PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

DISEÑO DE UN SISTEMA DE TRANSMISIÓN/RECEPCIÓN BASADO EN OFDM PARA COMUNICACIONES PLC DE BANDA ANCHA

Tesis para optar el Título de **Ingeniero Electrónico**, que presenta el bachiller:

Edward Máximo Mitacc Meza

ASESOR: Jorge Benavides Aspiazu

Lima, noviembre de 2013

RESUMEN

En el presente trabajo se realizó el diseño en hardware de una arquitectura de un sistema de transmisión-recepción de banda ancha orientado a comunicaciones por las redes eléctricas (*power line communications, PLC)*, basado en el esquema de modulación FFT-OFDM e inspirado en el modelo de capa física del estándar IEEE 1901. La arquitectura fue descrita mediante lenguaje de descripción de hardware VHDL, para su posterior implementación en un dispositivo FPGA.

El sistema diseñado consta de dos módulos principales: el transmisor y el receptor. El primero se encarga de generar una señal OFDM a partir de una trama de entrada de 4096 bits, mientras que el segundo realiza el proceso inverso, es decir, decodifica una trama de 4096 bits a partir de una señal OFDM recibida. Para los procesos de modulación y demodulación, se emplean núcleos de IFFT y FFT de 4096 puntos, y se utiliza el esquema QPSK para la codificación de cada una de las subportadoras. Asimismo, ambos módulos, transmisor y receptor, cuentan con mecanismos de codificación y corrección de errores, a fin de reducir la propagación de los mismos en los paquetes de datos transmitidos.

La descripción en VHDL del sistema de transmisión-recepción diseñado fue sintetizada, utilizando las herramientas del software ISE 14.4 de Xilinx®, para el dispositivo FPGA Spartan-6 XC6SLX45. Entre los resultados obtenidos, destaca que la máxima frecuencia de operación alcanzada por el sistema es de 107,68 MHz. Asimismo, en simulaciones realizadas de la operación del sistema en presencia de modelos de ruido periódico síncrono y ruido periódico asíncrono, se obtuvieron cero errores para valores de SNR mayores a 11 dB.

A mis padres Maritza y Máximo, por su cariño, su apoyo incondicional, y sus enseñanzas y consejos

A Vicky, por ser una segunda madre para mí

A mis hermanos Edwin, Danitza y Erika, por ser un ejemplo a seguir

A mis primos John, Hugo, Cristian, a mis tíos Richard y Roger, por su apoyo sincero

A mi asesor Jorge Benavides, por toda la confianza y todo el apoyo mostrado

A mi estimado Grupo de Microelectrónica (GuE) y al Dr. Carlos Silva, por su constante apoyo en todo momento

A Henry Block, José Quenta, Cristopher Villegas, Andres Jacoby, Christian Cano, Guillermo Garayar, Sammy Cerida, Gonzalo Parra, Niels Prieto y demás compañeros del Grupo de Microelectrónica, por su apoyo y amistad

A Rodrigo Paz-Soldán, Jessenia Gonzalez, Gabriel Jiménez, Henry Iglesias y demás amigos, por su amistad brindada

A Pedro Aquino, Luis Miguel Guevara, Edgard Perez-Palma, y demás amigos

A todos, gracias!

“Imagination is more important than knowledge.
For knowledge is limited to all we now know and understand,
while imagination embraces the entire world,
and all there ever will be to know and understand.”

- Albert Einstein

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En las últimas décadas, la demanda por servicios multimedia de banda ancha se ha incrementado considerablemente, y continúa creciendo a un ritmo vertiginoso. Esto ha dado lugar al desarrollo de una amplia variedad de tecnologías de acceso de banda ancha, tanto alámbricas como inalámbricas. Entre las primeras, destaca la tecnología *Power Line Communications* (PLC), la cual consiste en el uso de las redes eléctricas como medio de transmisión de datos.

