



UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS

FACULTAD DE INGENIERÍA

Trabajo Final

GENERADOR DE FUNCIONES POR SÍNTESIS DIGITAL DIRECTA

Curso

Diseño Digital (EL255)

Profesor

Ing. Sánchez Huapaya Alonso Sebastián

Integrantes

Alumno	Código	Carrera
Reymundo Ramos, Renzo Edmundo	U202119710	Ingeniería Electrónica
Aldana Antezana, Andy Jesús	U202115092	Ingeniería Electrónica
Calixto Ruiz, Joastin Jeremi	U20201B551	Ingeniería Electrónica
Vargas Quispe, Alexandra Antonella	U202017257	Ingeniería Electrónica

Sección

EL61

2024-2

1.ÍNDICE

2. Introducción

3. Objetivos

4. Materiales y herramientas

5. Desarrollo del proyecto

5.1. Diagrama de bloques del sistema

5.2. Diagrama RTL

5.3. Cálculos teóricos realizados para el desarrollo del proyecto

5.4. Esquemático del circuito electrónico

5.5. Pin Planner

5.6. Sistema implementado

6. Evaluación del funcionamiento del sistema

6.1. Prueba Nro 1.

6.2. Prueba Nro 2.

7. Conclusiones

8. Referencias bibliográficas

2. Introducción

Este proyecto desarrolla un sistema de Síntesis Digital Directa (DDS) para generar formas de onda utilizando un FPGA, integrando un procesador Nios II para controlar la frecuencia y configurar el sistema. La DDS es una técnica que genera formas de onda análogas manipulando digitalmente un reloj y pasando la salida por un convertidor digital-analógico. Esta técnica ofrece ventajas como cambios rápidos entre frecuencias y alta resolución de frecuencia en un amplio rango. La DDS incluye dos componentes principales: un acumulador de fase y una LUT o ROM, donde se almacenan los valores muestreados correspondientes a las diversas señales a generar. Asimismo, este proyecto se tratará de un sistema el cual permita la salida de tres tipos diferentes de señales (senoidal, diente de sierra y triangular) a través de un selector (botones) en la entrada.

3. Objetivos

- a) Diseño del Sistema DDS:
 - Investigar sobre la técnica de DDS mediante fuentes y videos.
 - Implementar un sistema DDS en un FPGA capaz de generar ondas senoidales, triangulares y diente de sierra.
- b) Integración de la Selección de Formas de Onda:
 - Configurar pulsadores para seleccionar la forma de onda mediante el procesador Nios II.
 - Implementar la lógica en VHDL para cambiar entre las diferentes formas de onda según el pulsador.
- c) Investigación sobre el Convertidor PWM:
 - Investigar el módulo PWM y desarrollar su diagrama de funcionamiento.
- d) Integración de la Comunicación UART:
 - Implementar la comunicación UART entre el procesador Nios II y la computadora.
 - Verificar que los comandos se envíen y reciban para comprobar el control de la frecuencia de la señal.
- e) Simulación del Sistema:
 - Simular el módulo DDS junto con el convertidor PWM y la comunicación UART.
 - Verificar que las formas de onda generadas cumplan con las especificaciones.

4. Materiales y Herramientas

- a) Hardware:
 - Pulsadores:
Un pulsador es un dispositivo eléctrico con forma de botón que, al ser presionado, puede activar o desactivar los circuitos eléctricos a los que está conectado. En este caso, se usarán para seleccionar el tipo de forma de onda (senoidal, diente de sierra, triangular) y ajustar la frecuencia.
 - Resistencia 10K:

La resistencia es una medida de la oposición al flujo de corriente en un circuito eléctrico. En este caso, se usará una resistencia para implementar un filtro pasa-bajas pasivo de 1er orden RC.

- Capacitor 0.1 μ F:

Un capacitor es un dispositivo que almacena energía en un campo eléctrico interno, utilizado frecuentemente en circuitos electrónicos, tanto analógicos como digitales. En este caso, se usará este capacitor para implementar un filtro pasa-bajas pasivo de 1er orden RC.

- FPGA EP2C5T144C8:

El FPGA de la serie Cyclone II del modelo EP2C5T144C8 es el encargado de implementar la síntesis digital directa para producir las formas de onda necesarias.

- MÓDULO USB A TLL FT232RL:

Este módulo conversor serial es una placa de conexión básica para FTDI FT232RL USB a serial IC. También se puede utilizar para aplicaciones seriales generales

b) Software:

- Quartus II:

Será necesario para la síntesis, implementación y simulación del diseño en el FPGA.

- Eclipse:

Este será necesario para codificar la lógica de comunicación entre la PC y el UART para el control del tamaño de frecuencia administrado al sistema.

- Procesador Nios II:

Se utiliza para controlar y configurar el sistema. Maneja la lógica de control y la interfaz con el módulo UART. Es un procesador configurable en el FPGA que proporciona la interfaz de control.



Fig 1. Materiales y Herramientas

5. Desarrollo del proyecto:

5.1. Diagrama de bloques del sistema

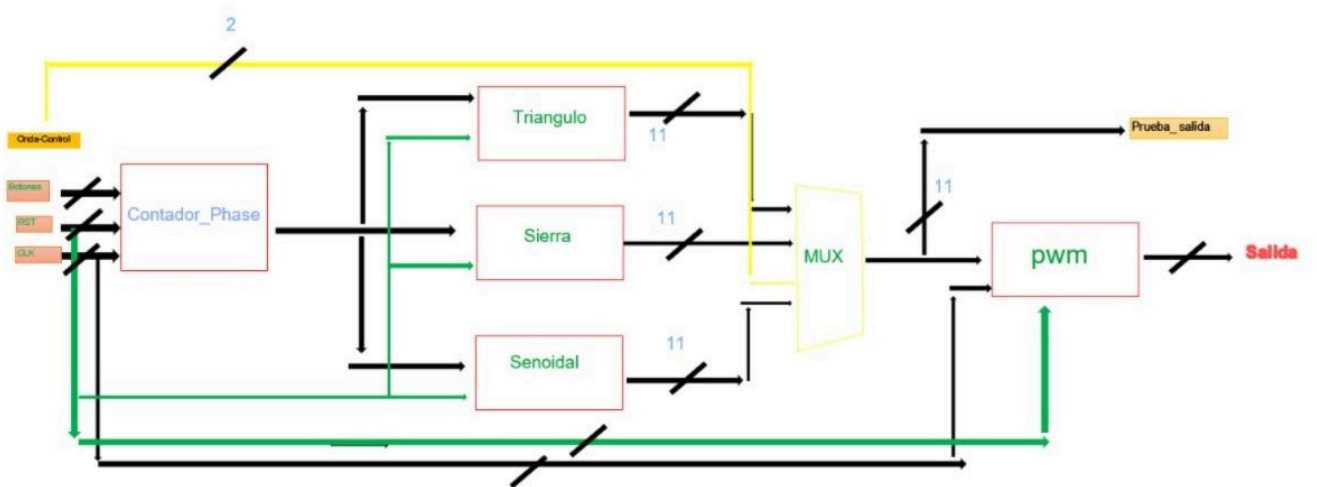


Fig 2. Diagrama de Bloques del sistema

En la figura 2, el diagrama de bloques representa un sistema generador de señales que utiliza un contador como base para generar formas de onda (triangular, senoidal y de sierra). Estas señales son seleccionadas mediante un multiplexor y procesadas en un módulo PWM para producir una señal de salida controlada.

