

EMULACIÓN EN FPGA DE LA CAPA FÍSICA DE UN TRANSECTOR IEEE 802.11N EN MODO DE ALTO DESEMPEÑO

*FPGA EMULATION OF THE PHY LAYER OF AN IEEE 802.11N
TRANSRECEIVER IN HIGH-PERFORMANCE MODE*

Fernando Evier López Pérez

Instituto Politécnico Nacional, México
flopezp@citedi.mx

José Cruz Núñez Pérez

Instituto Politécnico Nacional, México
nunez@citedi.mx

Recepción: 13/noviembre/2024

Aceptación: 27/diciembre/2024

Resumen

Este artículo tiene como objetivo evaluar la funcionalidad de la capa física, denominada PHY, del 802.11n, utilizando una tarjeta de FPGA. Para lo anterior, se reconstruyen los procesos de transmisión y recepción por medio de las técnicas requeridas por el estándar, desde la convolución binaria, la modulación de amplitud por cuadratura QAM, el intercalado de datos y la multiplexación por división de frecuencias ortogonales OFDM. El estándar IEEE 802.11n ha revolucionado las comunicaciones inalámbricas al mejorar significativamente las capacidades de las redes de área local inalámbricas. En este trabajo se plantea una visión general de la capa física del estándar 802.11n, centrándose en sus avances y los beneficios que aporta a las comunicaciones inalámbricas.

Palabras Claves: FPGA, IEEE 802.11n, OFDM, QAM, VHDL.

Abstract

This article aims to evaluate the functionality of the physical layer, known as PHY, of the 802.11n standard using an FPGA board. To achieve this, the transmission and reception processes are reconstructed using the techniques required by the standard, including binary convolution, quadrature amplitude modulation (QAM),

data interleaving, and orthogonal frequency division multiplexing (OFDM). The IEEE 802.11n standard has revolutionized wireless communications by significantly enhancing the capabilities of wireless local area networks. This work provides an overview of the physical layer of the 802.11n standard, focusing on its advancements and the benefits it brings to wireless communications.

Keywords: *FPGA, IEEE 802.11n, OFDM, QAM, VHDL.*

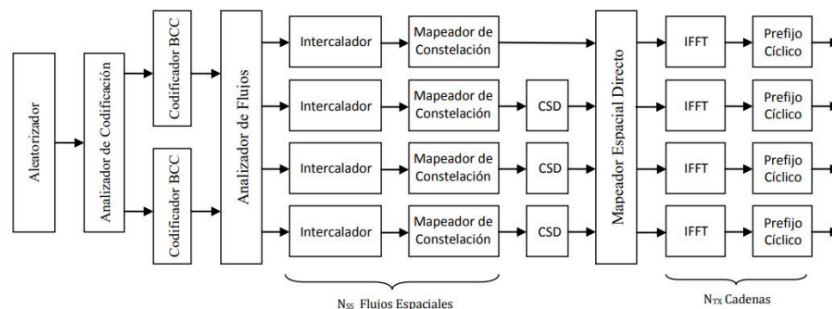
1. Introducción

En las últimas décadas se han presentado grandes avances tecnológicos impulsados tanto por la globalización como por el crecimiento poblacional. Estos mismos factores son el origen de las necesidades de intercambio de información y de comunicación que existen en la actualidad. Una de las principales herramientas que permiten satisfacer dichas necesidades es el internet, sin embargo, el acceso a este depende completamente de las redes disponibles para las personas. En un principio solo se contaba con redes informáticas con acceso a internet que hacían uso de cableado o un medio físico para establecer una conexión por Ethernet el cual se basa en el estándar IEEE 802.3. Este tipo de redes cuentan con sus ventajas, sin embargo, presentan problemas como son el costo alto de instalación, dificultad para agregar a nuevos usuarios o dispositivos a la red, y el hecho de que el punto de acceso es fijo. Como solución a estos problemas en 1990 se comenzó a desarrollar un nuevo tipo de red basada en redes inalámbricas que hacían uso de radiofrecuencia (RF) para establecer una conexión entre un dispositivo y otro. Este tipo de tecnología fue denominada red de acceso local inalámbrica (WLAN, por sus siglas en inglés) y se basa en el estándar IEEE 802.11 [IEEE, 2020].

El estándar IEEE 802.11 se introdujo por primera vez en 1997 con una velocidad de transmisión de hasta 2 *Mbps*. En ese momento, se trataba de una tecnología revolucionaria que ofrecía una alternativa conveniente a las redes cableadas. Sin embargo, la demanda de mayores velocidades y capacidades llevó a la necesidad de enmiendas y mejoras en el estándar. En 2007 surge el estándar 802.11n, el cual incrementa las velocidades de transmisión y la cobertura. Esta versión del estándar es capaz de alcanzar velocidades de hasta 600 *Mbps* con anchos de banda de 20 o

40 MHz, y presenta la opción de operar tanto en 2.4 como en 5 GHz. Además de contar con un uso más eficiente del ancho de banda disponible que sus predecesores [López, 2010]. La principal tecnología que permitió mejorar las tasas de transferencia fue MIMO (Múltiples Entradas Múltiples Salidas, por sus siglas en inglés), tecnología que permite transmitir y recibir información por hasta 4 antenas aplicando multiplexación por división espacial, transmitiendo múltiples flujos de datos en un mismo canal [Tektronix, 2011].

