

# TRABAJO PRÁCTICO FINAL

## “Voltímetro digital con salida VGA”

*Electrónica Digital I*

ECyT - UNSAM

Rocío Belén Fernández ([rfernandez@estudiantes.unsam.edu.ar](mailto:rfernandez@estudiantes.unsam.edu.ar))

**Docentes :**

Álvarez, Nicolás

Sagreras, Miguel Ángel

**Período de cursada:**

1er Cuatrimestre 2023

# Índice

<b>1. Objetivo</b>	<b>4</b>
<b>2. Resumen</b>	<b>4</b>
<b>3. Desarrollo</b>	<b>4</b>
a. Consideraciones	4
b. Diseño	4
• Flip -Flop D (ffd.vhd)	5
• Registro de N bits (reg_Nb.vhd)	5
• Contador BCD (BCD_counter.vhd)	5
• Contador de unos (cOnes.vhd)	5
• Contador binario (c33K.vhd)	5
• Comparador de N bits (comp_Nb.vhd)	6
• Multiplexor de 4 bits (mux.vhd) (mux_color.vhd) (mux_selector.vhd)	6
• ROM de caracteres (carac_rom.vhd)	6
• Sincronismo horizontal y vertical (hsync.vhd) (vsync.vhd)	6
• VGA (vga_controller.vhd)	7
• Voltímetro (Voltímetro.vhd)	7
c. Simulación	8
d. Implementación	10
e. Primeras pruebas con FPGA	11
<b>Anexo</b>	<b>12</b>
Flip - Flop	12
Esquemático:	12
Simulación :	12
Registro	13
Esquemático:	13
Registro de 4 bits	14
Esquemático:	14
Simulación :	14
Comparador de 10 bits	15
Esquemático:	15
Simulación :	15
Contador BCD	16
Esquemático:	16
Simulación :	16
Contador de unos	16
Esquemático:	16
Simulación :	17
Contador 3 300 000	17
Esquemático:	17
Simulación :	17
Lógica MUX	18
Esquemático:	18

Simulación :	18
Color MUX	18
Esquemático:	18
Simulación :	19
MUX	19
Esquemático:	19
Simulación :	19
Controlador VGA	20
Esquemático:	20
Simulación :	20
ROM - Caracteres	20
Esquemático:	20
Simulación :	20
Sincronismo horizontal	21
Esquemático:	21
Simulación :	21
Zona visible	21
Sincronismo vertical	22
Esquemático:	22
Simulación :	22
Zona visible	22

# 1. Objetivo

El objetivo del presente trabajo es diseñar un voltímetro digital con salida VGA en un rango de tensión de 0 a 3,3V. Para llevarlo a cabo se debe diseñar, describir la arquitectura, simular, sintetizar e implementar en una FPGA.

# 2. Resumen

En este informe se detallan las funciones de cada bloque y cómo se organizan jerárquicamente. También se incluyen las mediciones y simulaciones realizadas, tanto en Vivado como en la práctica real. Además, se explican los criterios de diseño que se usaron antes y después de hacer las pruebas, y se muestran algunas tablas generadas o tomadas durante el proceso de síntesis e implementación.

# 3. Desarrollo

## a. Consideraciones

Para poder desarrollar el dispositivo, se tuvo que tener en cuenta las siguientes restricciones. No se pueden utilizar las sentencias: if-then-else, case-when, for-while-loop-exit-next, process. La única excepción fue para la descripción del flip-flop D (ffd).

## b. Diseño

Se parte del diseño dado por la cátedra, figura 1. Se observa el “Bloque de procesamiento de datos y control” que puede implementarse como se ve en la figura 2, dado por la cátedra.

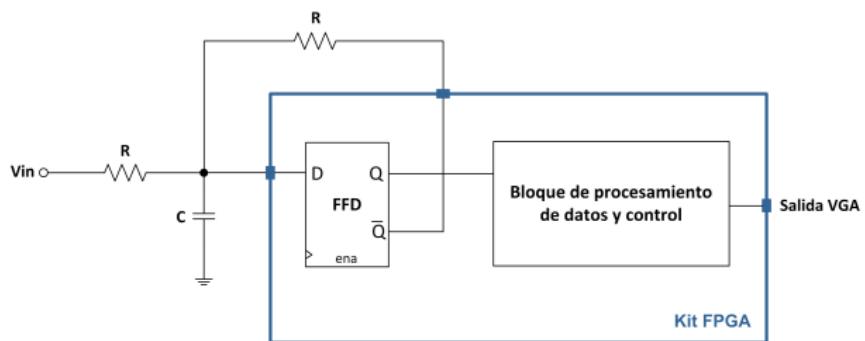


Figura 1: Diagrama simplificado del voltímetro entregado por la cátedra

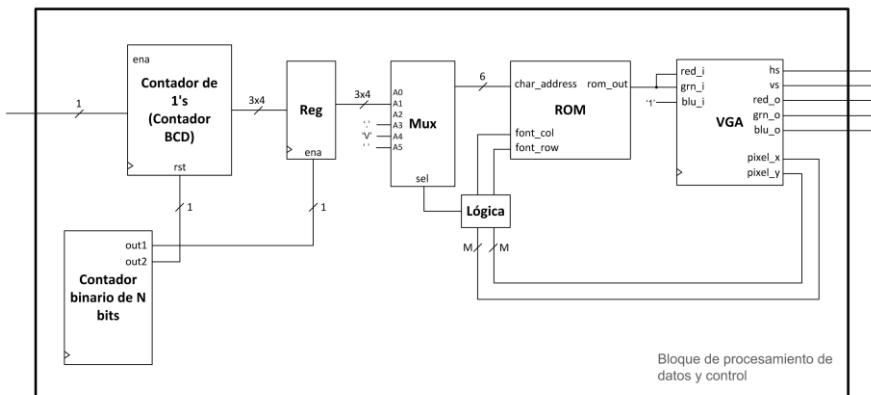


Figura 2: Diagrama del bloque de procesamiento entregado por la cátedra

Anterior al bloque de procesamiento de datos, hay un bloque de conversión analógico-digital. La función de dicho bloque es convertir la entrada analógica en una señal digital para poder ser procesada, en este trabajo se uso un Flip-Flop D (ffd.vhd).

A continuación detallaré cada componente desarrollado.

- **Flip -Flop D (ffd.vhd)**

Biestable implementado por descripción de comportamiento. Su salida complementaria se realimenta hacia la entrada, permitiendo cerrar el lazo de comparación necesario en el sistema.

Además, se va a utilizar como base de varios de los componentes detallados a continuación.

- **Registro de N bits (reg\_Nb.vhd)**

Implementación de un registro paralelo compuesto por N Flip-Flops tipo D. Su función es almacenar de forma simultánea N bits de información, conservando una palabra binaria estable durante el funcionamiento. Este bloque es esencial en contadores y sistemas secuenciales donde se requiere almacenamiento temporal.

- **Contador BCD (BCD\_counter.vhd)**

Este módulo implementa un contador BCD que genera una secuencia binaria del 0 al 9. Utiliza un registro de 4 bits, un sumador y un comparador para almacenar, incrementar y verificar el valor de la cuenta. La salida count indica el valor actual, mientras que max señala cuando se alcanza el máximo permitido.

- **Contador de unos (cOnes.vhd)**

Bloque que utiliza contadores BCD en cascada para contabilizar los pulsos provenientes del conversor analógico digital. La salida proporciona los 3 dígitos más significativos, que representan el valor de la tensión medida. Cada dígito se entrega en formato de 4 bits BCD.

- **Contador binario (c33K.vhd)**

Contador que realiza una secuencia de 0 hasta 3.300.000. Controla el momento en que se habilita el almacenamiento del valor de tensión y cuándo se debe resetear el contador de unos. Su implementación utiliza un registro de 22 bits, un sumador y dos comparadores estructurales.

- **Comparador de N bits (comp\_Nb.vhd)**  
Realiza la comparación entre dos valores de N bits. En su salida hay un 1 si la comparación es verdadera.
- **Multiplexor de 4 bits (mux.vhd) (mux\_color.vhd) (mux\_selector.vhd)**  
Selecciona entre múltiples entradas de 4 bits, permitiendo redirigir los datos almacenados en el registro de voltaje. Incluye entradas fijas que representan el punto decimal y el símbolo de unidad. La selección se controla mediante los 3 bits más significativos de la coordenada pixel X de la VGA.
- **ROM de caracteres (carac\_rom.vhd)**  
Esta ROM tiene almacenado mapeos de binarios para dibujar los caracteres en el monitor (0 al 9, punto y unidad). La dirección de acceso se forma combinando la salida del multiplexor y los bits relevantes de las coordenadas píxel X e pixel Y de la VGA. La salida de la ROM determina qué píxeles deben activarse para formar los caracteres.
- **Sincronismo horizontal y vertical (hsync.vhd) (vsync.vhd)**  
Generan las posiciones horizontales y verticales necesarias para la sincronización VGA. Incluyen zonas de back porch, front porch, área visible y los pulsos de sincronismo. El contador horizontal cubre 800 ciclos (píxeles) y el vertical 525 líneas. Se propuso trabajar con un contador binario genérico, cuya salida es evaluada mediante distintos comparadores. Según el valor actual de la cuenta, las salidas de estos comparadores modifican las señales de control conectadas a un flip-flop tipo D, generando como resultado un uno o un cero lógico, dependiendo del estado del conteo. Este esquema se representa en la figura 3 para el sincronismo horizontal y en la figura 4 para el sincronismo vertical.