5.2. Diagrama RTL de los bloques digitales desarrollados

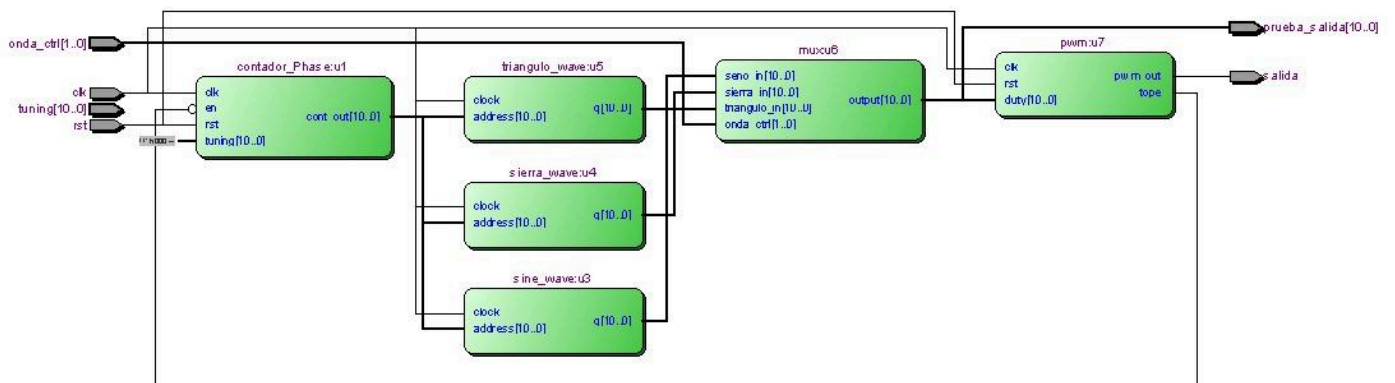


Fig 3. Diagrama de RTL del sistema

En la figura 3 notamos que el sistema genera diferentes formas de ondas digitales (senoidal, triangular y diente de

sierra) y permite seleccionar cuál de ellas se utiliza mediante un multiplexor controlado por “onda_ctrl”. La forma de onda seleccionada es modulada utilizando un esquema PWM para producir una señal de salida ajustada en frecuencia y amplitud.

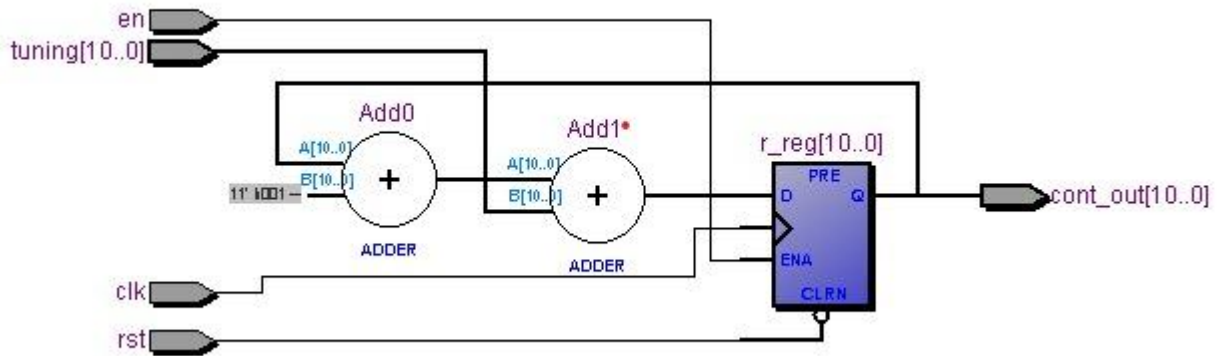
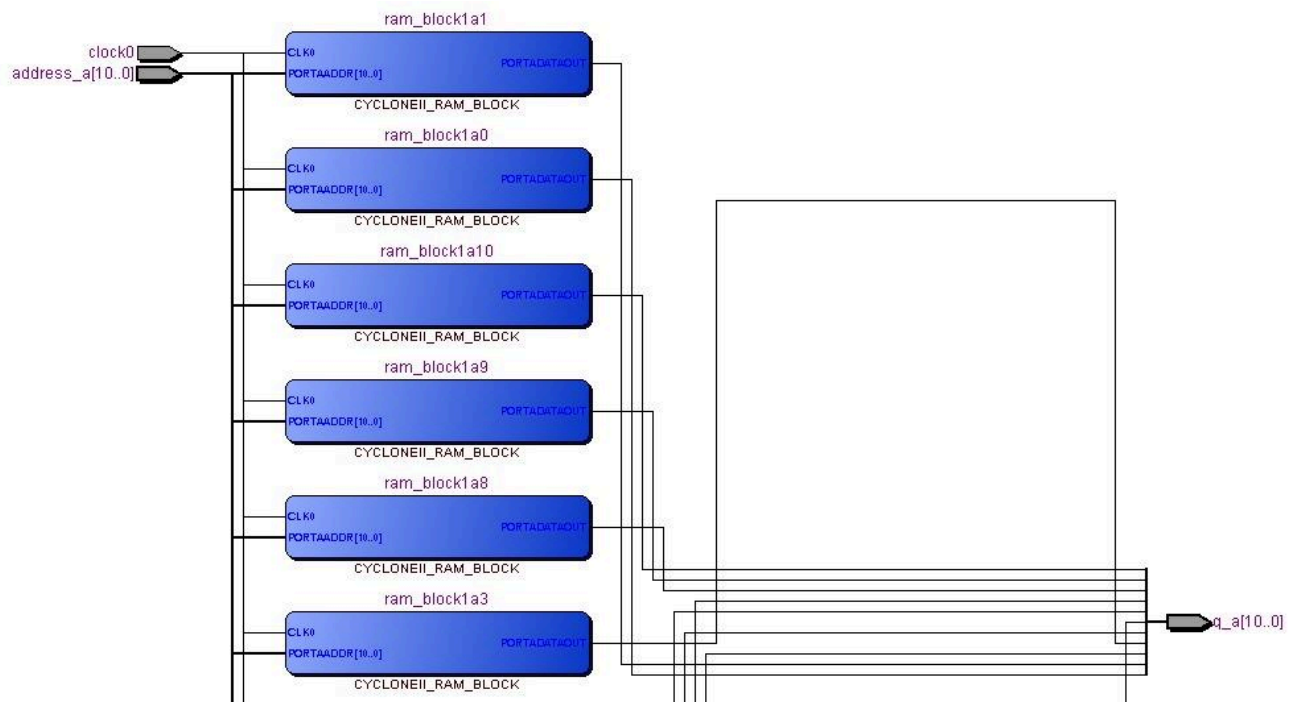


Fig 4. RTL del contador_phase

En la figura 4 notamos un contador configurable que se incrementa controladamente por la señal clk. El incremento está determinado por la entrada “tuning”, que define el paso de cada ciclo. Si la señal de habilitación en está activada, el contador avanza sumando el valor de tuning y un incremento base, y si se activa la señal de reinicio (rst), el contador vuelve a su estado inicial. Cada ciclo se almacena en el registro r_reg, que mantiene el estado del contador.



