IMPLEMENTACIÓN EN FPGA DE UN GENERADOR DE TONOS DE RADIOFRECUENCIA

FPGA IMPLEMENTATION OF A RADIO FREQUENCY TONE GENERATOR

Andrés Francisco Cabrera Machaen
Instituto Politécnico Nacional, CITEDI, México
acabrera@citedi.mx

Luis Arturo González Hernández
Instituto Politécnico Nacional, CITEDI, México
lgonzal@citedi.mx

José Cruz Núñez Pérez
Instituto Politécnico Nacional, CITEDI, México
nunez@citedi.mx

Recepción: 4/febrero/2025 **Aceptación:** 7/marzo/2025

Resumen

En este artículo se presenta la implementación de un diseño de radiofrecuencia en una FPGA Artix 7. Se detalla un procedimiento para la creación del sistema de radiofrecuencia utilizando Matlab-Simulink. Se describe la modelización matemática con System Generator for DSP de Xilinx y la simulación de cada componente, verificando así su funcionamiento. Los modelos se compilan en VHDL, para después ser cargados en la FPGA simulada desde el software Vivado Design Suite. Finalmente, se emula el sistema de radiofrecuencia en la tarjeta de evaluación modelo AC701 de Xilinx que contiene el FPGA Artix-7 XC7A200T-2FBG676C. Se desarrolló una interfaz gráfica para la interacción del usuario por medio de botones, barras de desplazamiento y gráficas para la modificación por software del sistema y este sincroniza cualquier frecuencia en un rango determinado por la tarjeta FPGA. Los límites para la generación de frecuencias dependen del reloj de la FPGA, la cual está en los $200\ MHz$.

Palabras Clave: FPGA, Generador, Implementación, Radiofrecuencia, Señal, Tonos, VHDL.

Abstract

This paper presents the implementation of a radio frequency design in an Artix 7 FPGA. A procedure for the creation of the radio frequency system using Matlab-Simulink is detailed. The mathematical modeling with Xilinx System Generator for DSP and the simulation of each component is described, thus verifying its operation. The models are compiled in VHDL, and then loaded into the simulated FPGA from the Vivado Design Suite software. Finally, the RF system is emulated on the Xilinx AC701 evaluation board containing the Artix-7 XC7A200T-2FBG676C FPGA. A graphical interface was developed for user interaction by means of buttons, scrollbars and graphs for software modification of the system and this synchronizes any frequency in a range determined by the FPGA board. The limits for frequency generation depend on the FPGA clock, which is at $200\ MHz$.

Keywords: *FPGA, Generator, Implementation, Radio Frequency, Signal, Tones, VHDL.*

1. Introducción

Los dispositivos FPGAs fueron creados por primera vez en 1984 por el fabricante Xilinx. A lo largo de los años han ido aumentando su complejidad, los circuitos integrados lógicos reemplazan los circuitos integrados dedicados (ASIC) y procesadores personalizados para procesamiento de señales y aplicaciones de control [Lee, 1999]. En muchas aplicaciones se consideran las tarjetas de FPGA con la funcionalidad de un sistema de control MIMO (entrada múltiple y salida múltiple), la entrada control del flujo (FC) y control de temperatura (TC) y al final V1, V2 y V3 representan las salidas.

La mayoría de los trabajos previos encontrados en la literatura se han centrado en aumentar la calidad de los diseños FPGA. Especialmente, en mejorar el flujo CAD para optimizar la arquitectura del dispositivo y en aumentar los recursos lógicos disponibles para los desarrolladores [Gabrick, 2006]. Los avances en estas áreas han permitido a los dispositivos de FPGA la capacidad de soportar aplicaciones sofisticadas, e incluso a Sistemas-en-Chip (SoCs) completos con menor tamaño, operando a una frecuencia más alta y consumiendo menos energía que sus

predecesoras. Sin embargo, la mayor capacidad del hardware ha traído consigo un aumento exponencial de la complejidad y los costos de ingeniería [Jie, 2007].

En muchos casos, la velocidad del lazo del sistema de control está limitada por los módulos de entrada y salida y no por la unidad de procesamiento. Al liberar la restricción de temporización en las herramientas CAD se consume un tiempo menor en la optimización del circuito y, por lo tanto, se generará un flujo de bits de configuración en la tarjeta de FPGA en menor tiempo. El ahorro de tiempo suele ser significativo dado que el diseño de hardware es sintetizado cientos de veces durante su ciclo de diseño, lo que conduciría a un tiempo de comercialización (TTM) más rápido [Jappe, 2010].

El uso de la herramienta Xilinx System Generator for DSP como una herramienta de diseño basada en MATLAB Simulink para el desarrollo de aplicaciones de procesamiento de señales, especialmente en los campos de procesamiento de imágenes y video es de vital importancia para la eficiencia en el diseño y la facilidad de simulación. También permite la generación rápida de prototipos y una transición más fluida desde el diseño algorítmico hasta la implementación en hardware [Yaman, 2019].