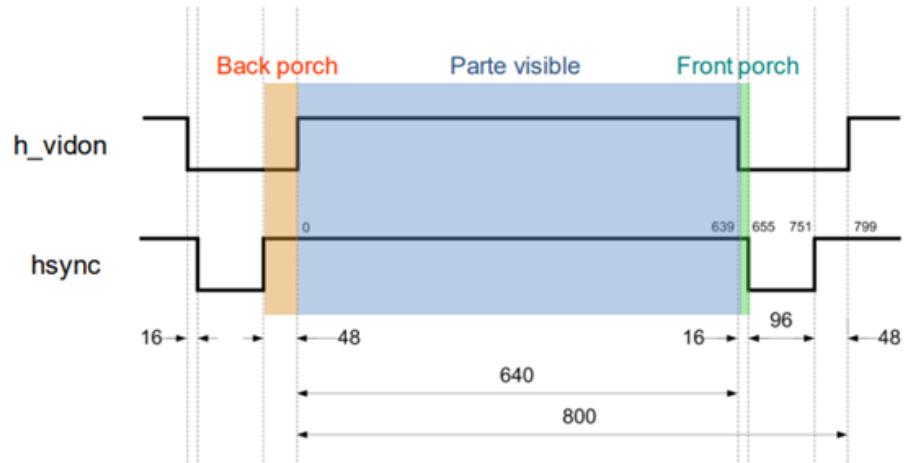


Figura 3: Pulsos de sincronismo horizontal.

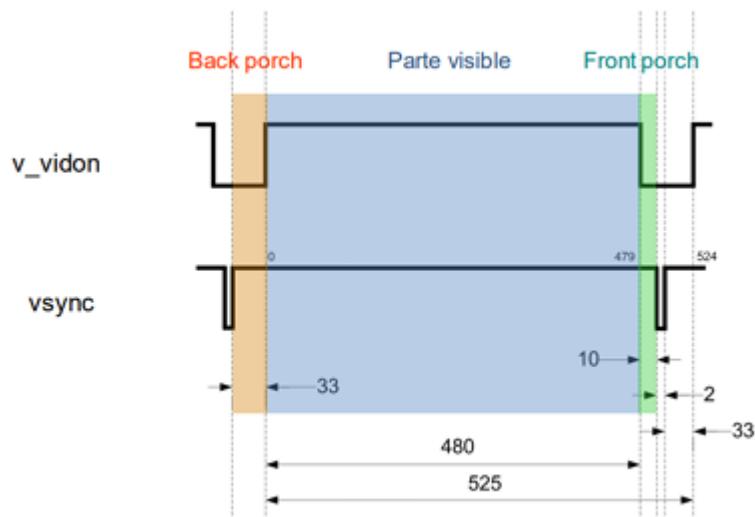


Figura 4: Pulsos de sincronismo horizontal.

- **VGA (vga\_controller.vhd)**

Bloque encargado de generar las señales de sincronismo y las coordenadas de los píxeles visibles (pixel X, pixel Y). En este diseño, se trabaja a una resolución estándar de 640x480. Las señales de color y sincronismo se conectan directamente al conector VGA de la FPGA.

- **Voltímetro (Voltmetro.vhd)**

Bloque principal que integra los componentes mencionados anteriormente necesarios para realizar la medición de tensión y su visualización.

En el esquema final, figura 11, se conectó un contador de 2 bits para convertir el reloj de entrada de 100 MHz en uno de 25 MHz, trabajando solo con el bit más significativo de la salida.

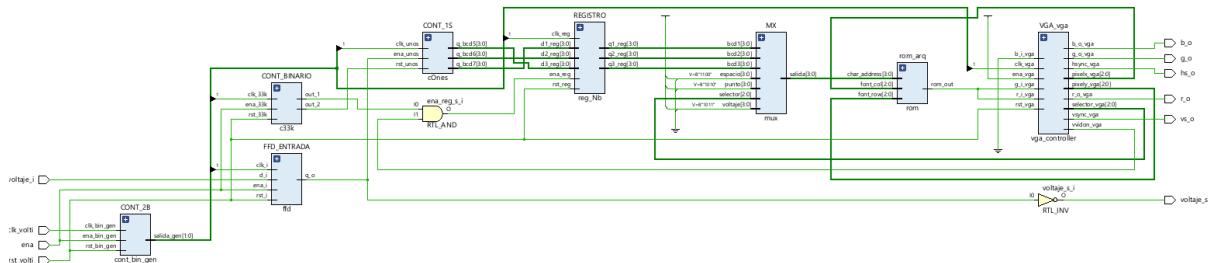


Figura 5: Esquema voltímetro

Los códigos desarrollados están en la carpeta “Códigos”.

Cada uno de los bloques mencionados fue implementado en VHDL y validado mediante simulaciones hechas usando las herramientas GHDL y GTKWave. Además, se diseñaron diagramas esquemáticos que me permitieron validar de forma visual la arquitectura interna y la conexión de señales.

Tanto las simulaciones como los esquemas fueron organizados en el anexo, donde se agrupan por nombre de componente. Los resultados simulados muestran que cada módulo responde de manera esperada frente a diferentes condiciones de entrada, lo que permitió avanzar con la integración final del voltímetro.

## c. Simulación

Se utilizó GTKWave para realizar las simulaciones de cada componente utilizando su banco de prueba correspondiente. La simulación más importante fue la del componente vga\_controller.vhd

En el tablero de pruebas del VGA, se comprobó el comportamiento de las señales de sincronismo para poder ver los píxeles en las zonas visibles. En esta primera parte de visualización me encontré con el error de medir los tiempos del sincronismo vertical sin considerar los tiempos del sincronismo horizontal lo que me generaba una discrepancia en los tiempos.

En la figura 6 se observa el comienzo de la zona visible del sincronismo horizontal, y en la figura 7 donde termina, quedando un tiempo de 25,6us correspondiente a esa zona.

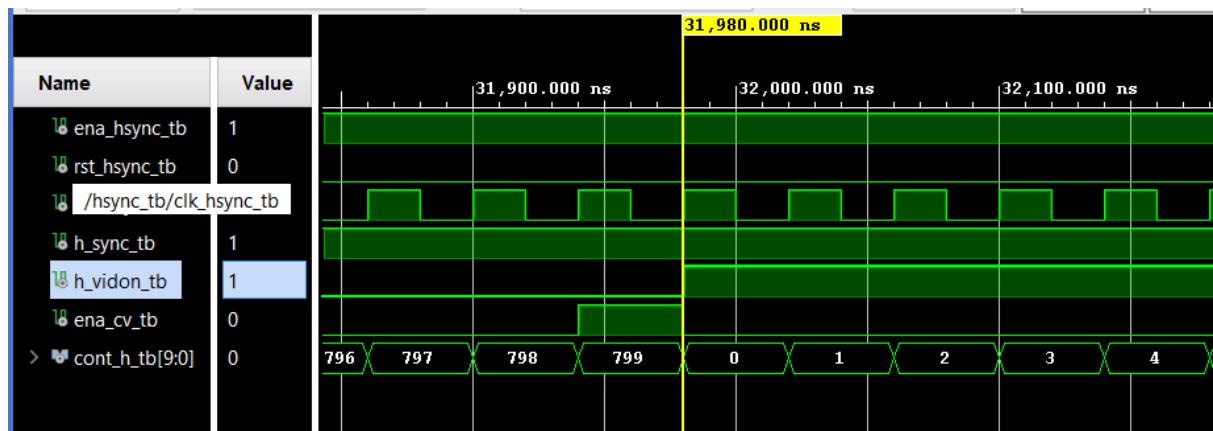


Figura 6: Inicio de la zona visible del sincronismo horizontal.