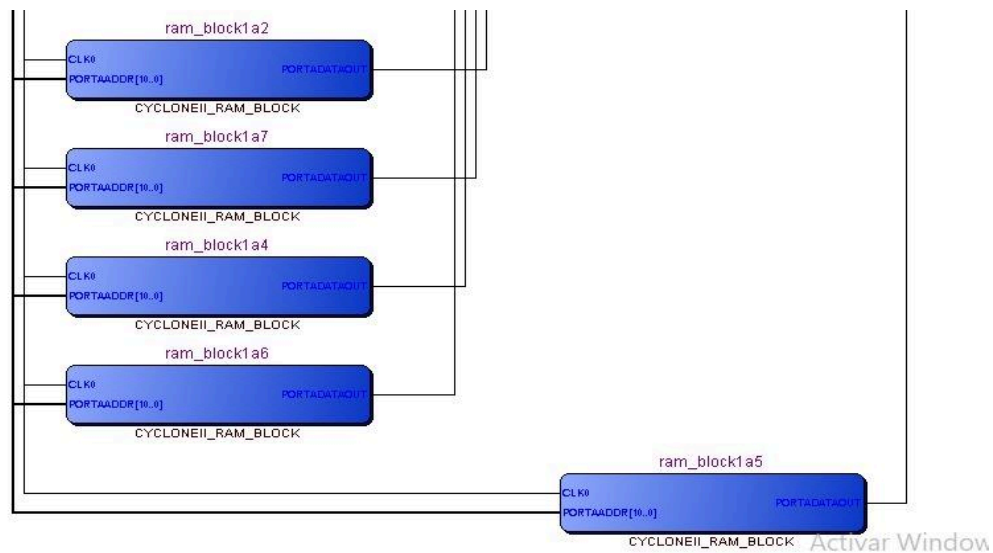
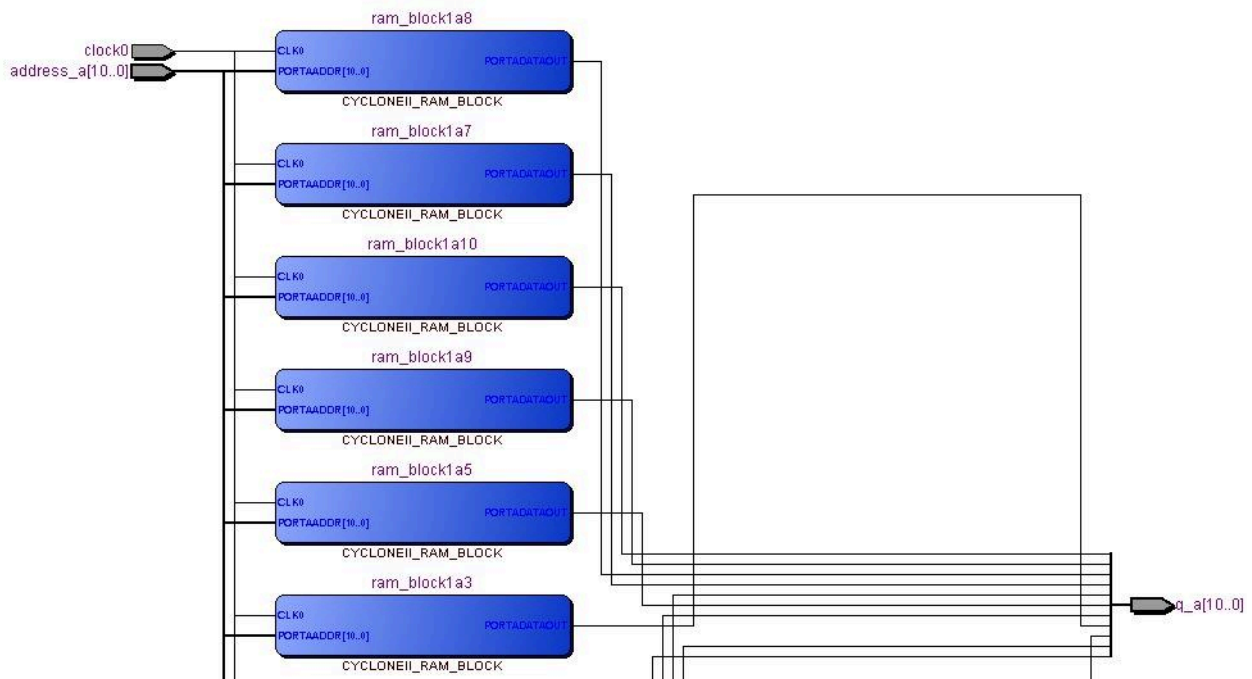


Fig 5. RTL de la señal de diente de sierra

En la figura 5, el diagrama implementa un generador de señal de diente de sierra basado en bloques de memoria organizados jerárquicamente, donde cada bloque almacena segmentos de la señal en forma de tablas de búsqueda (LUTs). Utiliza una señal de dirección, sincronizada con un reloj (clk), para indexar y recorrer los valores almacenados.



