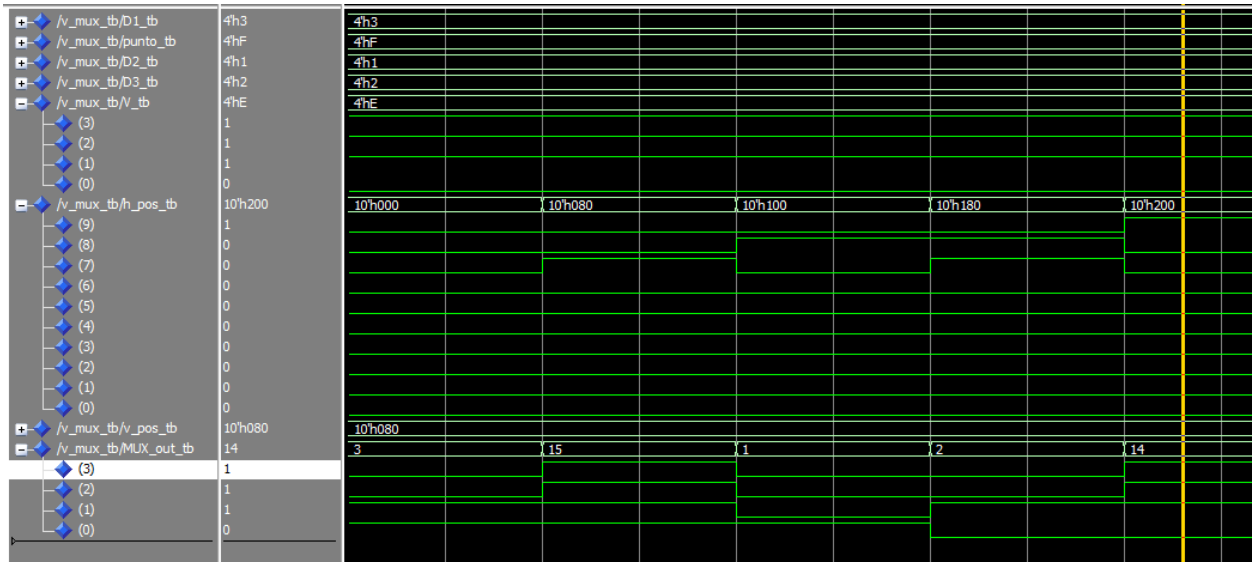
**v_cont_BCD_tb.vhd**



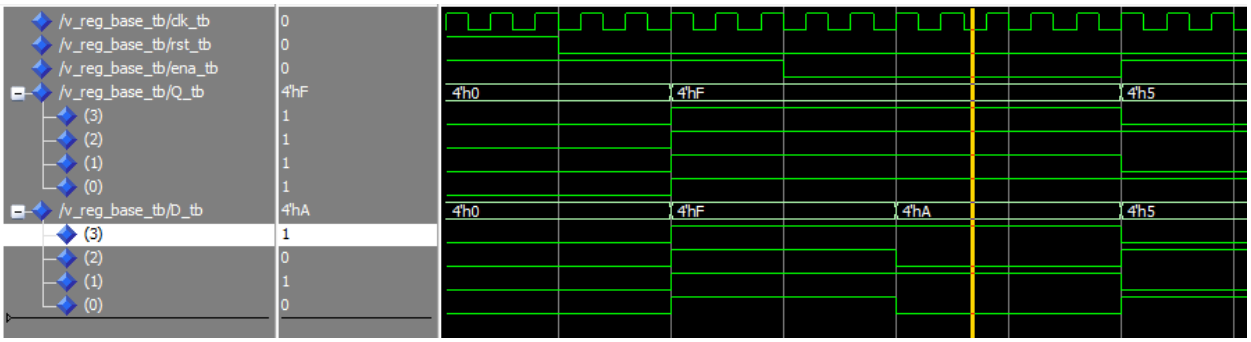
**v_div_frec_tb.vhd**



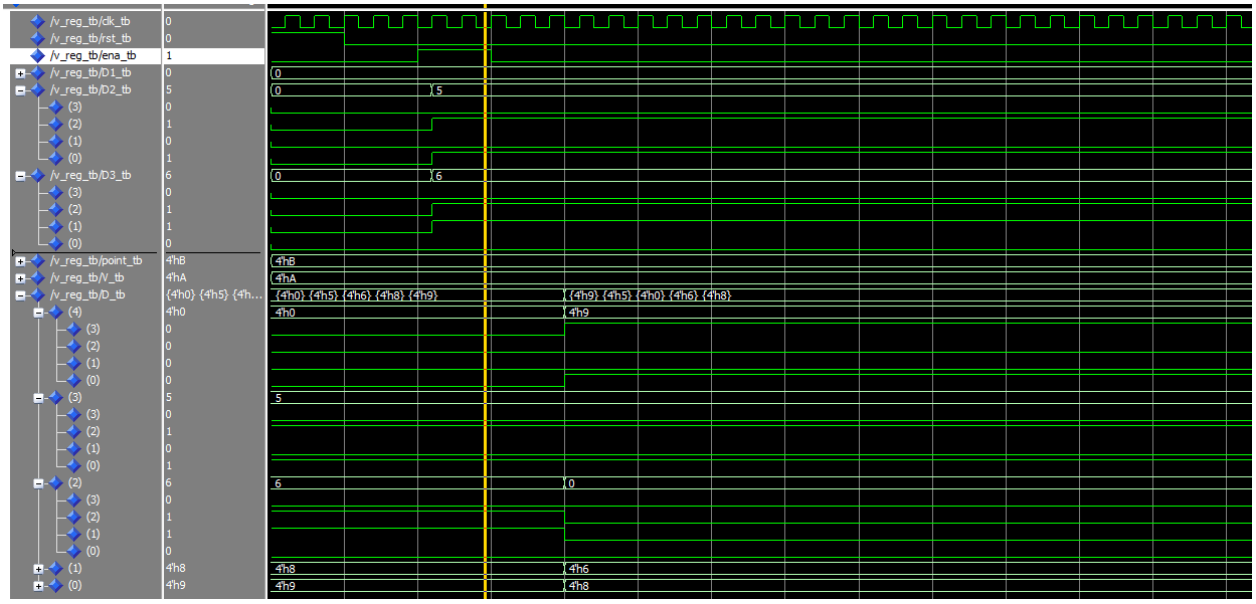

**v_MUX_tb.vhd**




**v_reg_base_tb.vhd**


/v_reg_base_tb/clk_tb
/v_reg_base_tb/rst_tb
/v_reg_base_tb/ena_tb
/v_reg_base_tb/Q_tb
(3)
(2)
(1)
(0)
/v_reg_base_tb/D_tb
(3)
(2)
(1)
(0)
4'h0
4'hF
4'h5
4'hA

**v_reg_tb.vhd**


/v_reg_tb/clk_tb
/v_reg_tb/rst_tb
/v_reg_tb/ena_tb
/v_reg_tb/D1_tb
/v_reg_tb/D2_tb
/v_reg_tb/D3_tb
/v_reg_tb/point_tb
/v_reg_tb/V_tb
/v_reg_tb/D_tb
4'h8
4'hA
4'h9
4'h5
4'h6