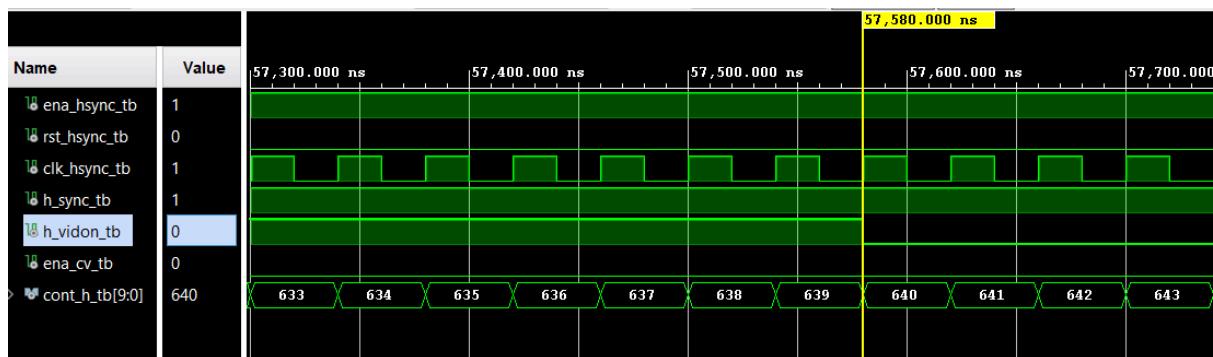


Figura 7: Fin de la zona visible del sincronismo horizontal.

Y en las figuras 8 y 9, se observa el comienzo y el final de la zona visible del sincronismo vertical, donde además se observa como actúa el contador horizontal y vertical entre si.

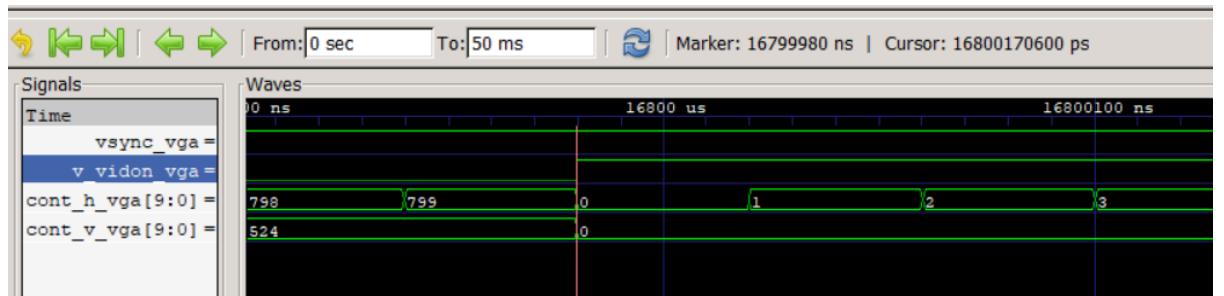


Figura 8: Inicio de la zona visible del sincronismo vertical.

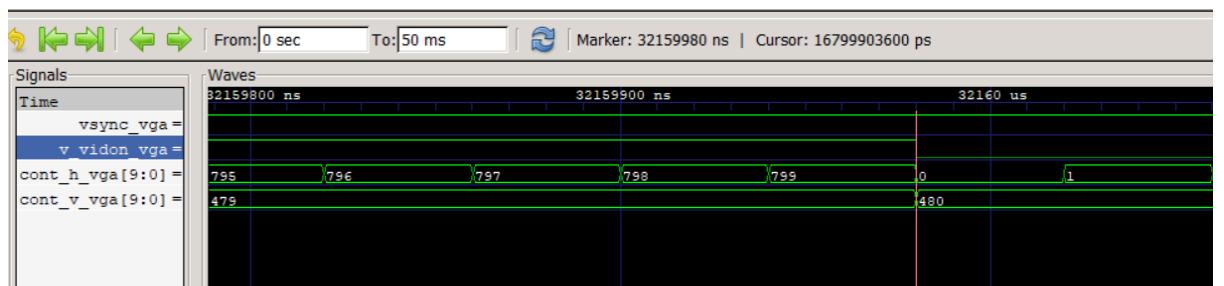


Figura 9: Fin de la zona visible del sincronismo vertical.

Además, al simular el tablero de pruebas del voltímetro, me encontré con un desfase en el reloj debido a que había definido erróneamente el contador que utilizaba .

Luego de modificarlo, ya pude observar el reloj necesario para el proyecto como se muestra en las figuras 10 y 11.

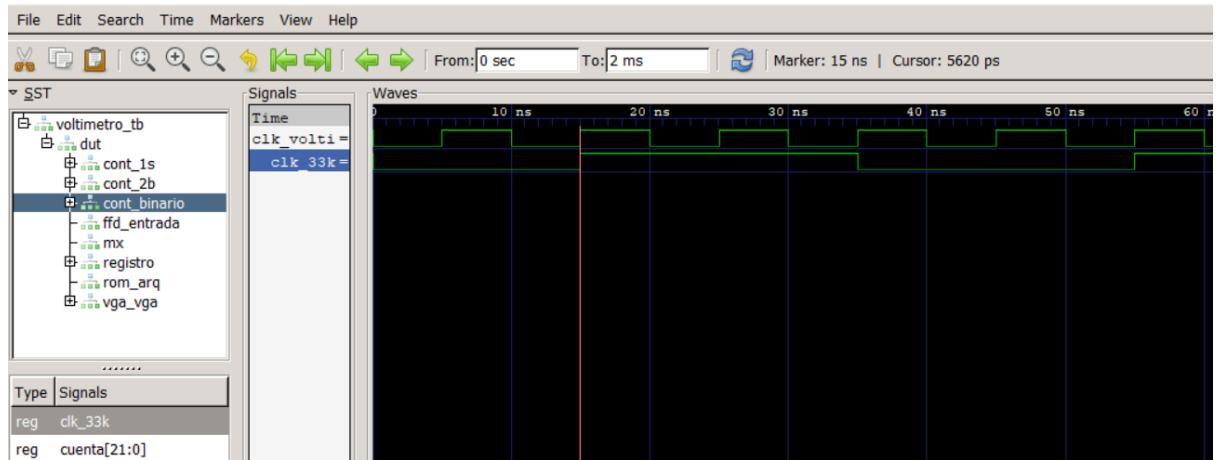


Figura 10: Inicio del primer ciclo del reloj.

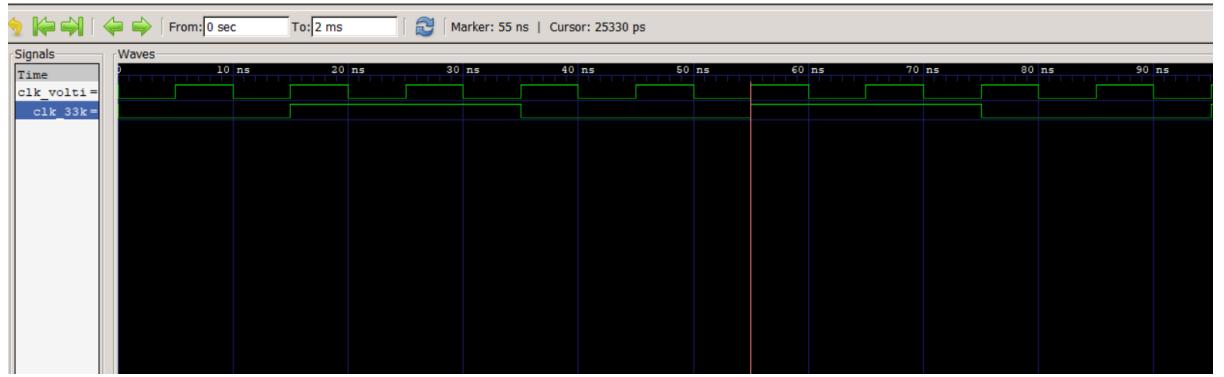


Figura 11: Fin del primer ciclo del reloj.

Tambien, se realizó la simulación completa utilizando el tablero de prueba, donde se observo el comportamiento esperado de las señales.



Figura 12: Simulación del voltímetro.

## d. Implementación

Luego de hacer las simulaciones, pasé a implementar el voltímetro en la FPGA en Vivado. Para ello, se diseñó el circuito analógico utilizando dos resistencias de  $10\text{k}\Omega$  y un capacitor de  $100\text{nF}$ , usando como referencia la figura 1 obteniendo la figura 13. Esta configuración permite su implementación, aunque introduce un offset en el voltímetro de aproximadamente 0,66V.



Figura 13: Conexión de placa y monitor para primeras pruebas.

A partir de Vivado, realice el archivo bitstream (.bit).

## e. Primeras pruebas con FPGA

Al implementar el proyecto tuve varias complicaciones. El primer gran error que tuve fue que al mostrar en pantalla solo aparecía "0.00V". Luego de utilizar el esquemático para guiarme y detectar dónde estaba el problema, pude encontrar que el error se debía a una mala conexión del voltímetro. En particular, había conectado la señal de enable del componente a '0', lo que lo mantenía desactivado desde la placa. Una vez corregido esto, el sistema comenzó a reflejar correctamente los cambios de tensión, que se observa en la siguiente figura.