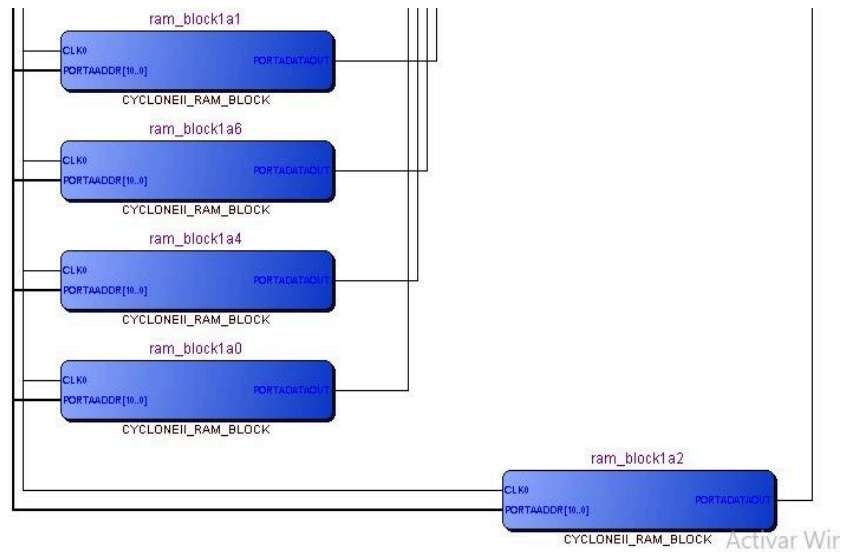
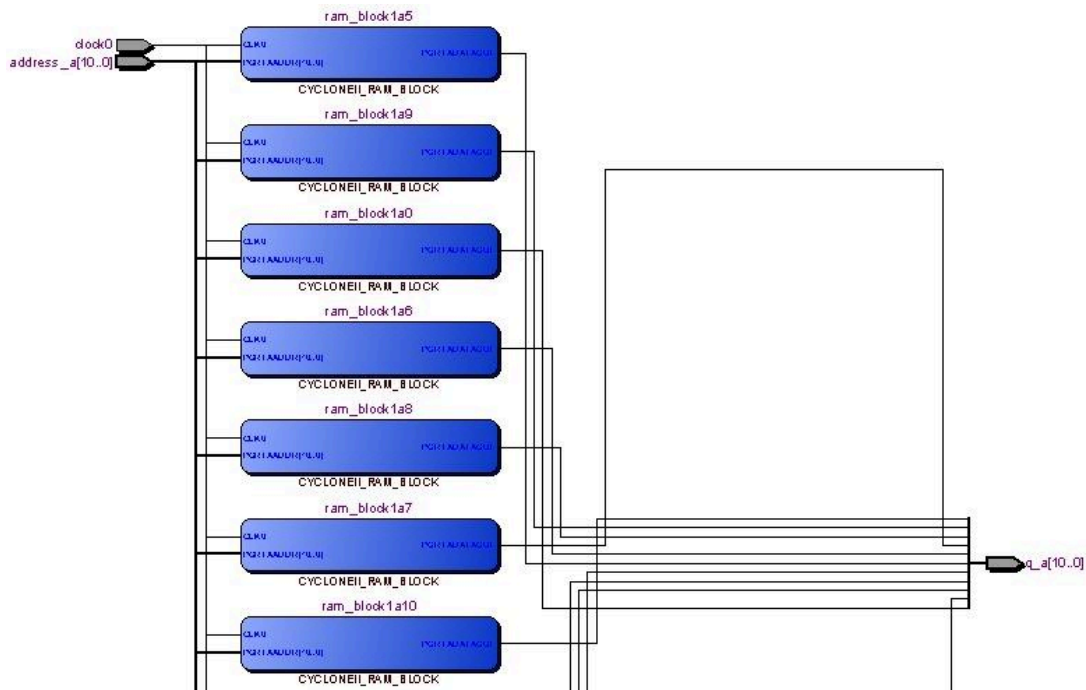


Fig 6. RTL de la señal senoidal

En la figura 6, el diagrama implementa un generador de señal de onda senoidal basado en bloques de memoria organizados jerárquicamente, donde cada bloque almacena segmentos de la señal en forma de tablas de búsqueda (LUTs). Utiliza una señal de dirección, sincronizada con un reloj (clk), para indexar y recorrer los valores almacenados.



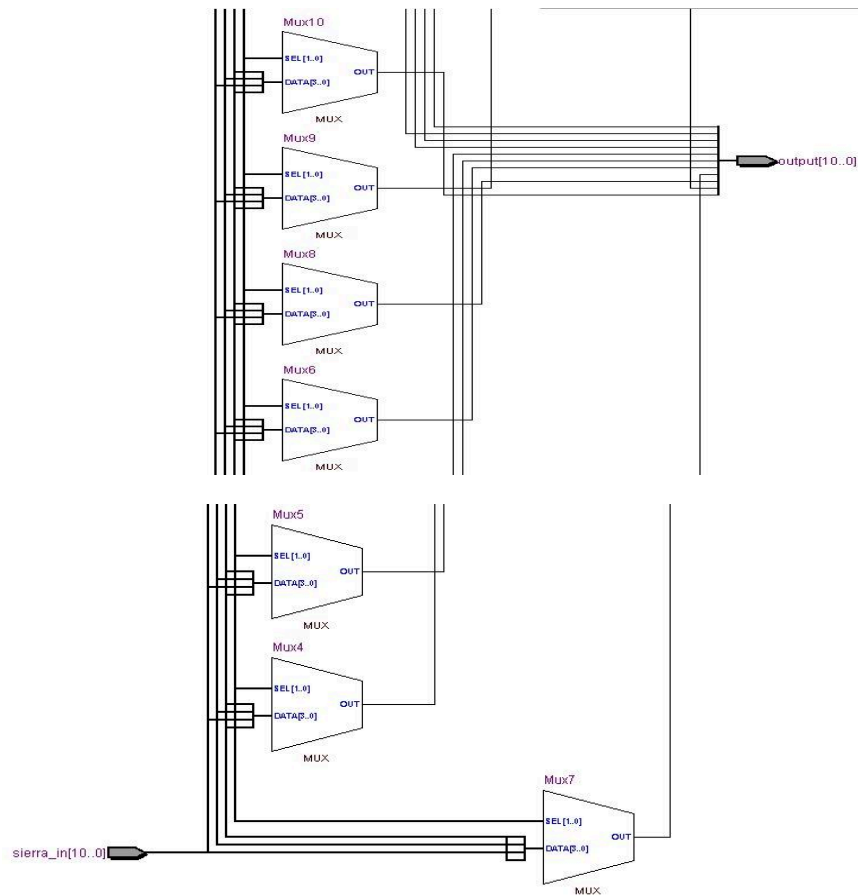


Fig 8. RTL del multiplexor

En la figura 8, el diagrama corresponde a un multiplexor, el cual permite seleccionar entre múltiples entradas de datos y direccionar una única salida según señales de control. Cada etapa del diseño emplea selectores para manejar las señales de entrada en forma jerárquica, permitiendo escalar el número de entradas manejadas.

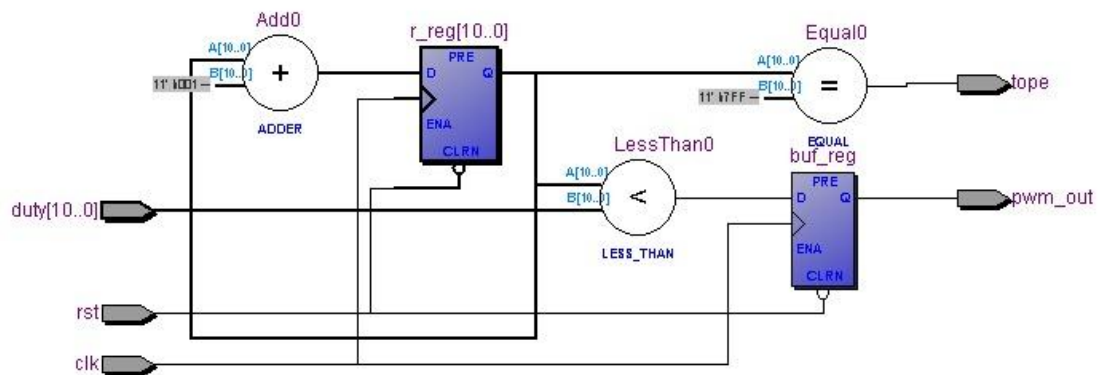


Fig 9. RTL del pwm

En la figura 9 notamos que el diagrama RTL describe un módulo de modulación por ancho de pulso (PWM), que genera una señal modulada a partir de una entrada de ciclo de trabajo “duty”.