En este artículo se presenta la emulación de la capa física de un transceptor basado en el estándar IEEE 802.11n en el modo de alto rendimiento, mejor conocido como modo Greenfield utilizando una tarjeta de FPGA. En el modo de alto desempeño se asume que no se requiere de una compatibilidad con los anteriores estándares 802.11, permitiendo un mayor rendimiento y velocidad de transferencia de información. El sistema por emular en FPGA debe seguir el proceso del diagrama de bloques de un sistema transmisor de datos de telecomunicaciones a nivel de capa física mostrado en la Figura 1.



Fuente: [Sendra, 2011]

Figura 1 Diagrama de bloques del transmisor.

Los datos de entrada pasan por un aleatorizador y un codificador con el fin de asegurar la información a transmitir, esto se realiza a través de un codificador convolucional y se le pueden aplicar hasta cuatro tasas de codificaciones distintas, dependientes de la transferencia de datos deseada. Posteriormente los datos ya codificados pasan por un analizador encargado de ordenar los flujos espaciales, estos se someten a un intercalador que tiene la función de cambiar el orden de salida de los datos con el fin de evitar errores causados por ruido. Después, se

aplica una modulación para aplicar el mapeo en constelaciones de los datos, proceso por el cual se convierte en números complejos el flujo serial de bits. Posteriormente, se da el mapeo en flujos espaciales y la transmisión de los datos usando OFDM. El bloque de prefijo cíclico recupera los datos sin los efectos del canal. Cabe recalcar que la estructura del receptor es análoga al transmisor, realizando los procesos contrarios para la obtención de los datos transmitidos [Espinosa, 2011], [Bach, 2009].

Posterior a la emulación en un FPGA es necesario comprobar la funcionalidad del sistema por medio de una serie de pruebas de evaluación de cada bloque del transceptor, verificar la integridad de los datos transferidos/recibidos, transmitir una imagen y reconstruirla por medio de simulación.

El estándar 802.11n no es una novedad, sin embargo, es el más usado debido a las limitaciones en hardware de la mayoría de los equipos residenciales e industriales. Aún existen muchos dispositivos en circulación sin el hardware necesario para usar los estándares más nuevos como son 802.11ac y 802.11ax. Sin embargo, la mayoría de estos dispositivos si tienen la capacidad de operar el estándar 802.11n permitiendo usar y en este caso ignorar los estándares anteriores para maximizar el rendimiento de la red [Sendra, 2011].

La implementación en FPGA de este sistema capaz de replicar la capa física del estándar 802.11n en modo de alto desempeño antes de llevarla a una implementación a gran escala es esencial para garantizar que el diseño cumpla con las especificaciones y requisitos del estándar. Esto evita errores costosos y retrasos en el proceso de desarrollo al igual que permite explorar técnicas avanzadas de procesamiento de señales, algoritmos de cancelación de interferencias, corrección de errores y técnicas de mitigación de ruido para mejorar el rendimiento y la calidad de las redes inalámbricas.

2. Métodos

El estándar 802.11 describe las funciones, servicios y características que deben tener las redes LAN inalámbricas como son las funciones y requisitos requeridos por un dispositivo para operar dentro de redes independientes, personales e

infraestructurales. Además, define los procedimientos de señalización en la capa física (PHY) y de control de acceso al medio (MAC) necesarios para la correcta operación de la red [IEEE, 2020].

El estándar IEEE 802.11n, también conocido como Wi-Fi 4, es una enmienda significativa al estándar IEEE 802.11 lanzada en 2009. Se diseñó para mejorar la velocidad, alcance y fiabilidad de las redes inalámbricas, superando las limitaciones de los estándares anteriores, como 802.11a, 802.11b y 802.11g. Introdujo la tecnología MIMO, que permite el uso de múltiples antenas tanto en el punto de acceso como en los dispositivos cliente [Gast, 2005], [O'Hara, 2012]. Esto significa que pueden enviarse y recibirse múltiples flujos de datos simultáneamente a través de diferentes antenas. Con MIMO, se pueden utilizar técnicas de procesamiento de señales para separar y combinar los flujos de datos, lo que resulta en una mejora significativa en la velocidad y la capacidad de la red logrando velocidades teóricas de hasta 600 *Mbps* en la banda de 2.4 *GHz* y hasta 1300 *Mbps* en la banda de 5 *GHz*. Además, se mitigan los efectos de interferencia y las pérdidas de señal causadas por obstáculos físicos, lo que se traduce en una mayor estabilidad de la conexión. También ofrece un mayor alcance en comparación con los estándares anteriores. Esto se logra por medio de beamforming, tecnología que por medio de antenas dirige la señal inalámbrica hacia los dispositivos cliente de manera más precisa y eficiente. Otra característica importante del estándar 802.11n es su compatibilidad con los estándares anteriores. Los dispositivos compatibles con 802.11n pueden conectarse a redes basadas en los estándares 802.11a, 802.11b y 802.11g. Sin embargo, para aprovechar las mejoras del estándar 802.11n, todos los dispositivos en la red deben ser compatibles con este estándar debido a la existencia del modo de alto rendimiento o modo Greenfield, al operar exclusivamente en esta tecnología [Espinosa, 2011].