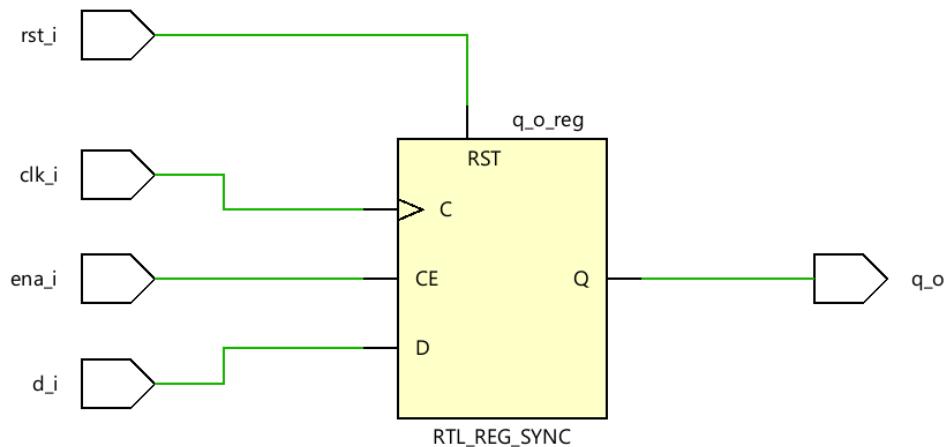
Figura 14: Voltímetro funcionando.

Además, en la figura se puede observar que el color del valor obtenido es amarillo que se debe a una definición errónea en las variables sobre el color en el proyecto.