La búsqueda e implementación de algoritmos complejos de procesamiento de señales en hardware, particularmente en el campo de las comunicaciones y aplicaciones de separación de fuentes puede ser una tarea complicada. El uso de ZYBO Z7 y Xilinx System Generator ofrece una forma eficiente y flexible de llevar algoritmos desde la teoría hasta la práctica en hardware. [Mekhfioui, 2020].

El aprovechamiento de las capacidades de FPGAs para cumplir con restricciones críticas de tiempo puede ser llevada a cabo a partir del System Generator for DSP. Este permite una ruta de desarrollo más ágil y un proceso de implementación más eficiente comparado con los enfoques tradicionales basados en lenguajes de descripción de hardware [Durgakeri, 2019].

Los sistemas de clasificación de imágenes en tiempo real utilizando FPGAs facilita el proceso de desarrollo y reduce el tiempo necesario para llevar los algoritmos de la simulación al hardware, lo que resulta especialmente útil en aplicaciones que requieren procesamiento rápido y de baja latencia [Lei, 2022].

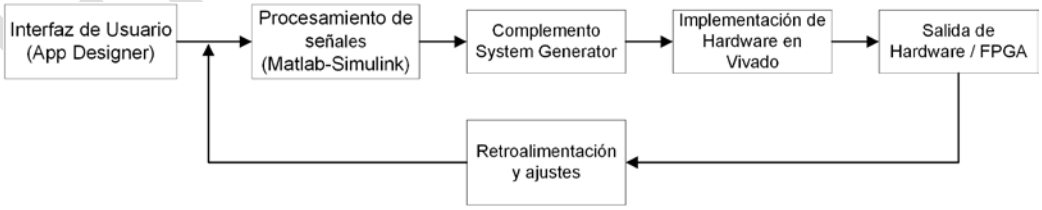
En el campo de la electrónica de potencia y control de convertidores, la combinación de Xilinx System Generator y la plataforma FPGA proporciona una solución poderosa para el desarrollo de sistemas de control de alta velocidad y precisión [Kasim, 2021].

Este articulo describe el desarrollo y la emulación de un sistema de radiofrecuencia, programado en Matlab-Simulink versión R2020b. Posteriormente, el sistema se genera utilizando bloques de System Generator for DSP de Xilinx. Finalmente se realiza la emulación en una tarjeta de FPGA Artix-7 (XC7A200T) de Xilinx en el programa Vivado Design Suite versión 2020.2, replicando digitalmente el comportamiento de un sistema de generación de señales de RF.

2. Métodos

Esta sección incluye los conceptos teóricos relacionados a las FPGA específicamente la Artix 7 de Xilinx con su definición, su uso en la industria y los cambios significativos a lo largo de su evolución. Además, se presenta el lenguaje VHDL, donde se explica la estructura básica en los tipos de datos con los números reales, como es el punto fijo y flotante.

En la Figura 1 se presenta el diagrama de bloques del sistema de radiofrecuencia (RF) que incluye una interfaz de usuario diseñada en App Designer, se interactúa con el sistema para definir parámetros de los tonos de radiofrecuencia. Después, el procesamiento de señales por parte de Matlab Simulink. En este se procesan las señales según las especificaciones dadas por el usuario.



Fuente: elaboración propia

Figura 1 Diagrama de bloques del sistema de RF.

El complemento System Generator convierte los datos de Simulink en un formato adecuado para la síntesis de hardware, generando los archivos necesarios para

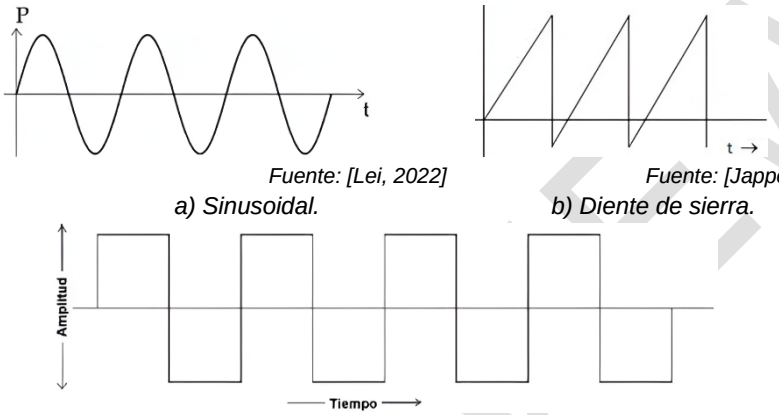
Vivado. En el bloque de implementación de hardware en Vivado se utilizan los valores generados y se programa el FPGA, creando el sistema que generará los tonos de RF en el hardware. En la salida de hardware el sistema en el FPGA está activo, produciendo tonos de RF basados en las entradas y configuraciones iniciales del usuario. Finalmente, se muestra el bloque de retroalimentación y ajustes el cual consiste en los ajustes en tiempo real hacia la interfaz de usuario en App Designer, indicando que el usuario puede hacer ajustes basados en los resultados.