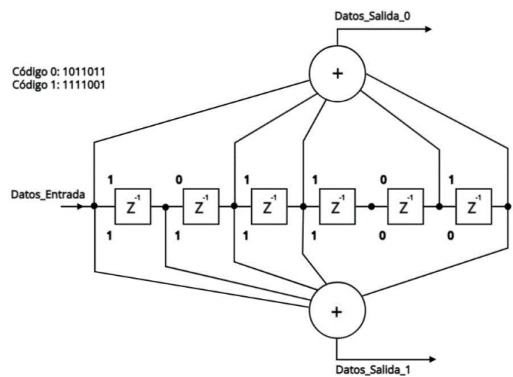
El estándar 802.11n es capaz de manejar 3 formatos de transmisión distintos:

- Non-HT, este formato está destinado para la comunicación de protocolos que no son de alto rendimientos.
- HT-GF, este formato no contiene compatibilidad con formatos que no son de alto rendimiento pues excluye los paquetes de datos compatibles con estos.

- HT-MF, este es un formato mixto, contiene las señales y cabeceras para establecer comunicación con formatos de cualquier tipo, ya sean de alto rendimiento o no.

El desarrollo del sistema se realizó en VHDL para el cual se utiliza una señal de reloj de 100 MHz, en conjunto con dos señales de reinicio, una que se mantiene en alto para los IP de convolución binaria y decodificación Viterbi, mientras que la otra señal se mantiene en bajo y se utiliza por el resto de los bloques del sistema, por último, se tiene la señal de entrada de datos la cual consta de un arreglo de 52 bits.

Basándose en la Figura 1 se tiene que la primera etapa en transmisión es un bloque de convolución binaria correspondiente al de la Figura 2.

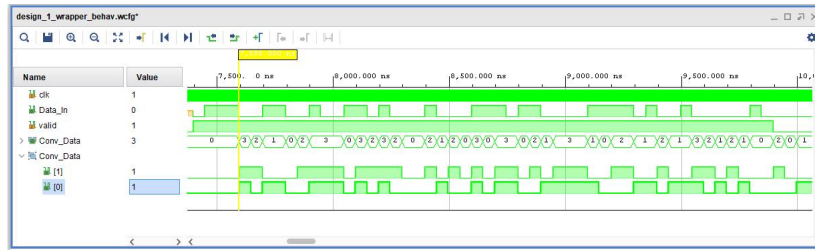


Fuente: elaboración propia

Figura 2 Diagrama de convolucionador binario.

El bloque de convolución binaria se implementó por medio del IP de Vivado, con códigos de convolución en formato octal de 171 y 133 con una longitud de restricción de 7 y una tasa de codificación de $\frac{1}{2}$ de acuerdo con la Figura 2.

La Figura 3 presenta la simulación, donde la entrada denominada Data_In tiene un tamaño de un bit por el cual se suministra una secuencia de 52 datos a lo largo del tiempo. A esta secuencia se aplica la convolución binaria y se obtiene la señal Conv_Data, que representa un vector de 8 bits del cual los 2 menos significativos almacenan el resultado de aplicar la convolución a cada uno de los bits, obteniendo una secuencia de 52 datos de 2 bits, es decir 104 datos totales.



Fuente: elaboración propia

Figura 3 Simulación de bloque de convolución binaria.

Posterior al bloque de convolución binaria se tiene un bloque de intercalado de datos en el cual los datos de entrada se dividen en bloques más pequeños para facilitar el procesamiento. Los bits se reorganizan y se distribuyen en un patrón predefinido de acuerdo con un algoritmo de intercalado, el objetivo es dispersar los *bits* en diferentes posiciones para evitar ráfagas de errores consecutivos en la transmisión. La Figura 4 presenta la simulación del bloque, el resultado se presenta en dos instancias de tiempo distintas debido a que al bloque se le suministran los 104 datos del convolucionador binario en paquetes de 52 *bits*.



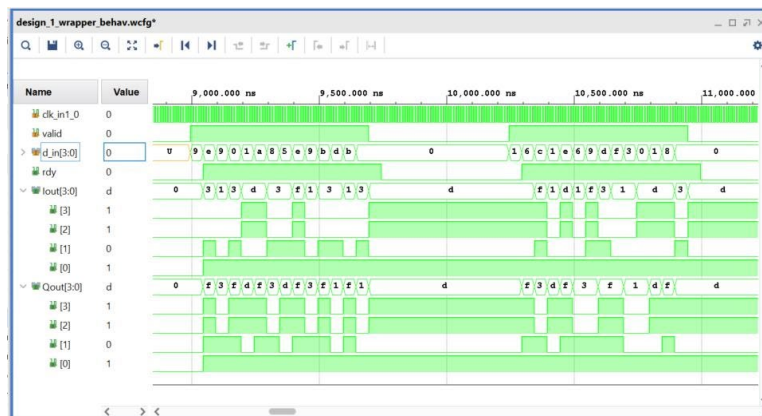
Fuente: elaboración propia

Figura 4 Simulación de bloque intercalador.