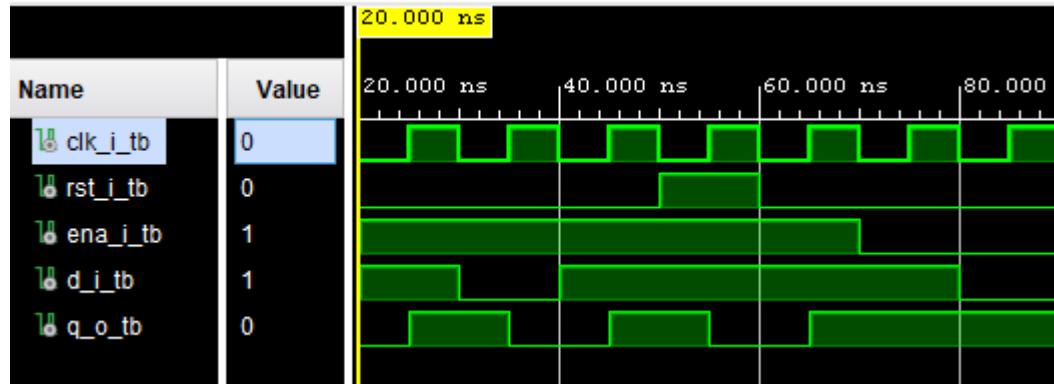
# Anexo

## Flip - Flop

### Esquemático:

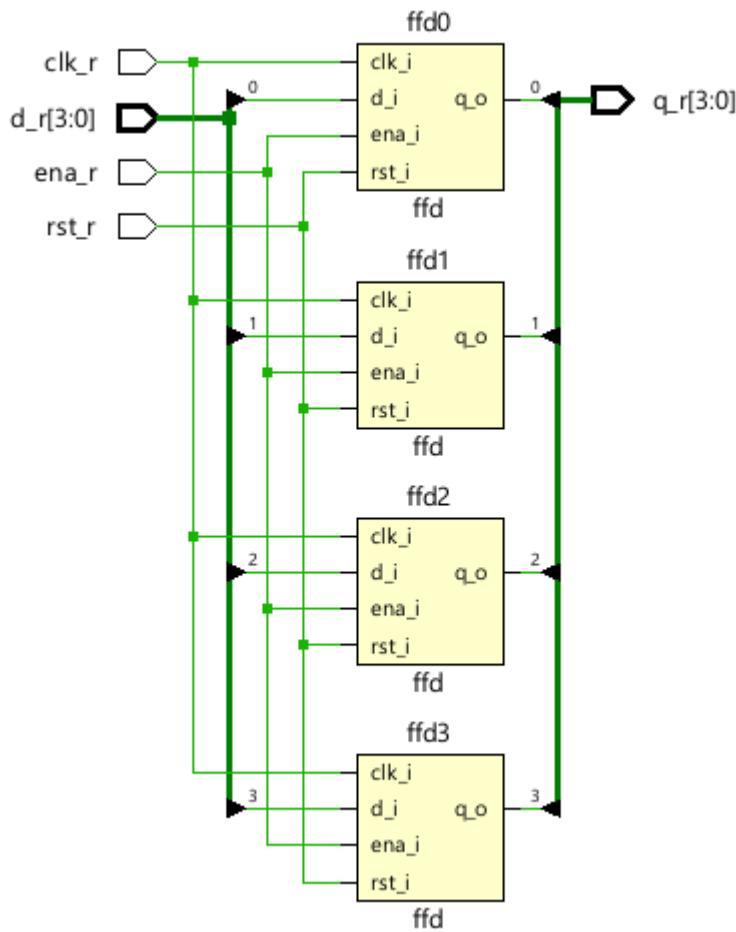


### Simulación:



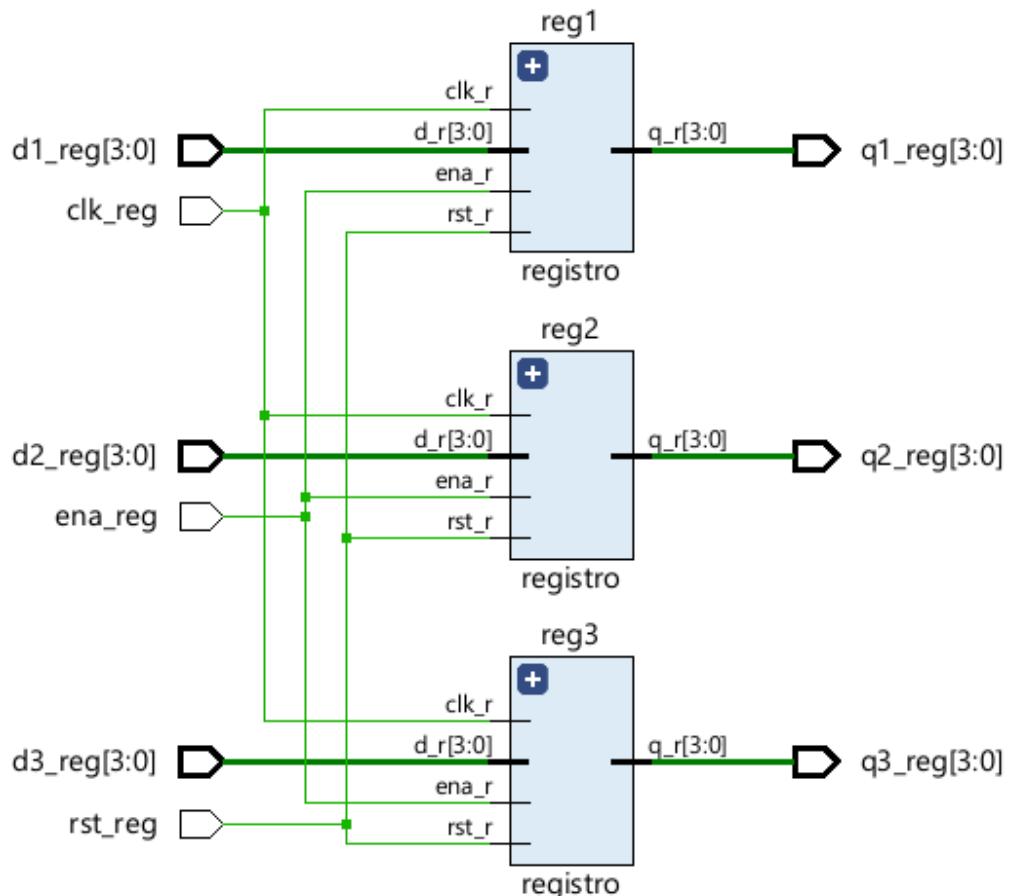
## Registro

### Esquemático:

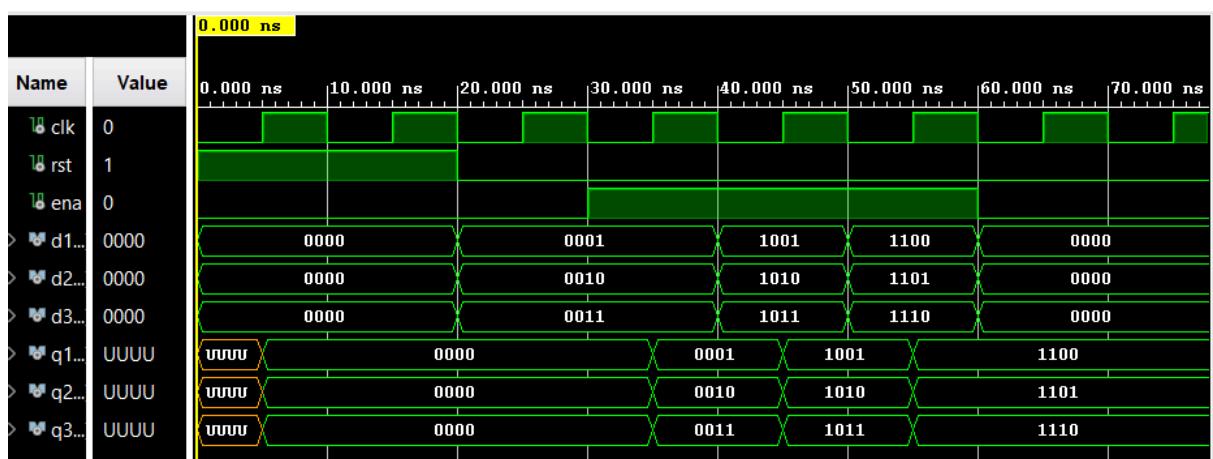


## Registro de 4 bits

### Esquemático:

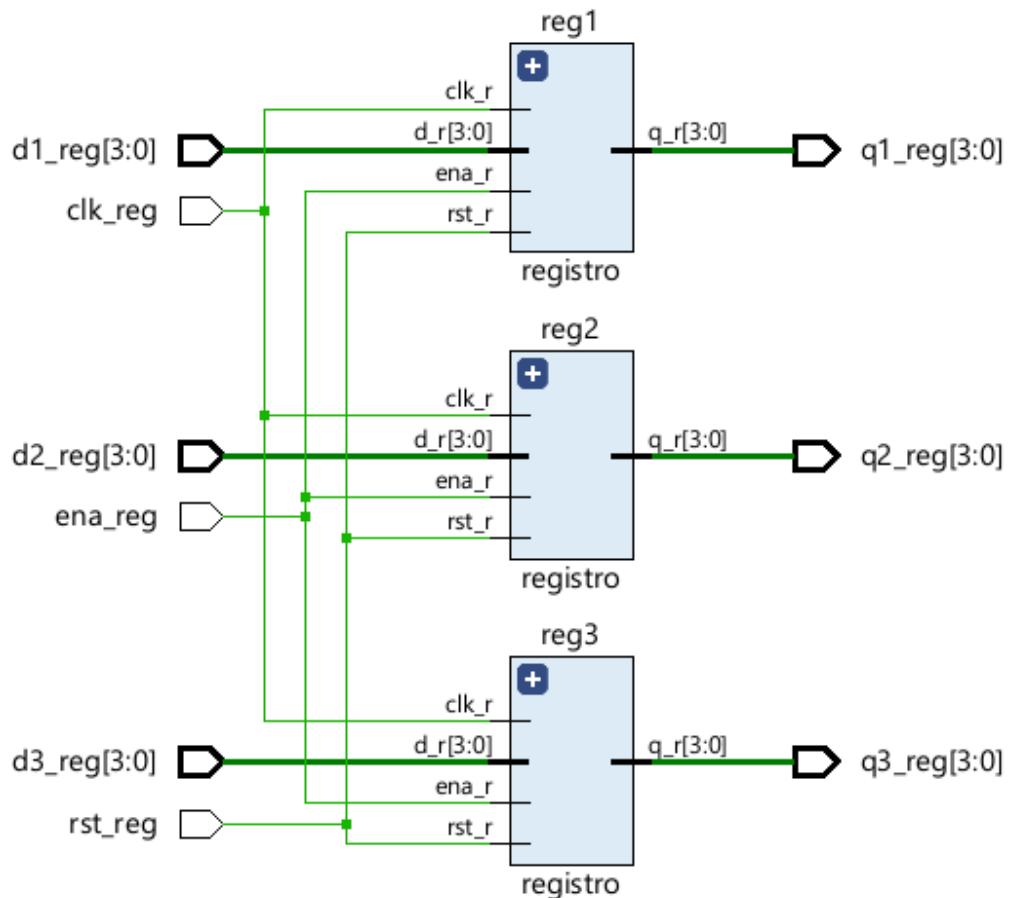


### Simulación:

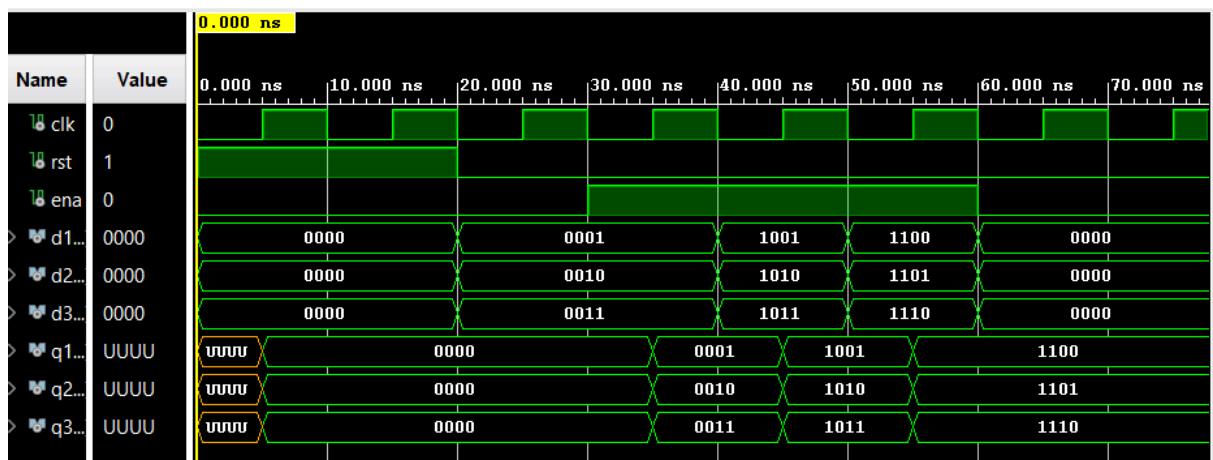


## Comparador de 10 bits

### Esquemático:

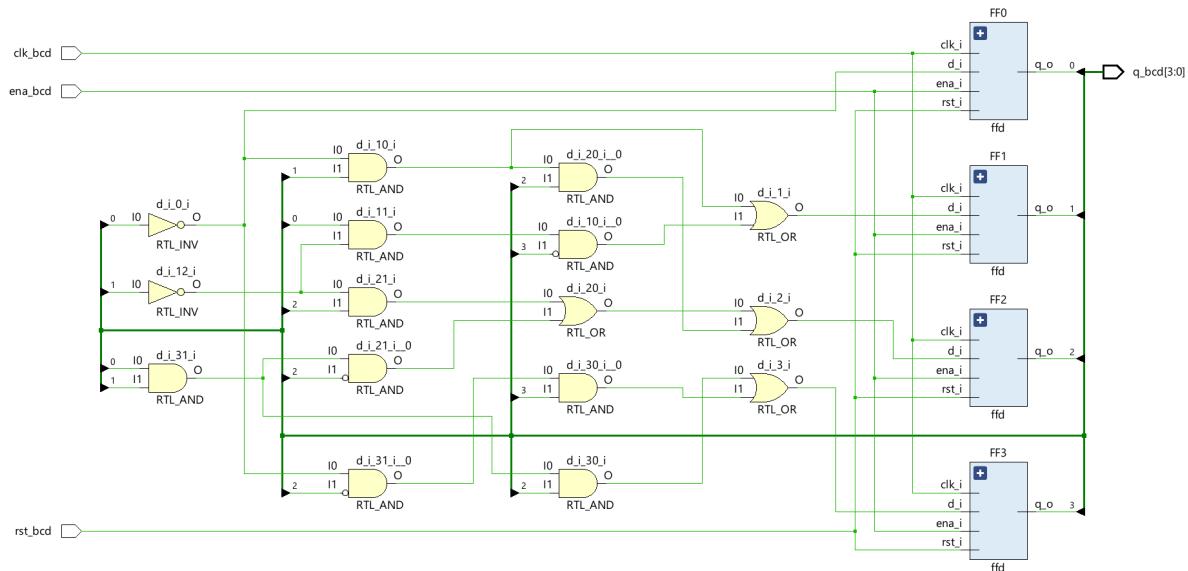