Ondas sinusoidales, triangulares y cuadradas

La forma de onda sinusoidal está presente en varias aplicaciones e incluye características matemáticas armónicas. El voltaje que proviene de un interruptor de pared recorre el circuito siguiendo un patrón sinusoidal. Las señales de prueba que emite el circuito oscilador de un generador de señales son sinusoidales. Aunque el voltaje varía, la mayoría de las fuentes de energía de corriente alterna (CA) generan ondas sinusoidales. Se menciona la onda sinusoidal ya que como se utiliza dentro del sistema de radiofrecuencia, se necesita saber sus características para ser aplicadas en el software. Sin embargo, la onda sinusoidal varía su amplitud en el tiempo, en la Figura 2a se muestra un circuito oscilador, su amplitud puede ser constante o aumentar o disminuir progresivamente.

Las ondas triangulares y de diente de sierra emergen de circuitos concebidos para regular voltajes de manera lineal, como se observa en el barrido horizontal de un osciloscopio analógico o el barrido de trama en un televisor. También en el caso de estas ondas se mencionan los circuitos y los niveles de voltaje para su utilización en la interfaz de usuario con la finalidad de analizar las transiciones y conocer su aplicación en software. Las transiciones entre los distintos niveles de voltaje en estas ondas se efectúan a un ritmo constante; a dichas transiciones se les llama rampas, las cuales se ilustran en la Figura 2b. La onda cuadrada es un tipo de voltaje que se activa y desactiva en periodos regulares, sirviendo como una onda estándar para la evaluación de diversos circuitos, incluyendo los amplificadores. Los amplificadores eficaces son capaces de expandir la amplitud de una onda cuadrada con mínima distorsión [Abu-Rub, 2004]. Para el caso de la onda cuadrada es

relevante mencionar sus aplicaciones para demostrar su uso cotidiano en sistemas que comúnmente están presentes en la vida real. Circuitos de televisión, radio y computadoras frecuentemente emplean ondas cuadradas para sincronizar señales, visualizables en la Figura 2c, la onda rectangular guarda similitud con la cuadrada, pero se diferencian en que los periodos de tiempo alto y bajo no son equitativos, siendo especialmente relevante en el análisis de circuitos digitales.



Fuente: [Lei, 2022] *Fuente: [Jappe, 2010]*

a) Sinusoidal. *b) Diente de sierra.*

Fuente: [Abu-Rub, 2004]

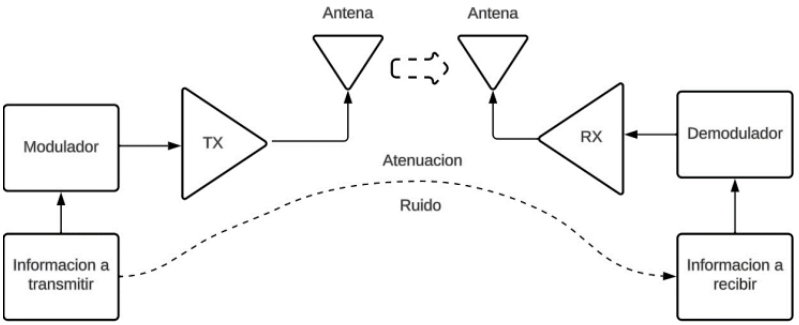
c) Cuadrada.

Figura 2 Tipos de Ondas.

Sistemas de RF

Los sistemas de radiofrecuencia se refieren al uso de ondas electromagnéticas en el rango de radiofrecuencia (generalmente de $20\ kHz$ a $300\ GHz$) para transmitir información (voz, datos, video) entre dos puntos sin un medio físico de conexión.

En la Figura 3, se muestra un ejemplo de un diagrama de bloques de un sistema de radiofrecuencia, el cual está conformado por un transmisor (TX), un receptor (RX) y las dos antenas para establecer el enlace. Tiene como objetivo enviar y/o recibir información de dos lugares remotos. La información será trasmitida por un enlace inalámbrico realizado mediante dos antenas. El enlace atenúa la señal enviada y, además, adiciona ruido a la señal. En este caso se explica el diagrama ya que es esencial conocer cómo funciona el sistema para la transmisión de datos a través de la interfaz gráfica de MATLAB Simulink para guardar la información en el *Workspace* para después ser recibida y mostrada como señales en el software.



Fuente: elaboración propia

Figura 3 Diagrama de bloques de sistema de RF.