La señal de entrada del bloque se denominó Conv_Out, y aparenta cambiar a lo largo del tiempo debido al reajuste de los datos de salida del convolucionador. Una vez que se tiene un paquete de 52 *bits* se pasan por el intercalador utilizando la señal de validez para realizar un handshake con dicho bloque. Este proceso se observa claramente ya que una vez la señal validez tiene un valor de '1' la señal de salida Intlvr_Data toma el valor de los 52 *bits* ya intercalados. Una vez concluido el intercalado de los datos se procede a aplicar la modulación QAM, sin embargo, previo a la modulación QAM es necesario dividir en bloques de 4 *bits* los datos. La

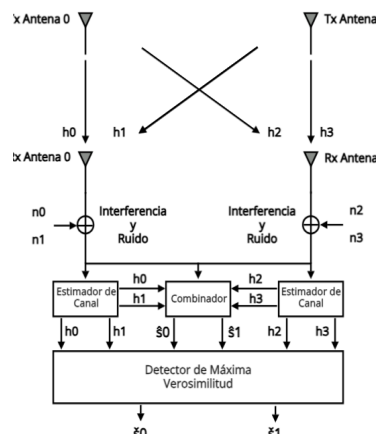
Figura 5 muestra el resultado de la modulación y la segmentación, es posible apreciar que a partir de la señal de entrada se generaron dos señales de salida $lout$ y $Qout$, correspondientes a cada una de las componentes de la modulación QAM, cada señal de salida contiene un total de 13 vectores de 4 bits.

Posterior a la modulación es necesario aplicar MIMO, para esto se hará uso del esquema Alamouti. En este esquema se propone un arreglo de antenas para obtener una multiplexación en los datos que se desean transmitir en una red. El esquema Alamouti utiliza dos antenas de transmisión independientemente de la cantidad de antenas de recepción. En la Figura 6 se presenta el diagrama de los componentes que forman parte del esquema Alamouti y de los bloques requeridos para realizar la decodificación.



Fuente: elaboración propia

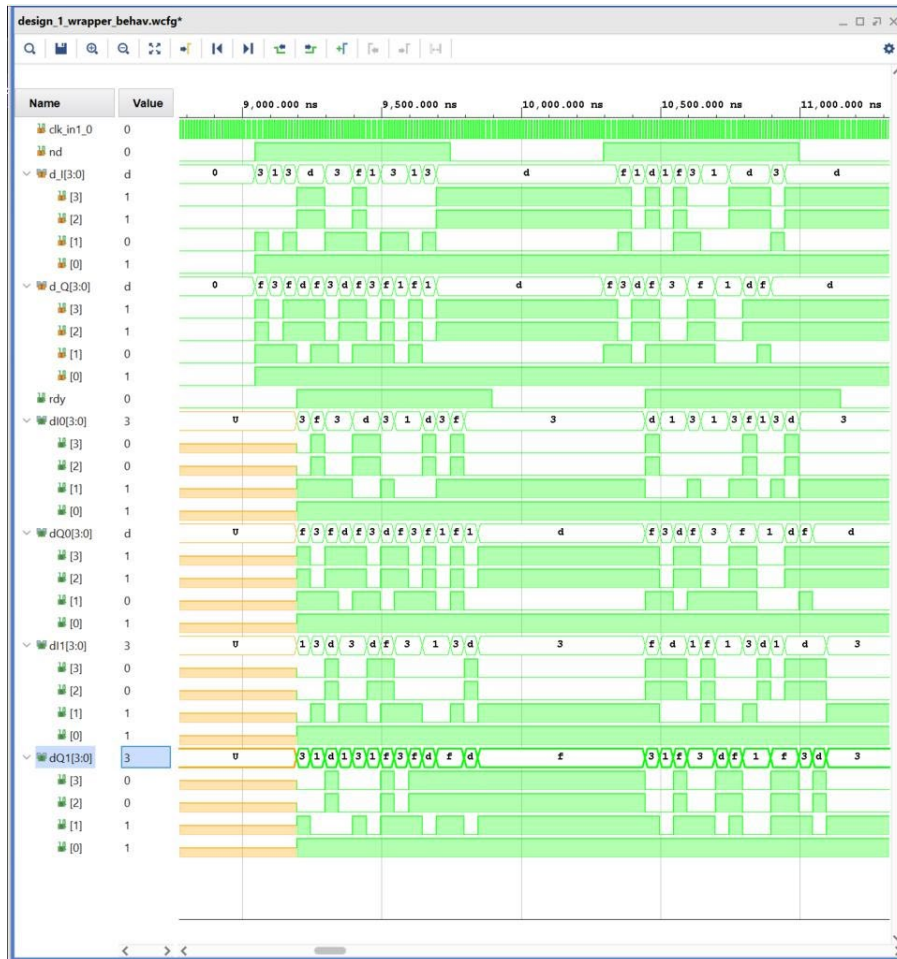
Figura 5 Simulación de bloque de modulación QAM.



Fuente: [Alamouti, 1998]

Figura 6 Diagrama de bloques de esquema Alamouti.

La Figura 7 presenta el resultado de aplicar dicho esquema. En base a ambas componentes resultantes de la modulación QAM es necesario enviar los datos en el orden presentado en la Tabla 1, como resultado se obtienen cuatro señales, estas son correspondientes a las señales de las 2 antenas del esquema.



Fuente: elaboración propia

Figura 7 Simulación de bloque codificador Alamouti.