5.3. Cálculos teóricos realizados para el desarrollo del proyecto

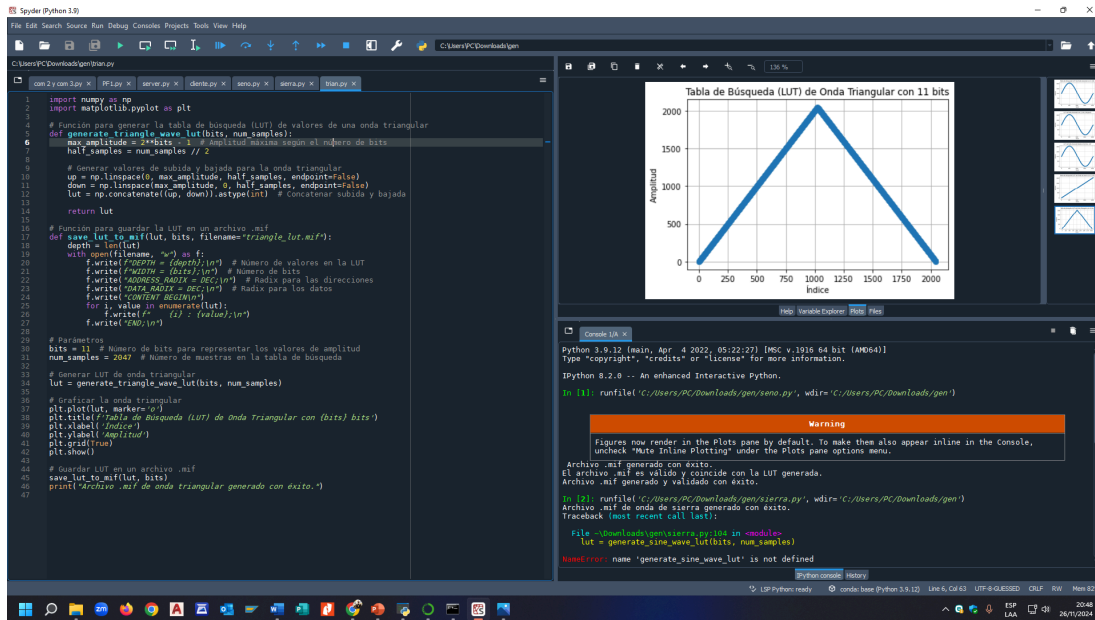


Fig 10. Cálculos de onda triangular Python

En la figura 10, el código en Python genera una tabla de búsqueda (LUT - Lookup Table) para una onda triangular basada en un número específico de bits y muestras, con la opción de exportarla a un archivo de formato .mif.

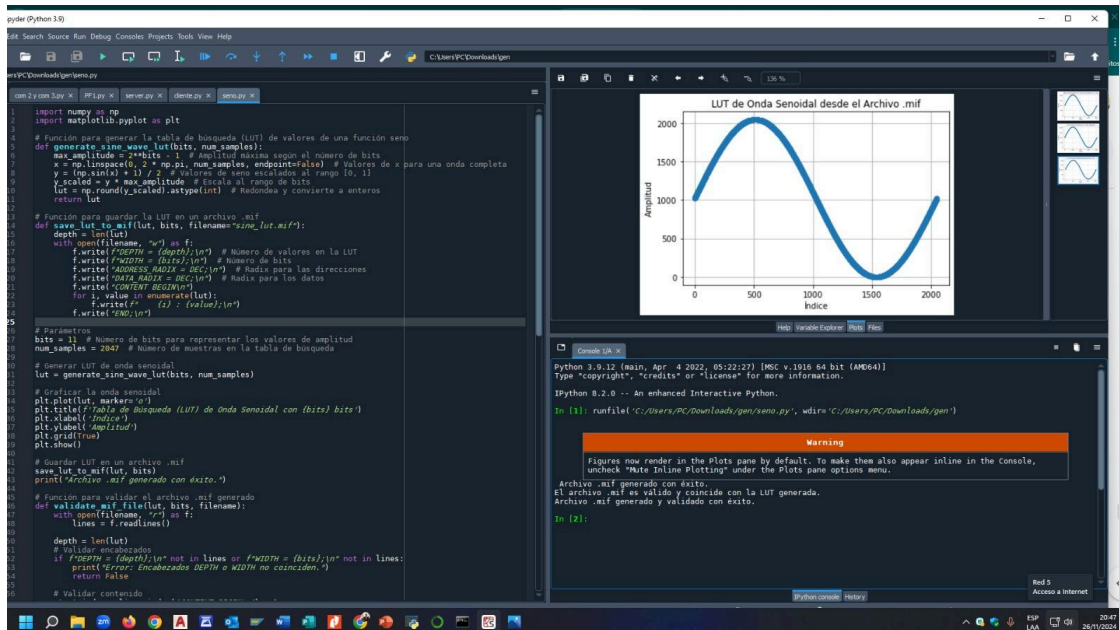


Fig 11. Cálculos de onda Senoidal en Python

En la figura 11, el código en Python genera una tabla de búsqueda (LUT - Lookup Table) para una onda senoidal basada en un número específico de bits y muestras, con la opción de exportarla a un archivo de formato .mif.