FPGA

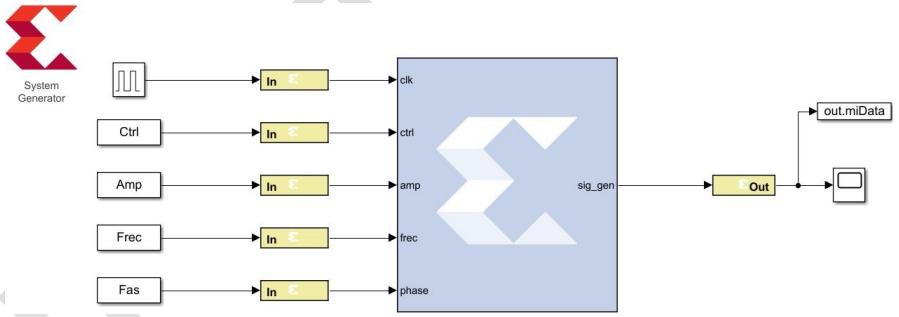
Los FPGA son circuitos integrados de silicio que se reprograman y emplean bloques lógicos preconstruidos y con capacidades de enrutamiento. Para lograr esto, las tareas de computación digital deben crearse en software y compilarse en un archivo de configuración que especifique cómo se deben conectar los componentes [Chin, 2011]. Antes de la aparición de las herramientas de diseño de alto nivel, trabajar con la tecnología FPGA requería un amplio conocimiento en el diseño de hardware digital. Las nuevas tecnologías traducen código ANSI, C, gráficos de bloques y diagramas en circuitos de hardware digital [Chin, 2009]. La tarjeta del kit de evaluación modelo AC701 de Xilinx utiliza el chip XC7A200T-2FBG676C, además contiene diversos los periféricos que se usan para variedad de aplicaciones.

VHDL

Es un lenguaje destinado a la descripción de circuitos electrónicos digitales, opera con distintos grados de abstracción y se utiliza para detallar tanto circuitos síncronos como asíncronos, optimizando así el procedimiento de diseño. Este lenguaje tiene como finalidad principal el modelado, que es la formulación de un modelo para la simulación de un circuito o sistema, y la síntesis, que es la transición desde una especificación de entrada en un nivel de abstracción determinado hacia una representación más detallada y concreta de circuitos y sistemas electrónicos y digitales [Fratta, 2004].

Diagrama de bloques en Simulink

El diagrama de bloques desarrollado en Simulink se presenta en la Figura 4, el Black Box que como se mencionó anteriormente es el que almacena el código VHDL desarrollado en Vivado. En este caso tiene 5 entradas, las cuales son la de reloj (clk), Ctrl que permite manipular el sistema de radiofrecuencia y da la opción a elegir una señal cuadrada, la triangular que sería el diente de sierra y la sinusoidal. También se tienen los bloques To Workspace que contienen las variables del sistema Ctrl, Amp, Frec y Fas, en ellas se almacena el valor obtenido desde la interfaz desarrollada en App Designer de Matlab. Es importante saber que estos deben de estar conectados con los bloques de Gateway In de color amarillo al Black Box para realizar la conexión de manera correcta. De esta manera con el bloque Gateway Out se mostrará con un Scope la señal deseada y utilizando el bloque To Workspace se visualizarán las señales en los ejes de la interfaz. Más adelante se muestran los parámetros de los bloques ya que es de suma importancia entender cuáles son los valores elegidos con la finalidad de tener una simulación exitosa.



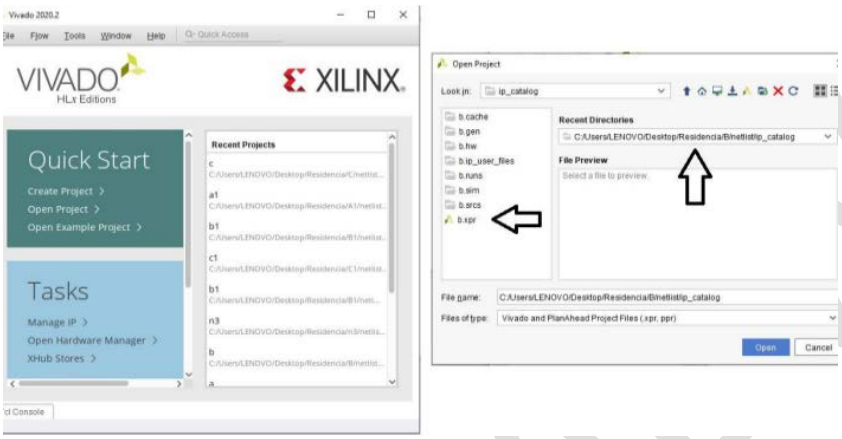
Fuente: elaboración propia

Figura 4 Diagrama de bloques Simulink.

Vivado Xilinx

Es de suma importancia el saber cómo se realiza el diseño del sistema en el software de Vivado, el inicio de este sistema fue una implementación realizada en código VHDL. En la sección Quick Start de la ventana inicial, se selecciona la opción Open Project. Enseguida se abre una ventana de explorador de archivos, se busca el archivo con extensión. xpr (el cual corresponde a un archivo de proyecto en Vivado) en la ruta seleccionada de la PC en el bloque System Generator.

Generalmente el archivo se encuentra en la carpeta netlist en la misma ruta donde se encuentra el archivo del modelo en Simulink y con el mismo nombre. En la Figura 5, se muestran las ventanas de Inicio y Open Project de Vivado.