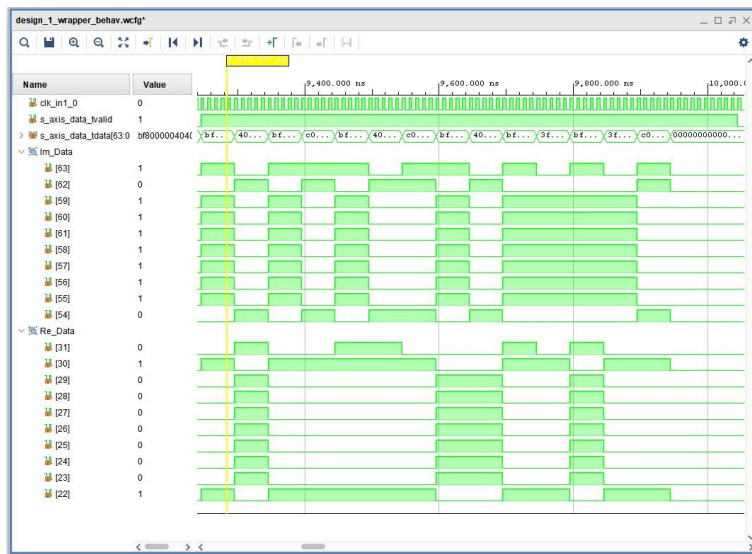
Tabla 1 Tabla de tiempos para esquema Alamouti.

Tiempos	Antena 0	Antena 1
Tiempo t	S0	S1
Tiempo t+T	-S1*	S0*

Fuente: elaboración propia

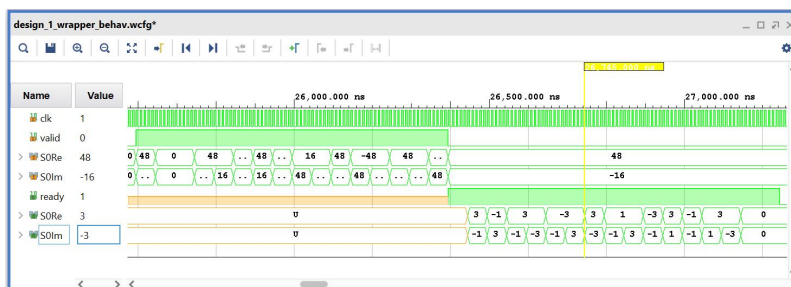
La operación IFFT se realizó utilizando el IP proporcionado por Vivado, sin embargo, es necesario acondicionar los datos que entran al bloque, cada componente se debe

expresar en el formato IEEE 754 y se deben incluir en una sola señal de datos. Esto produce un vector de 64 bits. La configuración utilizada en el IP fue una arquitectura radix burst-2, con una longitud de 16 y un reloj de 20 MHz. Además, el procesamiento de los datos por el IP genera los datos en orden invertido y es necesario agregar un bloque de acondicionamiento para reordenar los bits a un orden natural. La Figura 8 presenta la simulación del bloque concretamente para el primer set de datos de entrada. De forma similar para el bloque FFT se utilizará el IP de Vivado para transformadas rápidas de Fourier, este se construye con la misma configuración a su contraparte en transmisión y tras aplicar un acondicionamiento a los datos para revertir el formato IEEE754, el orden de salida de datos y el escalado introducido por la operación se obtienen la simulación de la Figura 9.



Fuente: elaboración propia

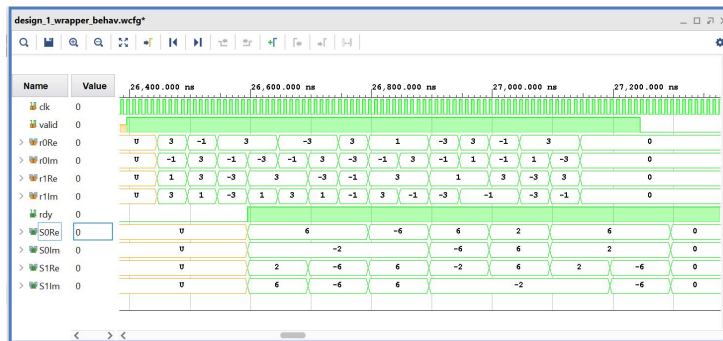
Figura 8 Simulación de bloque IFFT.



Fuente: elaboración propia

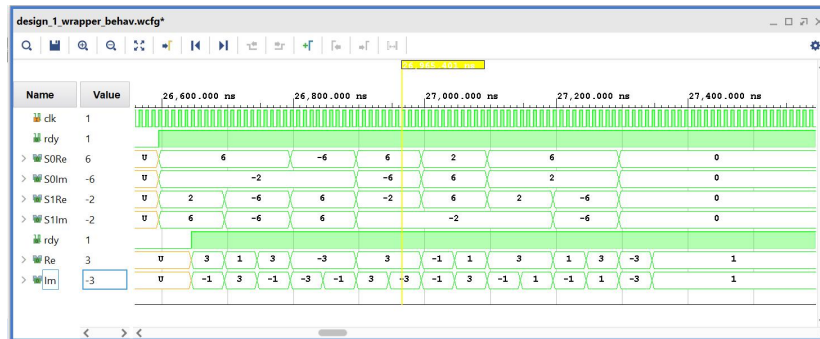
Figura 9 Simulación de bloque FFT.

Como se observó en la Figura 6 es necesario contar con un total de 3 bloques para realizar la decodificación del esquema Alamouti, sin embargo, considerando que se conoce el canal y no se introdujo ruido es posible omitir el estimador de canal resultando en dos operaciones, una etapa de combinación de datos y un detector de máxima verosimilitud. La Figura 10 presenta la primera etapa en donde se hace una combinación de los datos obtenidos tras aplicar el esquema Alamouti. Posteriormente se aplica el detector de máxima verosimilitud para obtener la secuencia de datos, esto se observa en la Figura 11.