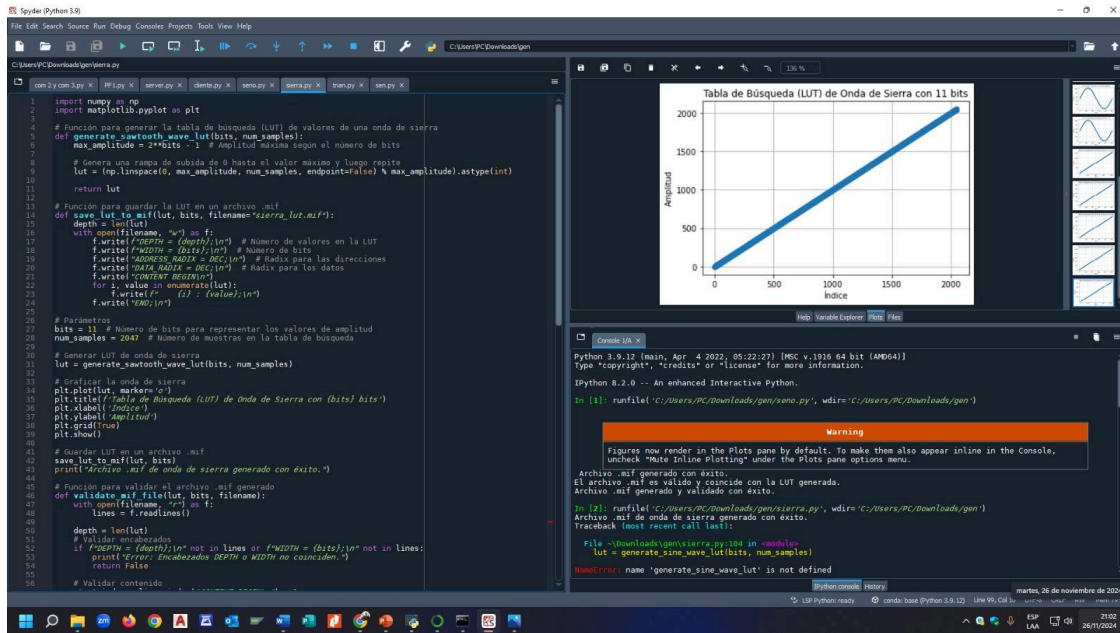


Fig 12. Cálculos de onda diente de sierra en Python

En la figura 12, el código en Python genera una tabla de búsqueda (LUT - Lookup Table) para una onda diente de sierra basada en un número específico de bits y muestras, con la opción de exportarla a un archivo de formato .mif.

5.4. Esquemático del circuito electrónico:

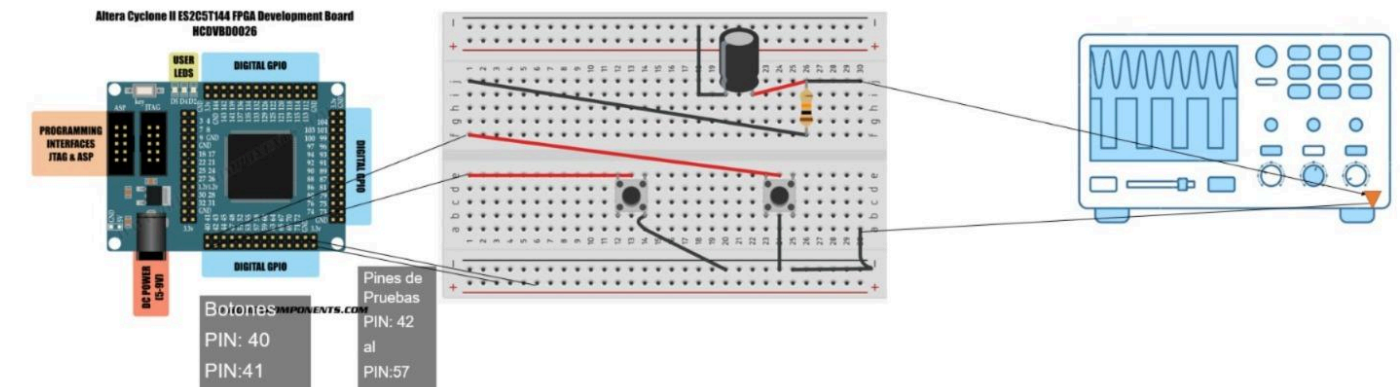
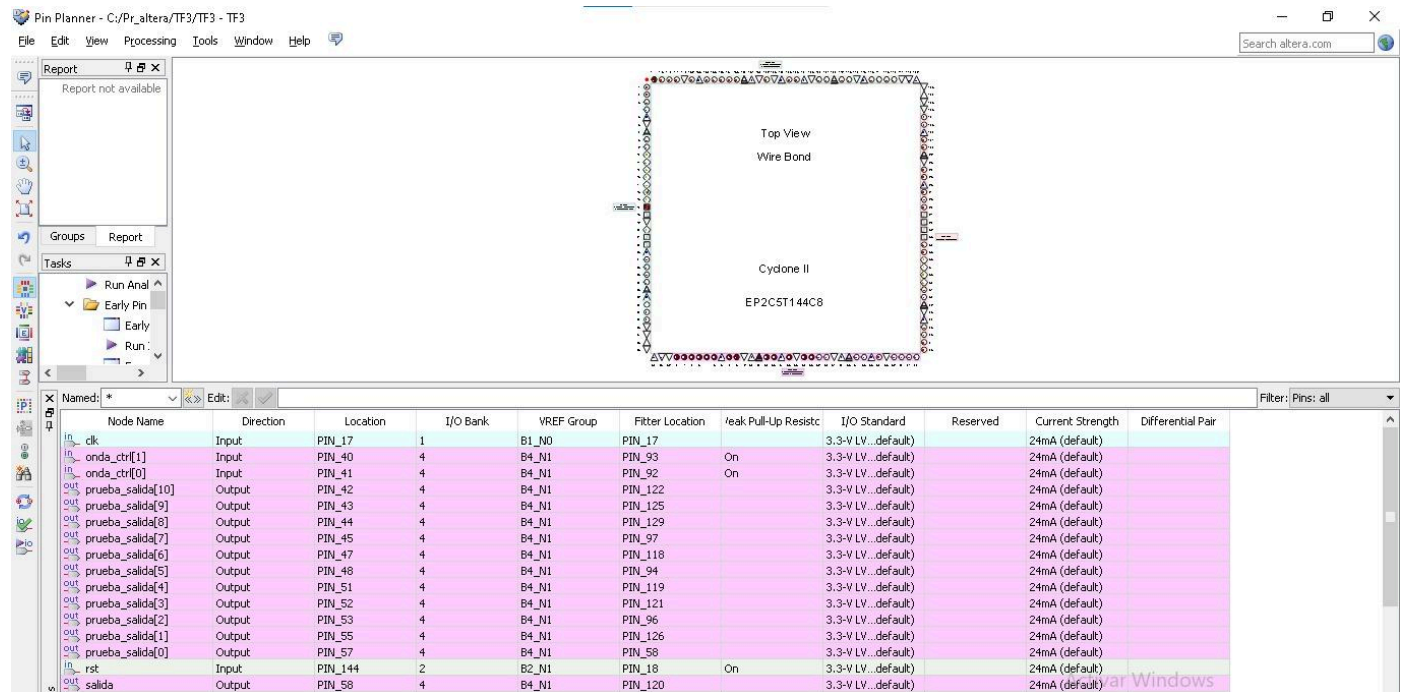


Fig 13. Esquemático del circuito implementado

En la figura 13, se realizó el esquemático del circuito implementado usando un capacitor de 0.1 uf y una resistencia de 10k para realizar el filtro pasa bajas también se etiquetaron los pines que se usarán para los botones y se puso un imagen de un osciloscopio como ejemplo de la ondas que se deben generar.