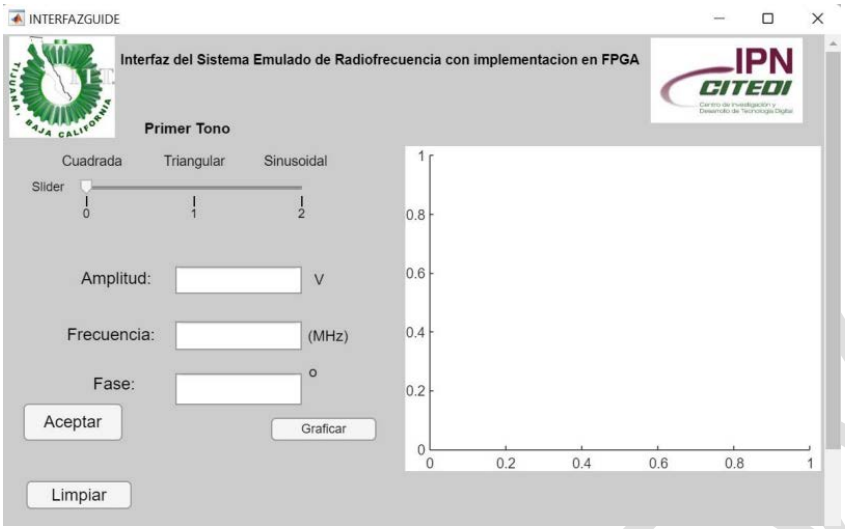


Fuente: elaboración propia

Figura 5 Ventana de Inicio y Open Project de Vivado.

App Designer de Matlab

En la Figura 6, se muestra la interfaz, incluye una etiqueta que señala el primer tono, una barra de desplazamiento con la opción a elegir de una señal cuadrada, una triangular que sería el caso del diente de sierra y la sinusoidal. Además, se tienen los campos utilizando cajas de texto para ingresar la amplitud, frecuencia y fase con sus unidades específicas. Después se coloca el botón aceptar el cual realiza el proceso de mandar las variables ingresadas al espacio de trabajo (workspace) para validarlas. Cuando las cinco variables estén guardadas en el workspace se mandará un mensaje diciendo que ya están listas, de ahí se procede al botón graficar donde al darle clic aparecerá la señal deseada. El mismo componente Axes da opción de posicionar la señal o maximizar para una mejor visualización del gráfico. El ultimo botón llamado limpiar es para borrar todas las selecciones tal como los datos ingresados en las cajas de texto y lo que se muestre en el gráfico. Sin embargo, lo más importante es que se borrarán los datos del espacio de trabajo de Matlab, ya que si no se realiza esa acción podría causar problemas al momento de realizar la ejecución de los procesos de simulación.



Fuente: elaboración propia

Figura 6 Diseño de la interfaz del sistema de radiofrecuencia.

En esta sección se realizó la explicación del desarrollo del sistema generador de tonos de radiofrecuencia comenzando por el software Vivado usando código VHDL para su realización. Después con la ayuda del entorno Matlab-Simulink se crea el diagrama de bloques el cual contiene el código VHDL de Vivado. Con el complemento llamado System Generator for DSP de Xilinx se especifican los parámetros de cada uno de los bloques. Además, fue esencial el haber explicado cada uno de ellos ya que son importantes para saber cómo está funcionando el sistema y si se llega a cambiar alguno de estos se modifica completamente su funcionamiento. Por último, se muestra el desarrollo de la interfaz con una introducción al entorno de App Designer y su importante uso para darle una correcta presentación a una aplicación. El desarrollo de este sistema es indispensable para demostrar cómo se pueden lograr ciertas simulaciones de sistemas digitales de una manera optimizada para ahorrar tiempo en pruebas y validación de procesos en la industria.

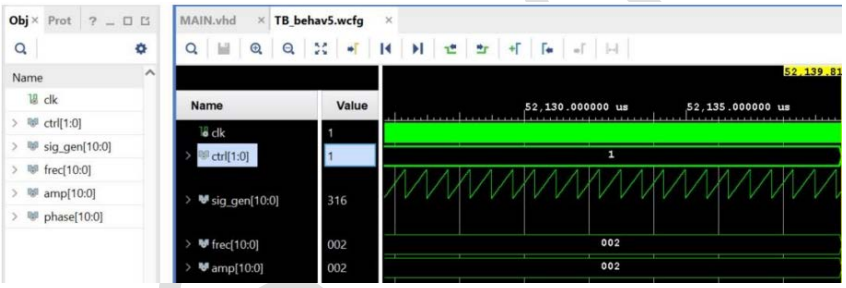
3. Resultados

En esta sección se presentan los resultados de las simulaciones del sistema generador de tonos de radiofrecuencia. Se comienza mostrando las señales resultantes en la ventana de la interfaz de Vivado, para después obtener el proyecto

en VHDL y emularlo en la tarjeta de FPGA Artix7 AC701 de Xilinx con la ayuda del entorno Matlab Simulink y la herramienta System Generator for DSP de Xilinx.