Fuente: elaboración propia

Figura 10 Simulación de bloque combinador Alamouti.

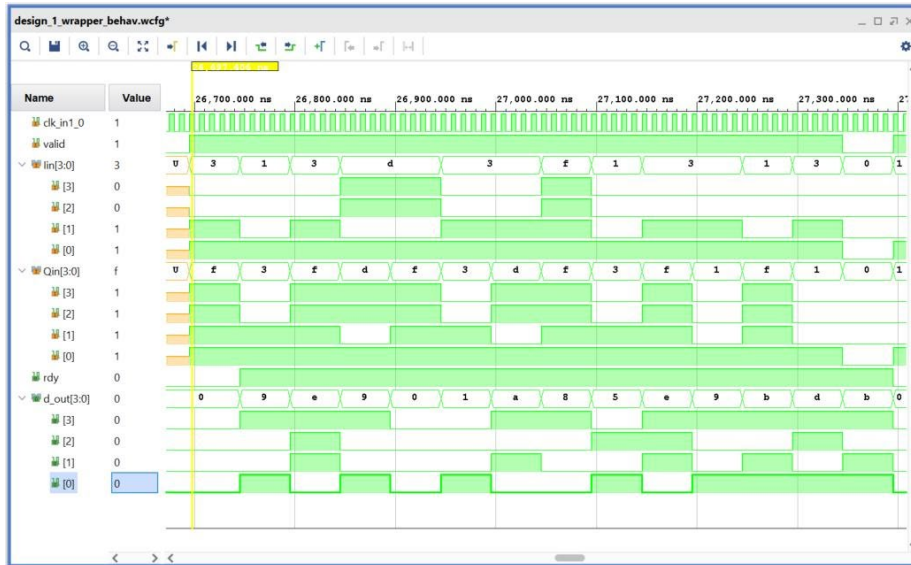


Fuente: elaboración propia

Figura 11 Simulación de bloque de máxima verosimilitud para esquema Alamouti.

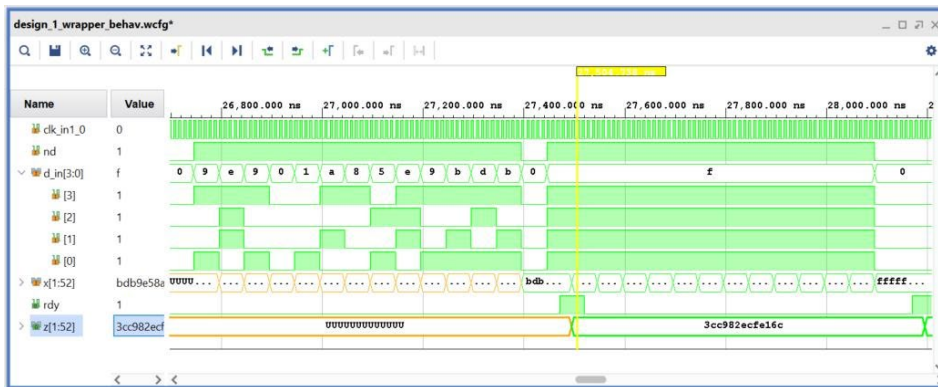
El funcionamiento del bloque demodulador QAM consiste en un demultiplexor que recibe una señal de datos de 4 bits y se transforma en una de 2 bits. La Figura 12 presenta la simulación de la demodulación. Posterior a la demodulación QAM se procede con el desintercalado de los datos, para aplicar este proceso es necesario agrupar los datos de salida del demodulador en un solo vector y posteriormente

aplicar un reajuste a los datos de acuerdo con la secuencia aplicada en la etapa de transmisión, la Figura 13 presenta la simulación de esta etapa.



Fuente: elaboración propia

Figura 12 Simulación de bloque demodulador QAM.

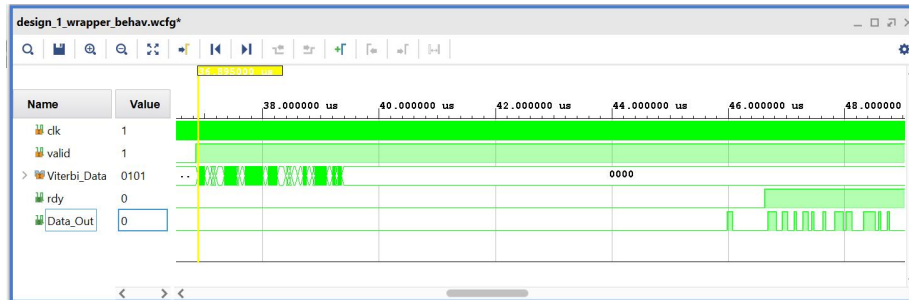


Fuente: elaboración propia

Figura 13 Simulación de bloque des intercalador.