### Simulación:

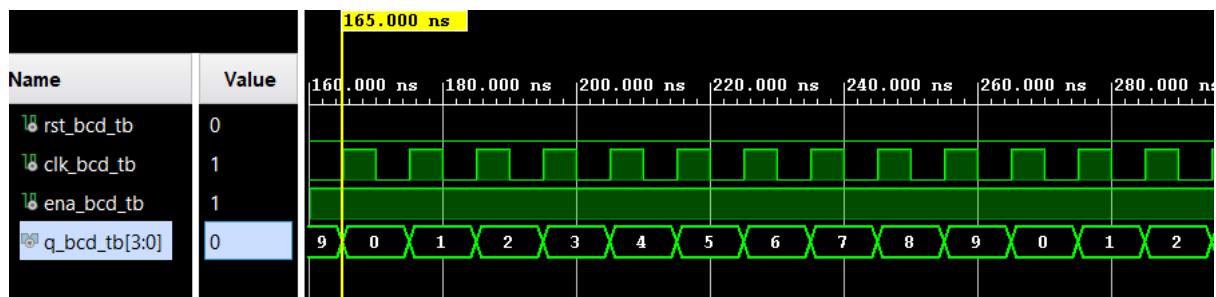


## Contador BCD

## Esquemático:

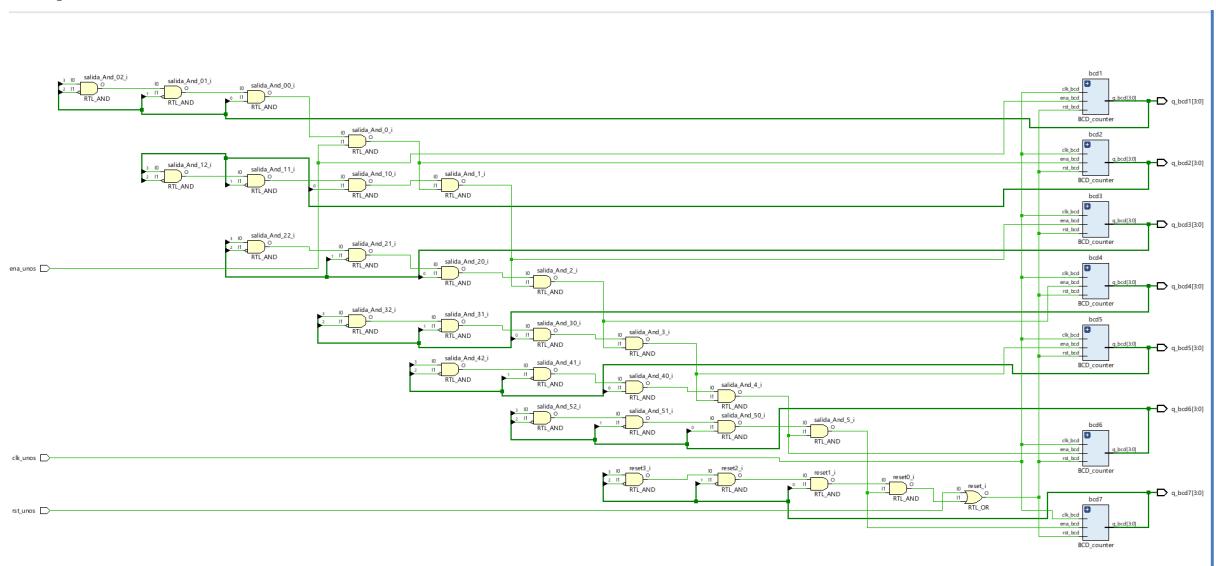


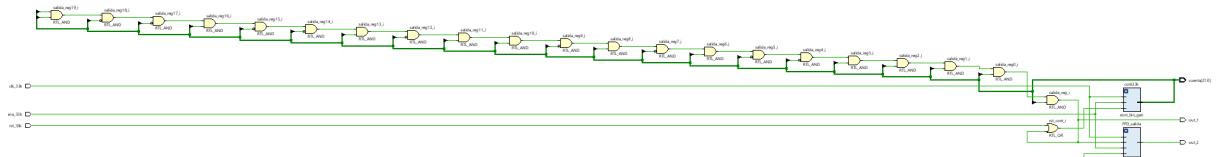
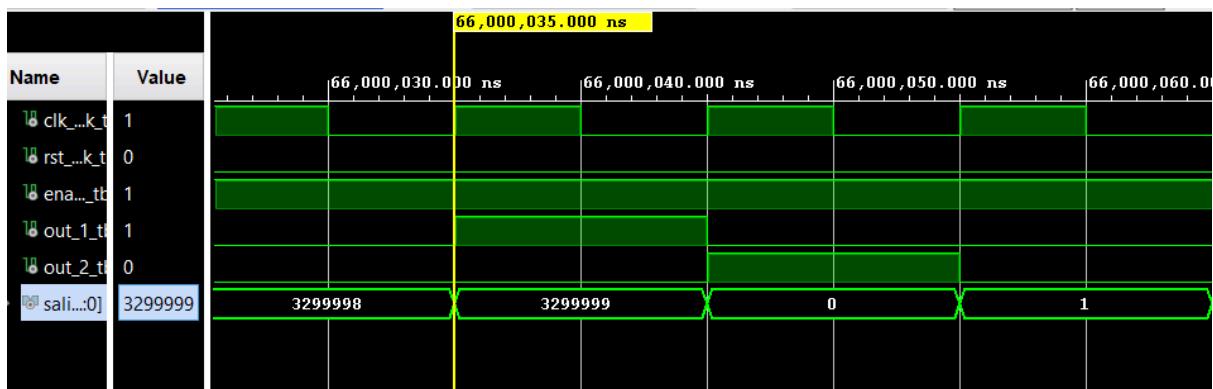
## Simulación :



# Contador de unos

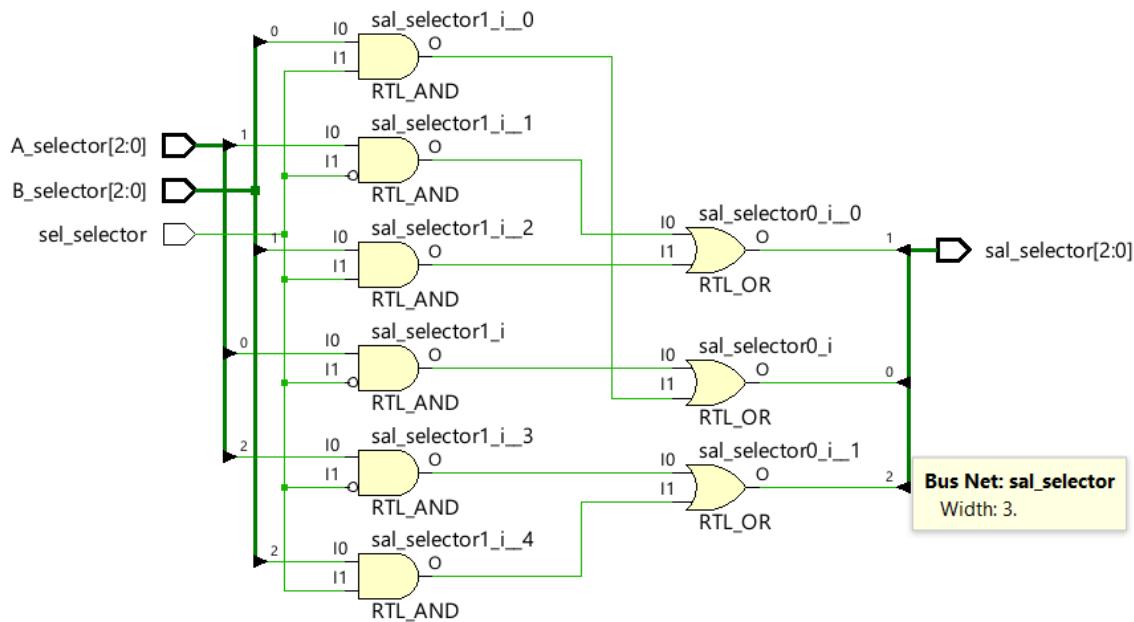
## Esquemático:



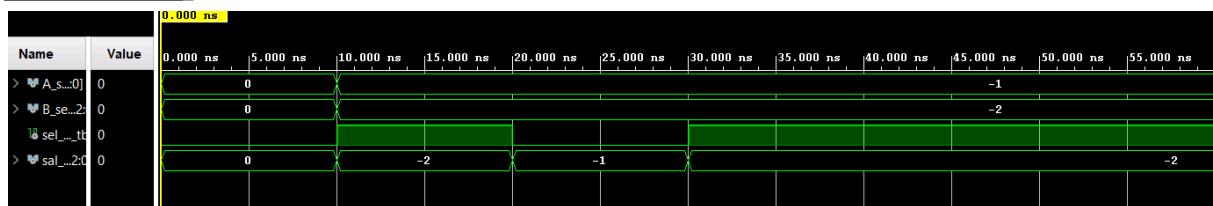
Simulación:**Contador 3 300 000**Esquemático:Simulación:

## Lógica MUX

### Esquemático:

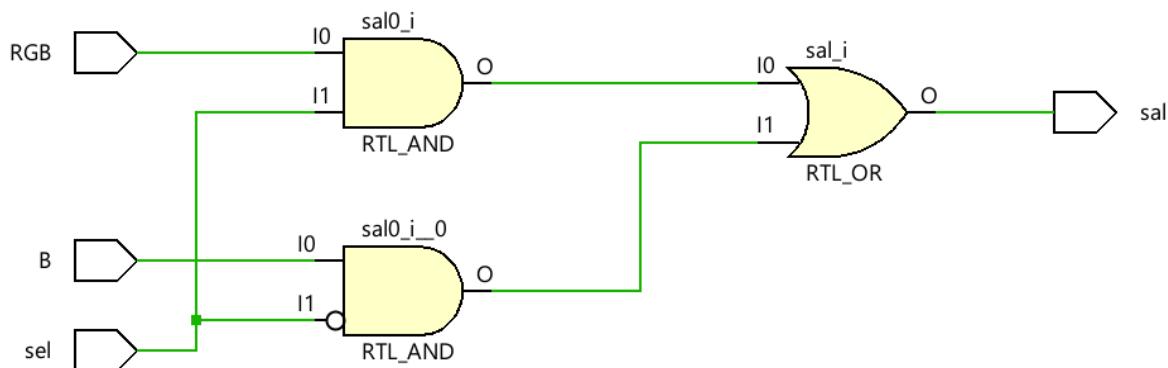


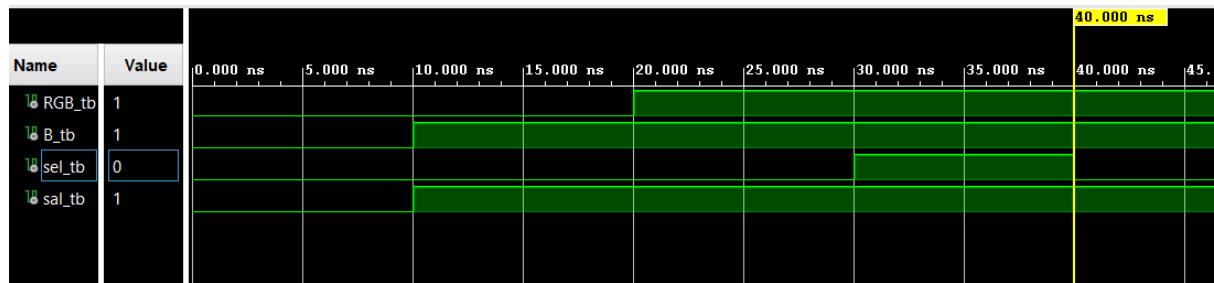
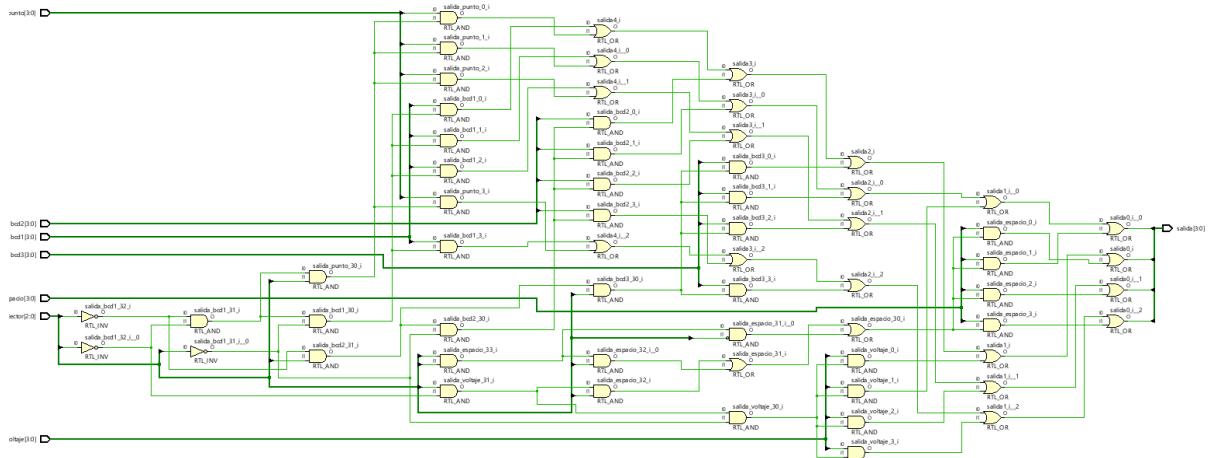
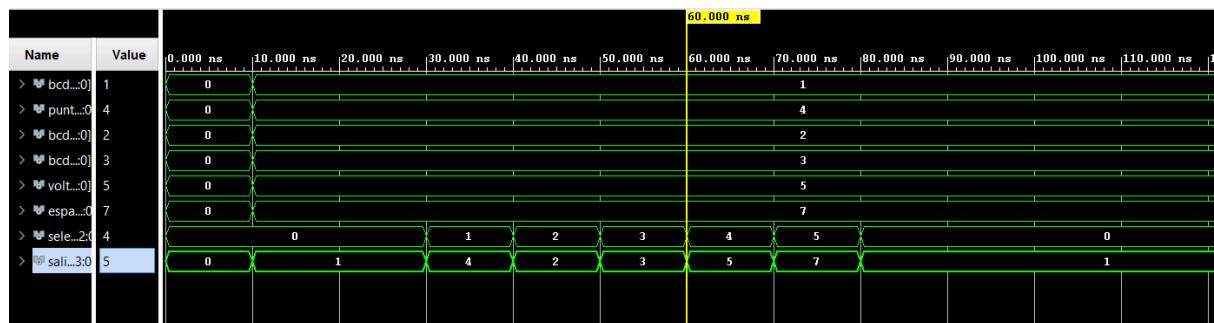
### Simulación:



## Color MUX

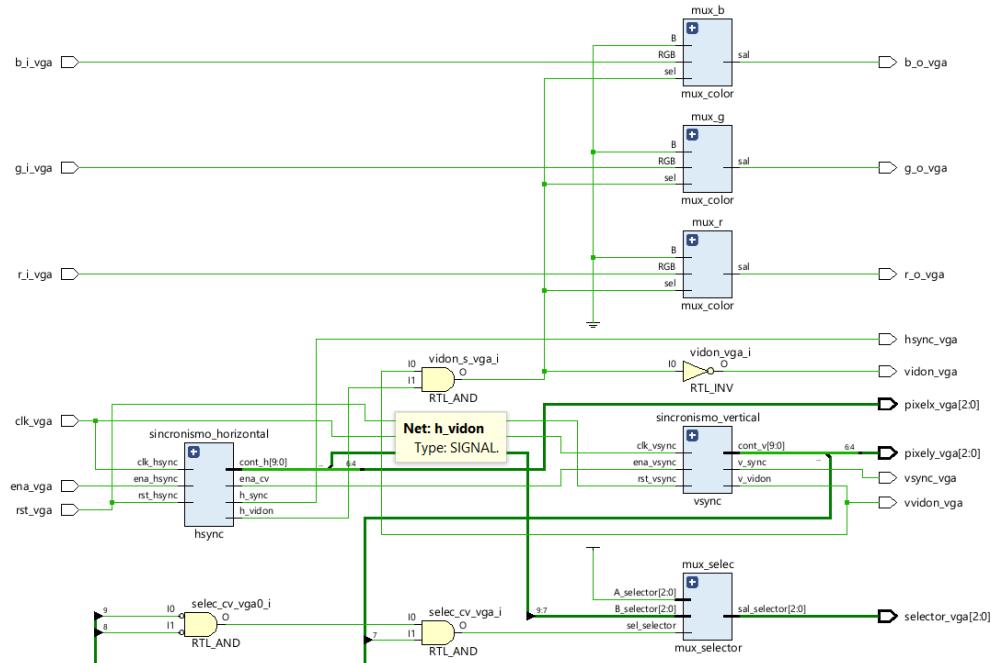
### Esquemático:



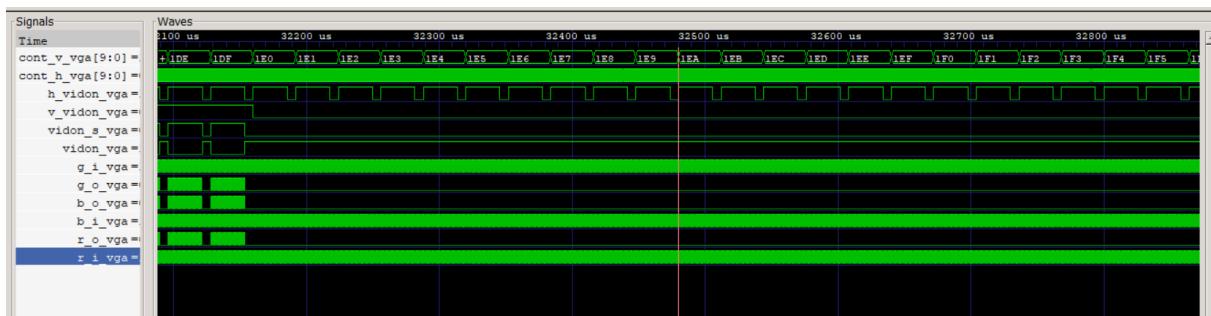
**Simulación:****MUX****Esquemático:****Simulación:**

## Controlador VGA

### Esquemático:

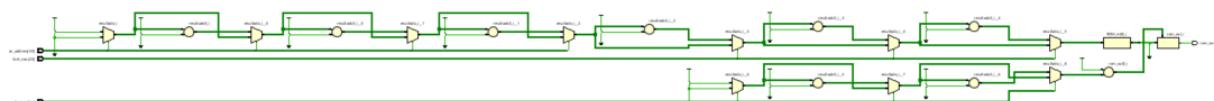


### Simulación:



## ROM - Caracteres

### Esquemático:

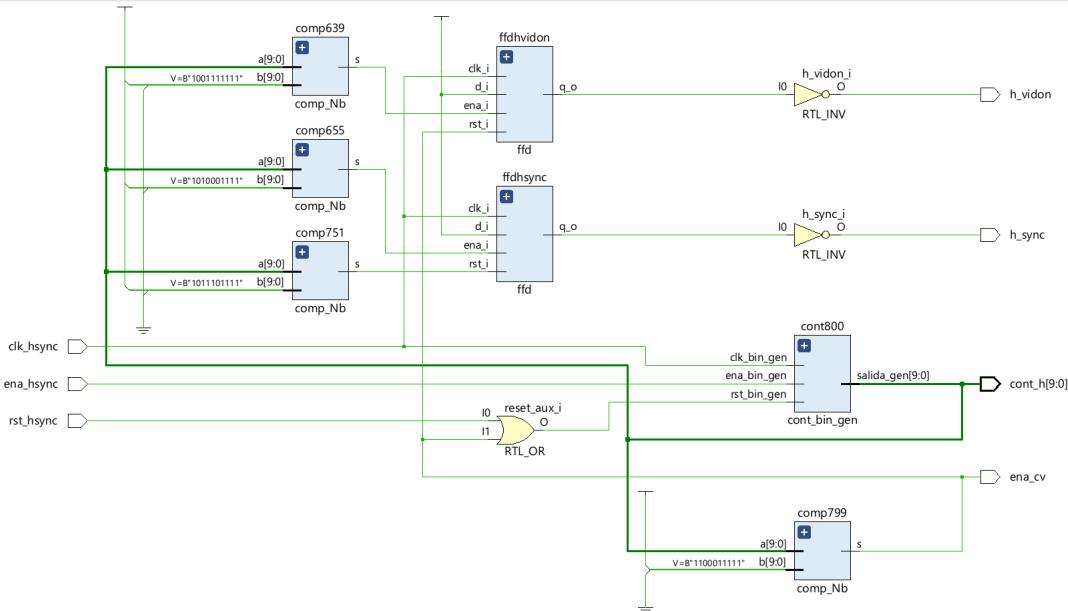


### Simulación:



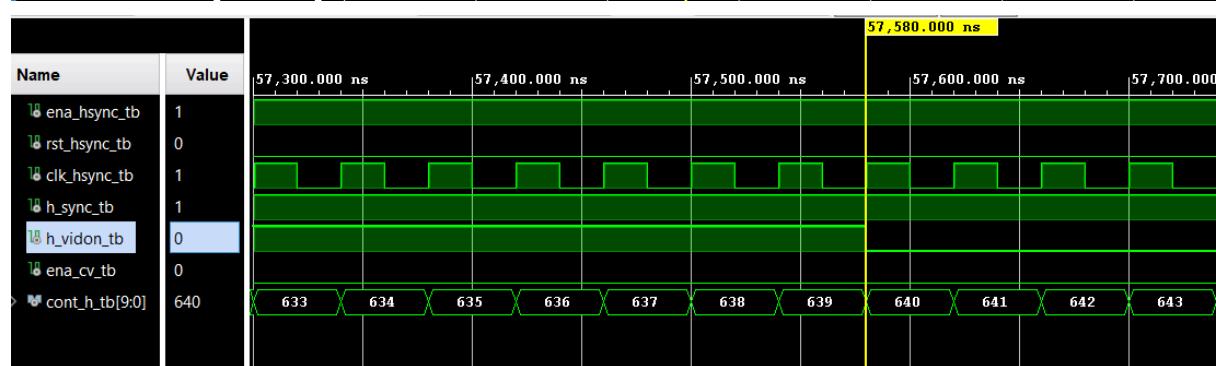
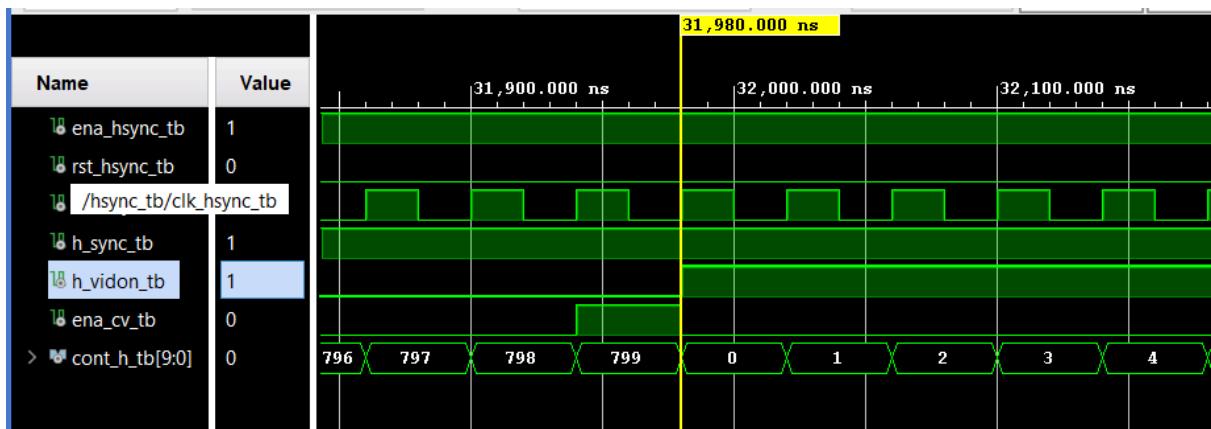
## Sincronismo horizontal

### Esquemático:



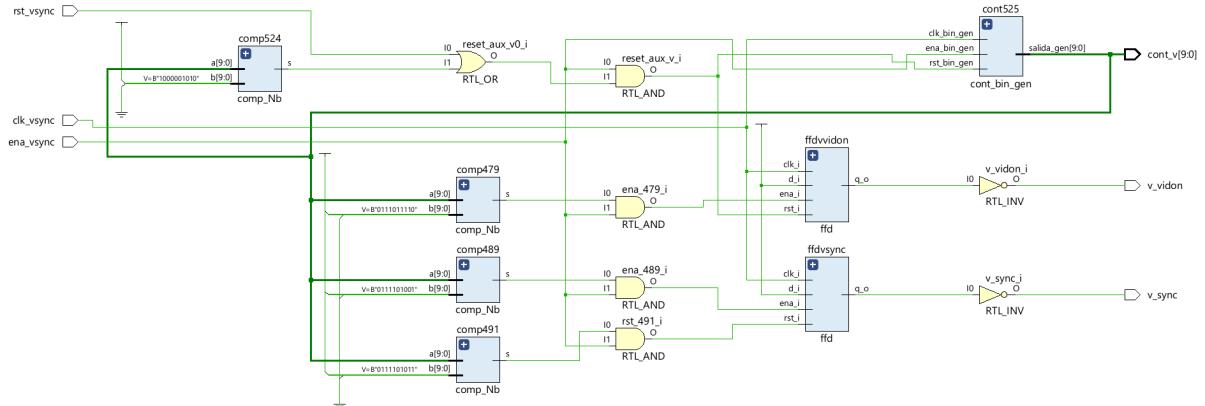
### Simulación:

#### Zona visible



# Sincronismo vertical

## Esquemático:



## Simulación:

### Zona visible