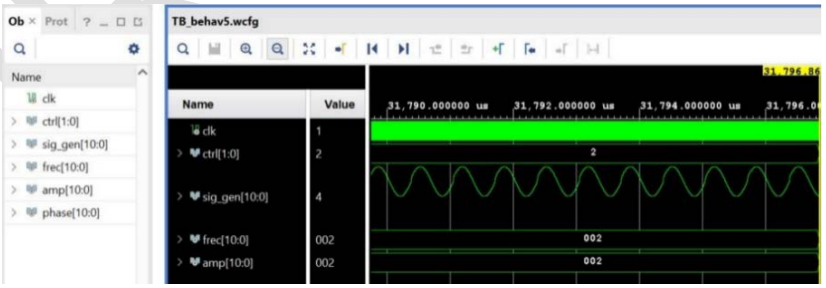
Simulación en Vivado Xilinx

La Figura 7 muestra la simulación de la onda diente de sierra que es referida como triangular, como se dijo anteriormente el parámetro de Control se cambia a "1" para que muestre esa señal. En la pestaña de objetos se muestran las variables que conforman al sistema con su número de bits. La frecuencia, fase y amplitud tiene un valor de 11 bits al igual que su salida para respetar el tipo de dato y reloj (CLK). La Figura 8 muestra la simulación de la señal sinusoidal la cual se selecciona poniendo el parámetro de Control con valor "2" para especificar su visualización en el waveform de Vivado.



Fuente: elaboración propia

Figura 7 Simulación de la señal de la onda diente de sierra en Vivado.



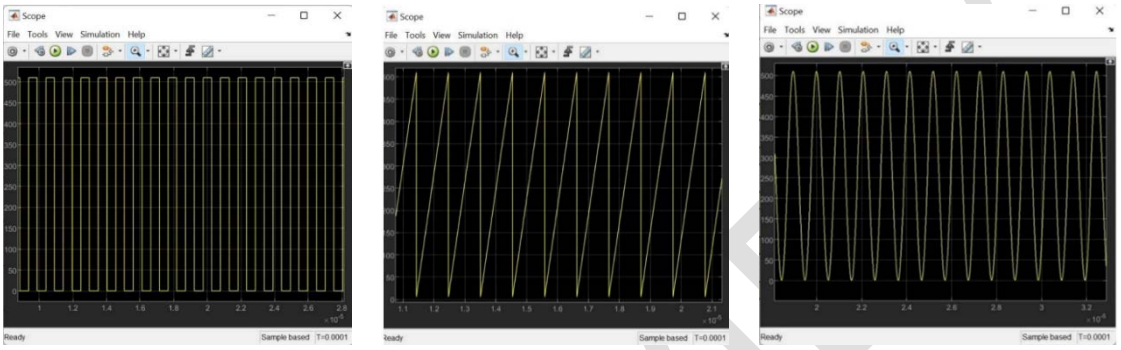
Fuente: elaboración propia

Figura 8 Simulación de la señal de la onda sinusoidal en Vivado.

Simulación en el entorno Matlab Simulink

En la Figura 9a se muestra la señal cuadrada seleccionada del sistema ya que se ingresó el valor 0 que está configurado para que en el osciloscopio aparezca. La

Figura 9b muestra la señal diente de sierra o triangular, seleccionada en el sistema con el valor de Control “1” que está configurado para que en el osciloscopio aparezca. En la Figura 9c se muestra la señal sinusoidal seleccionada del sistema ya que se ingresó el valor de Control “2” que está configurado para que en el osciloscopio aparezca.



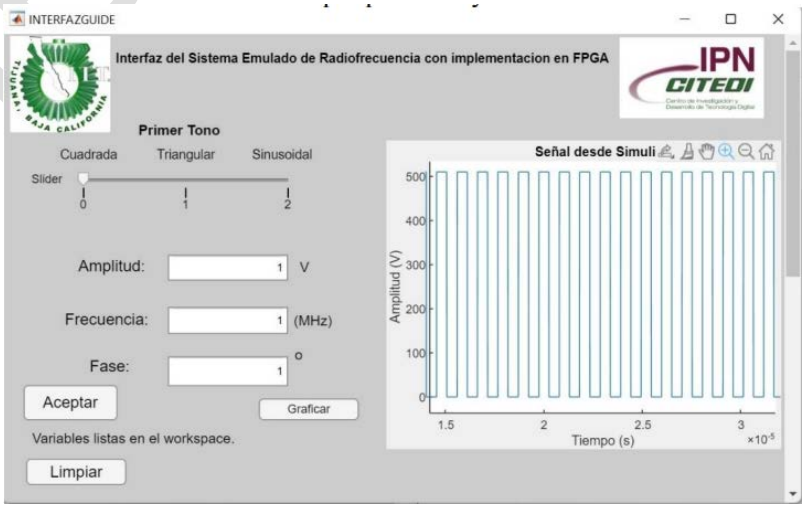
a) onda cuadrada. *b) onda diente de sierra.* *c) onda sinusoidal.*

Fuente: elaboración propia

Figura 9 Visualización de ondas en Simulink.