Por último, se implementa un decodificador Viterbi por medio de un IP de Vivado, para la cual es necesario manejar la misma configuración utilizada en el convolucionador binario del transmisor, es decir, una longitud de restricción de 7, códigos de codificación de 171 y 133, una tasa de $\frac{1}{2}$ sin perforado, manejando la arquitectura paralela y con una longitud de retroceso de 42. La simulación de dicho bloque se puede apreciar en la Figura 14 en donde se tiene una señal de entrada

Viterbi_Data con los valores a decodificar y se obtiene una cadena de bits en la salida Data_Out la cual debe corresponder a la secuencia original ingresada.

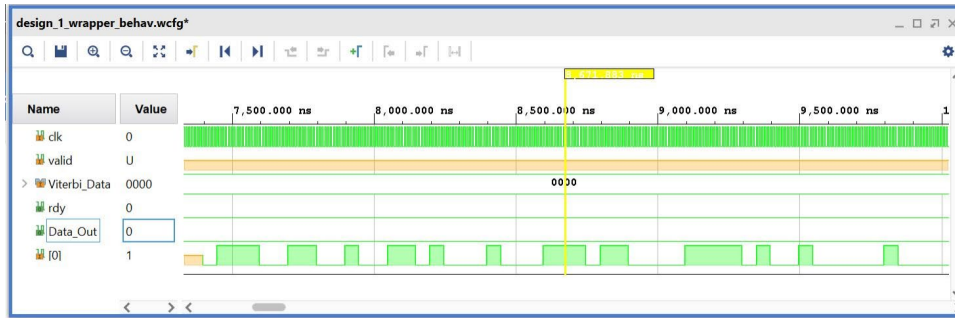


Fuente: elaboración propia

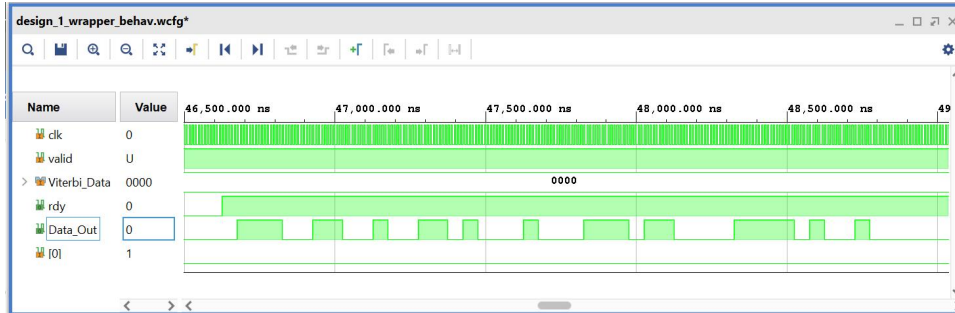
Figura 14 Simulación de bloque decodificador Viterbi.

3. Resultados

La aplicación de la primera prueba se dio directamente al evaluar entradas y salidas de cada uno de los bloques y comparando los resultados obtenidos en ambas etapas se observó que el funcionamiento de los bloques era el correcto ya que se tenían los mismos datos a la salida de cada bloque en la recepción con respecto a lo que entraba a cada bloque en transmisión. Tras confirmar el funcionamiento individual de cada bloque se realizó la comunicación de ambas etapas y se compararon la entrada y la salida del sistema completo obteniendo la Figura 15, en donde se tiene la entrada de datos a la etapa de transmisión y la salida de datos de la etapa de recepción respectivamente. Posteriormente es necesario realizar el envío de una imagen a través del sistema, para lo cual se utilizó Matlab con el fin de transformar una imagen en bits, posteriormente con la información obtenida a la salida del sistema se reconstruyó para facilitar la comparación entre la imagen original y la enviada a través del transceptor, dicha comparación se puede observar en la Figura 16. Se presentó un error en los primeros 8 bits de cada vector de bits, dicho error fue introducido por el bloque decodificador Viterbi. Es posible que, aplicando configuraciones distintas a las utilizadas en este proyecto, ya sea implementando la tasa de error de bit al bloque Viterbi o introduciendo los datos en formato de soft coding se pueda corregir en cierta medida dicho error y de no ser el caso se aplicaría una compensación a la cadena de bits.



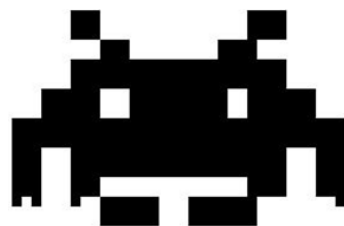
a) Datos de entrada a etapa de transmisión.



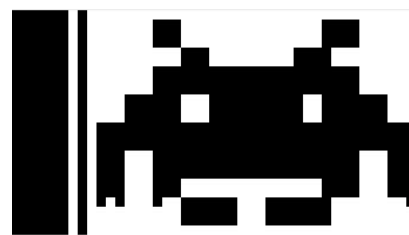
b) Datos de salida de etapa de recepción.

Fuente: elaboración propia

Figura 15 Comparación de datos originales con datos recibidos.



a) Imagen original.



b) Imagen recibida.

Fuente: elaboración propia

Figura 16 Comparación entre imagen original y recibida.

Finalmente se tiene la Tabla 2 en la cual se observa la utilización de recursos para este proyecto en la tarjeta FPGA Artix-7.

Tabla 2 Tabla de utilización de recursos de FPGA Artix-7.