5.5. Pin Planner empleado:



Node Name	Direction	Location	I/O Bank	VREF Group	Filter Location	Weak Pull-Up Resist	I/O Standard	Reserved	Current Strength	Differential Pair
clk	Input	PIN_17	1	B1_N0	PIN_17	On	3.3-V LV...default		24mA (default)	
onda_ctr[1]	Input	PIN_40	4	B4_N1	PIN_93	On	3.3-V LV...default		24mA (default)	
onda_ctr[0]	Input	PIN_41	4	B4_N1	PIN_92	On	3.3-V LV...default		24mA (default)	
prueba_salida[10]	Output	PIN_42	4	B4_N1	PIN_122		3.3-V LV...default		24mA (default)	
prueba_salida[9]	Output	PIN_43	4	B4_N1	PIN_125		3.3-V LV...default		24mA (default)	
prueba_salida[8]	Output	PIN_44	4	B4_N1	PIN_129		3.3-V LV...default		24mA (default)	
prueba_salida[7]	Output	PIN_45	4	B4_N1	PIN_97		3.3-V LV...default		24mA (default)	
prueba_salida[6]	Output	PIN_47	4	B4_N1	PIN_118		3.3-V LV...default		24mA (default)	
prueba_salida[5]	Output	PIN_48	4	B4_N1	PIN_94		3.3-V LV...default		24mA (default)	
prueba_salida[4]	Output	PIN_51	4	B4_N1	PIN_119		3.3-V LV...default		24mA (default)	
prueba_salida[3]	Output	PIN_52	4	B4_N1	PIN_121		3.3-V LV...default		24mA (default)	
prueba_salida[2]	Output	PIN_53	4	B4_N1	PIN_96		3.3-V LV...default		24mA (default)	
prueba_salida[1]	Output	PIN_55	4	B4_N1	PIN_126		3.3-V LV...default		24mA (default)	
prueba_salida[0]	Output	PIN_57	4	B4_N1	PIN_58		3.3-V LV...default		24mA (default)	
rst	Input	PIN_144	2	B2_N1	PIN_18	On	3.3-V LV...default		24mA (default)	
salida	Output	PIN_58	4	B4_N1	PIN_120		3.3-V LV...default		24mA (default)	

Fig 14. Pin Planner del sistema

En la figura 14. Se implementó resistencias PULL-UP, ya que las entradas son botones a excepción del clk. También, cabe aclarar que estas entradas están en lógica inversa por el PULL-UP. Asimismo, se decidió usar los pines del lado derecho del FPGA por comodidad. También, se usó el pin 17 como clk y el pin 144 (implementado en FPGA) como rst.

5.6. Sistema implementado:

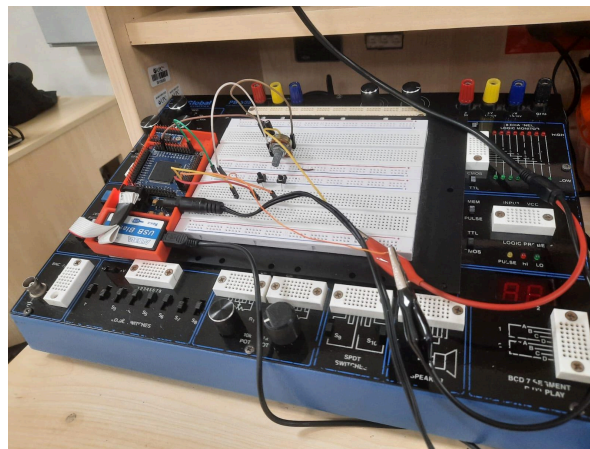


Fig 15. Implementando el Filtro Pasa bajas

En la imagen podemos visualizar cómo se implementa nuestro sistema en físico en protoboard de pruebas donde implementamos los botones y conectamos el FPGA EP2C5T144C8 nuestra laptop.

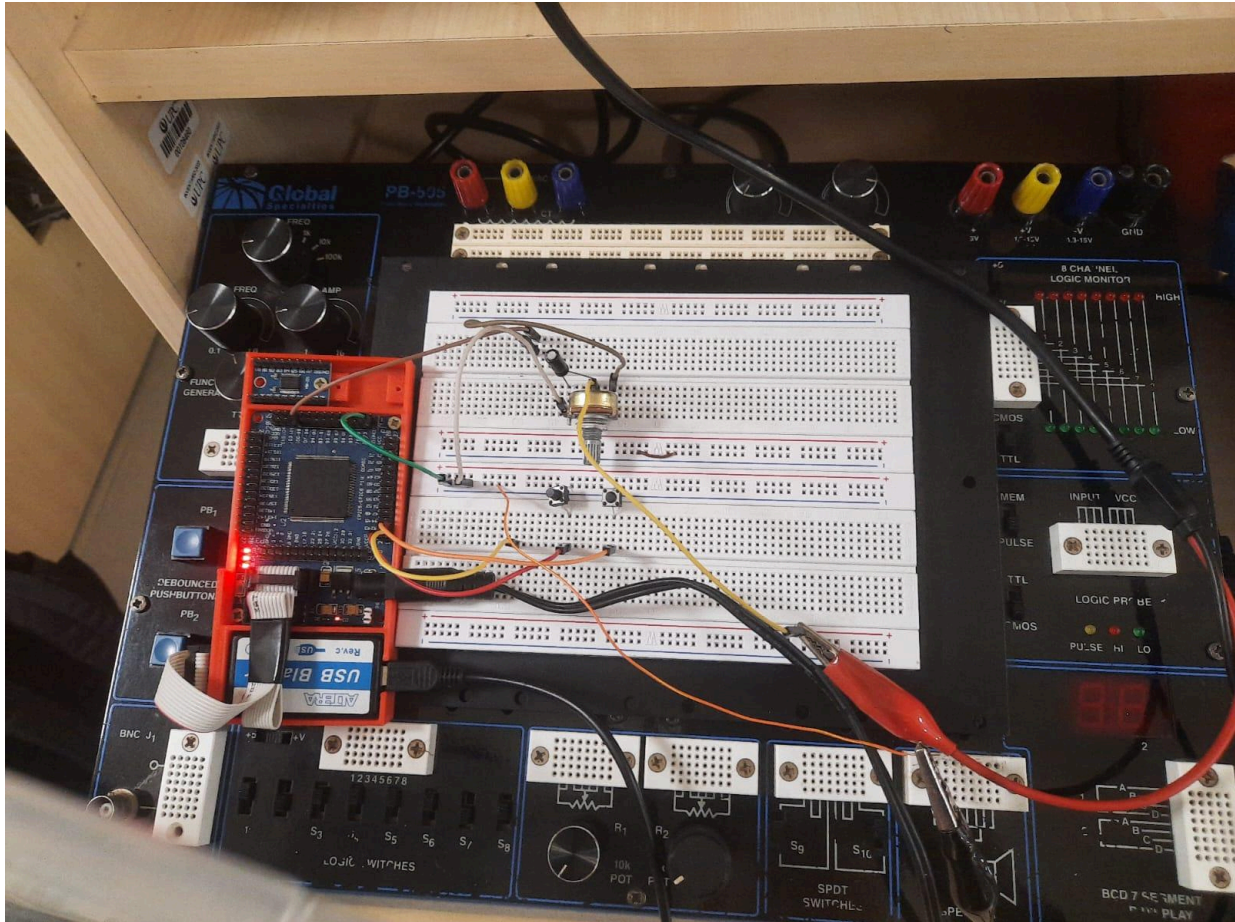


Fig 16. Sistema Implementado

En la imagen podemos visualizar cómo implementamos el filtro pasa bajas con un potenciómetro de 10k y un capacitor electrolítico de 0.1 uf y cómo conectamos los cables de diente de cocodrilo al osciloscopio.