Interfaz gráfica de usuario

En la Figura 10 se muestra la interfaz desarrollada en App Designer la cual ya tiene valores ingresados en amplitud, frecuencia y fase, también la selección de la onda cuadrada.

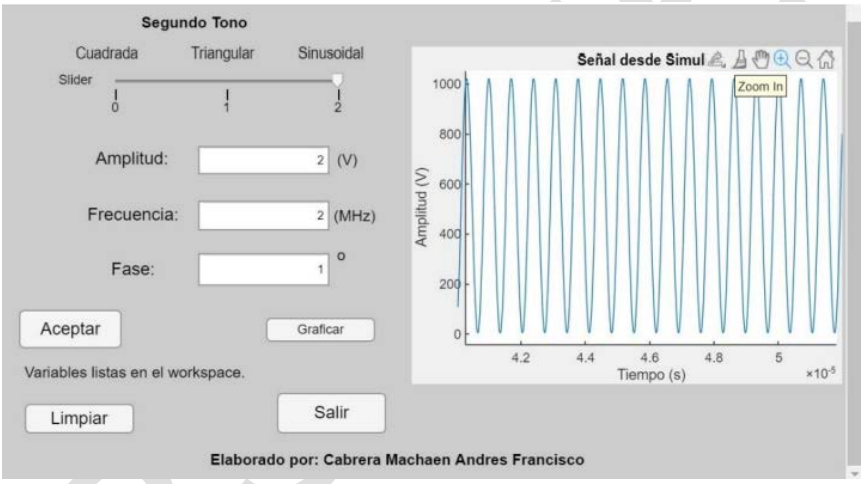


Fuente: elaboración propia

Figura 10 Interfaz con la señal cuadrada.

Al momento de darle clic al botón aceptar y esperar unos momentos aparece el mensaje de que las variables están listas en el espacio de trabajo de Matlab por lo que en el gráfico aparece la señal cuadrada en términos de amplitud y tiempo. Por lo tanto, se cumple con su función y comprobación de emulación de sistema por parte de System Generator for DSP de Xilinx.

La Figura 11 se muestra la ejecución de la simulación en la interfaz gráfica donde se ingresan los parámetros para configurar las señales de salida. Con el mismo mensaje las variables se guardan en el espacio de trabajo de Matlab. Como se mencionó anteriormente, en la interfaz tiene dos partes, para generar dos tonos de radiofrecuencia.



Fuente: elaboración propia

Figura 11 Interfaz con la señal sinusoidal.

4. Discusión

Se demostró el funcionamiento correcto del sistema generador de tonos de radiofrecuencia mediante la emulación de una tarjeta FPGA Artix 7 de Xilinx con la ayuda del software Vivado. El diseño del diagrama de bloques por parte del entorno Matlab Simulink y la herramienta System Generator for DSP con su compilación fue verificada como un proyecto de lenguaje VHDL.

La comparación de trabajos realizados con la herramienta System Generator for DSP es indispensable para conocer todas las ventajas que este complemento ofrece. Como se mencionó está presente en los campos de procesamiento de

imágenes y video donde se resalta la eficiencia en el diseño y facilidad de simulación, es flexible al momento de llevar algoritmos desde la teoría hasta la práctica en hardware. También puede ser enfocado en optimizaciones que permiten a los sistemas digitales operar dentro de los límites de tiempo crítico, es útil para aplicaciones que requieren procesamiento rápido y de baja latencia. Y por último proporciona una gran solución para el desarrollo de sistemas de control de alta velocidad y precisión por lo que con la implementación de este generador de tonos demuestra otro de los casos en que la emulación de una FPGA es una gran opción para solucionar estos problemas.

La creación de la interfaz en App Designer fue muy importante ya que es la manera más eficiente de controlar el sistema y no tener que estar abriendo varias pantallas al mismo tiempo, con la finalidad de ahorrar tiempo y optimizar procesos. Las simulaciones tienden a ser más rápidas porque Matlab Simulink está optimizado para realizarlas. La realización de un diseño en un FPGA real implica varios pasos, incluyendo la síntesis del diseño y la configuración. En general, implementar un diseño en un FPGA real suele ser más lento que simplemente realizarlo en el System Generator for DSP de Xilinx. Este logra la co-simulación con otros sistemas y componentes modelados en Simulink, por lo que se logra identificar y solucionar problemas en el diseño con las herramientas de depuración. En la Tabla 1 se presenta una comparación del tiempo estimado que toma la ejecución del sistema desde la interfaz y la implementación si se realizase físicamente para cuestión de análisis y pruebas.