FPGA	Artix-7 (XC7A200TFBG676-2)		
Recursos	Disponibles	Utilizado	Utilización (%)
LUTs	133,800	10,166	7.6%
Registros	267,600	15,376	5.75%
I/O	400	64	16%
RAM	365	18	4.93%
DSPs	740	32	4.32%

Fuente: elaboración propia

4. Discusión

En este artículo se realizó la emulación por medio de una tarjeta de FPGA del transceptor de la capa física del estándar 802.11n, estándar cuyo mayor impacto con respecto a sus predecesores es el incremento en velocidad de transferencia gracias a la implementación de la tecnología MIMO y Beamforming además del uso de tecnologías previamente utilizadas como lo son la modulación QAM y OFDM. Al realizar las pruebas al sistema se observó un error presente debido a las configuraciones del bloque de decodificación Viterbi, principalmente debido a la ausencia de una retroalimentación dentro del mismo bloque en base a la tasa de error de bit al realizar la decodificación, además se propone migrar el sistema al entorno de Simulink por medio de la herramienta de System Generator for DSP para facilitar la prueba del sistema con el uso de imágenes al ser posible extraer directamente los datos desde el espacio de trabajo de Matlab en lugar de introducir manualmente cada cadena de datos. Como tal el sistema aún tiene espacio para mejorar y será utilizado como base en una investigación para el desarrollo de un sistema más complejo y que será implementado y caracterizado por medio de FPGA y equipo de radiofrecuencias con el fin de probar el rendimiento de sistemas SDR trabajando a altas frecuencias.

5. Conclusiones

La emulación de este sistema en FPGA consume aproximadamente un 16% de recursos I/O 4.93% de memoria, 7.6% de bloques lógicos, 5.75% de registros y 4.32% de bloques de procesamiento de señales digitales para un Artix 7. Al comparar los datos de entrada del transmisor y los datos de salida del receptor se aprecia que se presentó un error de 1 bit en la transmisión de datos aleatorios, sin embargo, al realizar la prueba con la imagen el error presentado fue mayor y se presentó en cada uno de los arreglos que conformaban la imagen. Existen dos posibles métodos que pueden llegar a corregir dicho error, el primero es la implementación de un decodificador Viterbi con acceso a la tasa de error de bits con el fin de que este sea capaz de generar una secuencia más acertada en base a los estados que se le introdujeron. Por otro lado, se puede implementar un bloque de

estimación de canal para tomar en cuenta el ruido que se agrega en transmisión. Además, al observar la simulación se aprecia que le toma al sistema $49 \mu s$ todo el proceso de transmisión y recepción de un total de 52 datos, la transmisión toma un total de $25 \mu s$ y la recepción $24 \mu s$, es posible reducir estos tiempos empleando una arquitectura con un procesamiento más rápido en los bloques IFFT y FFT, aumentando la tasa de codificación de los datos y la magnitud del modulador QAM.

6. Bibliografía y Referencias

- [1] Alamouti S. M., A simple transmit diversity technique for wireless communications. *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 8, 1451–1458, IEEE, 1998.
- [2] Vo B. D., 802.11n Wireless Communication System Simulation Using Matlab and Implementation on FPGA, Faculty of Worcester Polytechnic Institute, Worcester, 2009.
- [3] Espinosa R. I., Uso de un FPGA (field programmable gate array) para la implementación de la sección de banda base de la capa física de un transmisor basado en el estándar IEEE 802.11 n en modo Greenfield, EPN Ecuador, Quito, 2011.
- [4] Gast M. S., 802.11 Wireless Networks: The Definitive Guide, 2da ed. O'Reilly Media, 2005.
- [5] IEEE Standard for Information Technology--Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks--Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Standards Association, IEEE Std 802.11-2020,1-4379, 2020.
- [6] Jiao X., Liu W., Mehari M., Aslam M., y Moerman I., openwifi: a free and open-source IEEE802.11 SDR implementation on SoC. *IEEE 91st Vehicular Technology Conference*, pp. 1-2, IEEE, 2020.
- [7] López F., El estándar IEEE 802.11 Wireless LAN, Universidad Politécnica, Madrid, España, 2010.

- [8] O'Hara B., IEEE 802.11 Handbook: A Designer's Companion, 1ra ed. Wiley-IEEE Press, 2012.
- [9] Sendra S., García M., Turro C., y Lloret J., WLAN IEEE 802.11 a/b/g/n Indoor Coverage and Interference Performance Study. *International Journal on Advances in Networks and Services*, 4(1), 209–222, IARIA, 2011.
- [10] Sharma R., Behavioral Modeling and FPGA Synthesis of IEEE 802.11 n Orthogonal Frequency Division Multiplexing (OFDM) Scheme, USF Tampa Graduate Theses and Dissertations, Tampa, 2016
- [11] Sujaya B. L., Y Prashanth S. B., PHY-DTR: An efficient PHY based digital transceiver for body coupled communication using IEEE 802.3 on FPGA platform. *International Journal of Advanced Computer Science and Applications*, 12(2), 2021.
- [12] Stratakos I., Papatheofanous E. A., Danopoulos D., Lentaris G., Reisis D., y Soudris D., Towards sharing one FPGA SoC for both low-level PHY and high-level AI/ML computing at the edge. 2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom) (pp. 76-81). IEEE, 2021.
- [13] Singh N., Santosh S. V. S., Darak S. J., Toward Intelligent Reconfigurable Wireless Physical Layer (PHY), *IEEE Open Journal of Circuits and Systems*, vol. 2, pp. 226-240, IEEE, 2021.