6. Evaluación del funcionamiento del sistema:

6.1. PRUEBA NRO. 1:

En la siguiente imagen se puede observar el funcionamiento del conjunto de botones que controlan la frecuencia y que manipulando estos botones se espera controlar la frecuencia de las ondas generadas. También se implementó el filtro pasa bajos para visualizar el resultado de mi DDS. Se utilizó un osciloscopio con una configuración temporal que me permita ver los períodos de la onda generada.

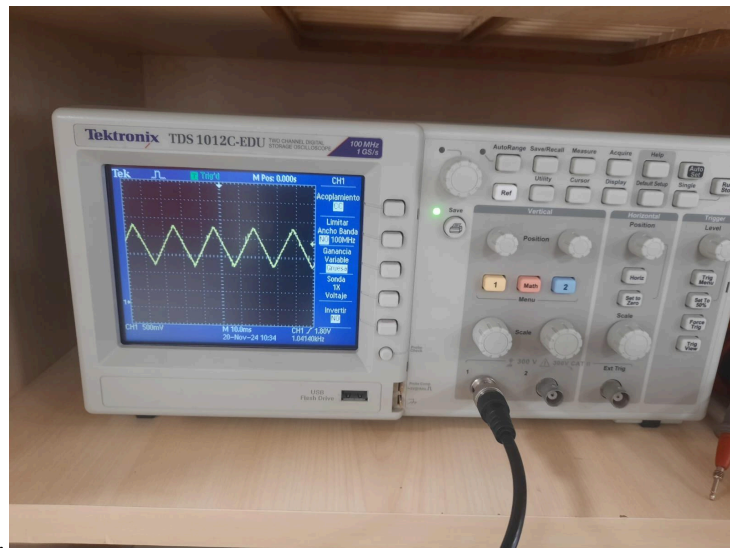


Fig 17. Onda triangular

6.2. PRUEBA NRO. 2:

En la segunda prueba que hicimos se espera que manipulando los conjunto de botones se generen las ondas diente de sierra y la onda senoidal que sí se logró visualizar en el osciloscopio.

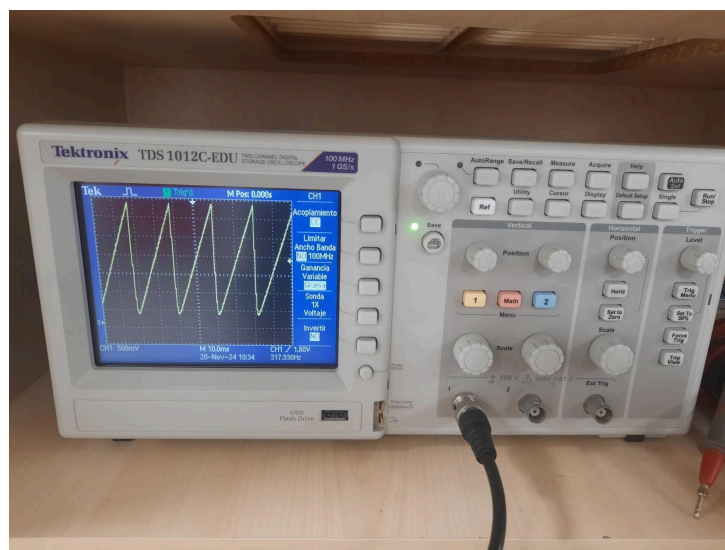


Fig 18. Onda diente de Sierra

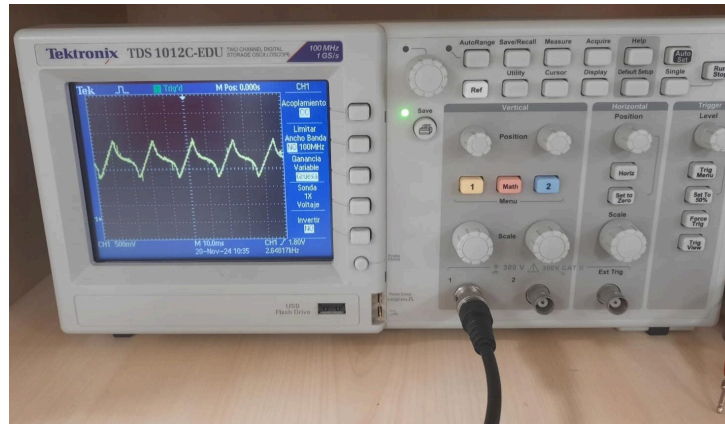


Fig 19. Onda Senoidal

7. Conclusiones:

- Se investigó sobre la técnica DDS mediante diversas fuentes y videos, adquiriendo un entendimiento sólido de su funcionamiento.
- Se implementó un sistema DDS en un FPGA, logrando generar exitosamente ondas senoidales, triangulares y de diente de sierra. Asimismo, se comprobó el funcionamiento de cada componente del sistema.
- Se configuraron los pulsadores para seleccionar la forma de onda mediante el VHDL. No se usó el NIOS II. Asimismo, se pudo garantizar un cambio eficiente entre las formas de onda.
- Se implementó la lógica en VHDL para cambiar entre las diferentes formas de onda según el pulsador, asegurando una respuesta precisa y oportuna.
- Se investigó el módulo PWM y se desarrolló su diagrama de funcionamiento, proporcionando un esquema claro para su integración.
- Se verificó que los comandos se enviaran y recibieran correctamente, asegurando el control de la frecuencia de la señal.
- Se verificaron las formas de onda generadas, cumpliendo completamente con las especificaciones establecidas.

8. Referencias bibliográficas

"¿Qué es un capacitor o condensador eléctrico?," Quartux, [En línea]. Disponible en:
<https://quartux.com/blog/que-es-un-capacitor-o-condensador-electrico/>. [Accedido: 24-Nov-2024].

"¿Qué es un pulsador eléctrico?," Promelsa, [En línea]. Disponible en:
<https://www.promelsa.com.pe/blog/post/que-es-pulsador-electrico.html>. [Accedido: 24-Nov-2024].

"¿Qué es la resistencia?," Fluke Corporation, [En línea]. Disponible en:
<https://www.fluke.com/es-pe/informacion/blog/electrica/que-es-la-resistencia>. [Accedido: 24-Nov-2024].