Tabla 1 Comparación de tiempo estimado entre emulación y aplicación física.

FPGA Artix-7 xc7a200t-2 (Física)	Tiempo (Estimado)	FPGA Artix-7 xc7a200t-2 (Emulada)	Tiempo utilizado
Ejecución del sistema	15 minutos	Ejecución del sistema	22 segundos

Fuente: elaboración propia

5. Conclusiones

En este trabajo se llevó a cabo la emulación de un sistema generador de tonos de radiofrecuencia en una tarjeta FPGA Artix 7 de Xilinx. Se encontró que el sistema implementado presenta resultados eficientes con su diseño en Simulink gracias al

diseño en Vivado y a la interfaz gráfica en App Designer. Al ejecutar la simulación se tarda 22 segundos en cargar todo el sistema, en comparación de los 15 minutos que tomaría la implementación física del sistema usando la tarjeta. Por lo que para empezar un sistema que necesite una tarjeta de FPGA es una excelente opción para realizar pruebas y ajustes sistemáticos. La emulación en Simulink con System Generator es una herramienta valiosa para el desarrollo inicial, la validación del diseño y la depuración, pero si se llegara a necesitar una representación de rendimiento mucho más precisa de la funcionalidad del sistema, se podría intentar implementar y probar el diseño en una FPGA real. Ambos enfoques son complementarios en el flujo del diseño de sistemas basados en FPGA, por lo que se requiere buscar la mejor opción en cuanto a las necesidades del sistema y de los recursos que se tiene en el momento.

6. Bibliografía y Referencias

[1] Abu-Rub H, Guzinski J, Krzeminski Z, Toliyat HA., Predictive current control of voltage-source inverters. IEEE Transactions on Industrial Electronics pp.585–593, 2004.

[2] Chin S, Wilton S., Static and Dynamic Memory Footprint Reduction for FPGA Routing Algorithms. ACM Transactions on Reconfigurable Technology and Systems (TRETS) pp.1-20, 2009.

[3] Chin S, Wilton S., Towards Scalable FPGA CAD through Architecture. ACM International Symposium on Field-Programmable Gate Arrays, Vancouver, BC, Canada, Feb, pp 150-174, 2011.

[4] Durgakeri B. S, Chiranjeevi G. N. Implementing Image Processing Algorithms using Xilinx System Generator with Real Time Constraints. 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), Bangalore, India, 17-18 Mayo 2019, pp. 230-234.

[5] Fratta A, Griffero G, Nieddu S., Comparative analysis among DSP and FPGA based control capabilities in PWM power converters. In: Proc. of Industrial Electronics Society, vol. 1, Atlanta, GA, 19-21 December, pp. 257–262, 2004.

[6] Gabrick M, Nicholson R, Winters F, Young B, Patton J., FPGA Considerations for Automotive Applications. In: Proc. SAE World Congress and Exhibition, Detroit, MI, 18-20 June, 2006.

[7] Jappe TK, Mussa SA, Rosendo RHS., Synchronous State Machine Inner FPGA Control PFC Boost Converter. IEEE International Symposium on Industrial Electronics, Memphis, TN, 13-15 Mayo, pp 1097–1102, 2010.

[8] Jie Q, Stroud CE, Dai FF., FPGA-Based Analog Functional measurements for Adaptive Control in Mixed-Signal Systems. IEEE Transactions on Industrial Electronics pp.1885-1897, 2007.

[9] Kasim M. Q, Hassan R. F, Humaidi A. J, Abdulkareem A. I, Nasser A. R, Alkhayyat A. Control Algorithm of Five-Level Asymmetric Stacked Converter Based on Xilinx System Generator. 2021 IEEE 9th Conference on Systems, Process and Control (ICSPC 2021), Malacca, Malaysia, 10-11 Diciembre 2021, pp. 174-179.

[10] Lee D, Choi A, Koo J, Lee J, Kim B., A Wideband DS-CDMA Modem for a Mobile Station. IEEE Transactions on Consumer Electronics, pp.1259-1269, 1999.

[11] Lei Z, Fang Y, Li C, Cui J, Zhang Y. Coin Classification Using Xilinx System Generator. 2022 7th International Conference on Intelligent Computing and Signal Processing (ICSP), Xi'an, China, 15-17 Abril 2022, pp. 553-557.

[12] Mekhfioui M, Satif A, Mouhib O, Elgouri R, Hadjoudja A, Hlou L. Hardware Implementation of Blind Source Separation Algorithm Using ZYBO Z7 & Xilinx System Generator. 2020 5th International Conference on Renewable Energies for Developing Countries (REDEC), Marrakech, Marruecos, 29-30 Junio 2020, pp. 1-5.

[13] Yaman S, Yildrim M, Karmşhoğlu B, Erol Y, Kürüm H. Image and Video Processing Applications Using Xilinx System Generator. 2019 7th International Symposium on Digital Forensics and Security (ISDFS), Barcelos, Portugal, 10-12 Junio 2019, pp. 1-5.