La tecnología PLC presenta grandes ventajas frente a otras tecnologías de acceso de banda ancha, entre las cuales destacan su amplio nivel de cobertura, facilidad de instalación y relativamente bajo costo. Sin embargo, diversos factores han limitado la implementación masiva de esta tecnología a nivel mundial. El principal obstáculo ha sido la ausencia de un estándar globalmente aceptado por la industria dedicada a la comercialización de PLC. No obstante, en febrero del 2011 se publicó el estándar IEEE 1901, el primer estándar de la IEEE para PLC, que cuenta con gran potencial para convertirse en el estándar PLC adoptado a nivel internacional.

En el presente trabajo de tesis, se propone el diseño en hardware de una arquitectura de un sistema de transmisión-recepción PLC de banda ancha basado en FFT-OFDM e inspirado en el estándar IEEE 1901. El diseño de la arquitectura se realiza mediante el lenguaje de descripción de hardware VHDL, y posteriormente se implementa en un dispositivo FPGA.

El presente documento se divide en cuatro capítulos. En el primer capítulo, se presenta una breve descripción sobre la tecnología PLC. En el segundo capítulo, se aborda el estado del arte de los transmisores-receptores PLC. Asimismo, se da una descripción de la modulación OFDM, del estándar IEEE 1901, de los dispositivos FPGA y del lenguaje VHDL. En el tercer capítulo, se presenta la arquitectura propuesta, y se detalla cada uno de sus bloques componentes. Finalmente, en el cuarto capítulo, se muestran las simulaciones realizadas y los resultados obtenidos.

CAPÍTULO 1:

POWER LINE COMMUNICATIONS

1.1. Introducción

Uno de los principales problemas asociados al acceso a Internet y otras aplicaciones de banda ancha es el denominado problema de la “última milla”, el cual aborda la problemática de cómo establecer una comunicación de alta velocidad, segura y de bajo costo entre el cliente y el proveedor de servicios de telecomunicaciones. El término “última milla” hace referencia a la distancia típica entre la red troncal del proveedor y las casas u oficinas a las cuales se les proporciona el servicio. Precisamente, este último tramo de la línea de comunicación entre el proveedor y suscriptor es donde se presentan los mayores inconvenientes que limitan la eficiencia de las transmisiones.

Actualmente, se dispone de una amplia variedad de tecnologías para establecer la conexión en la “última milla”, entre las cuales destacan DSL (*Digital Suscriber Line)*, fibra óptica, tecnologías inalámbricas y *Power Line Communications* (PLC). Cada tecnología presenta características distintas. Las dos primeras soportan altas velocidades de transmisión, pero presentan las desventajas de requerir cableado adicional y tener limitada cobertura. Por su parte, las tecnologías inalámbricas eliminan la necesidad de desplegar instalaciones alámbricas, mas son de costo relativamente alto. Por último, la tecnología PLC supera las desventajas mencionadas de las otras tecnologías, pero emplea un medio poco favorable para las comunicaciones.

En el presente capítulo se dará a conocer en qué consiste la tecnología PLC, así como también las aplicaciones actuales que presenta.

1.2. Descripción de la tecnología PLC

La tecnología *Power Line Communications* (PLC) consiste en la transmisión de información a través de las redes eléctricas. Esto significa que las mismas líneas usadas para proporcionar energía eléctrica pueden ser empleadas para proveer un medio para las comunicaciones.

La capacidad de poder transmitir señales de información en forma compartida con la señal de suministro eléctrico se debe a que ambas señales trabajan a frecuencias muy separadas entre sí. La señal de energía eléctrica es transmitida a 50 Hz ó 60 Hz, dependiendo del país donde se encuentre, mientras que las señales de datos se transmiten a frecuencias mayores a 1 MHz en el caso de aplicaciones de banda ancha.

Dado que la tecnología PLC emplea una infraestructura existente de gran presencia a nivel mundial como es la red eléctrica, presenta una serie de ventajas sobre las otras tecnologías de acceso de banda ancha:

- Amplio nivel de cobertura, incluso en los lugares más remotos.
- Instalación sencilla y rápida para el usuario (sólo se necesita la instalación de módems PLC).
- Costos accesibles.
- Elevada escalabilidad, pues las redes PLC pueden crecer y adaptarse a cambios sin perder la calidad de los servicios que brinda.

Sin embargo, esta tecnología hace uso de un medio que no ha sido diseñado específicamente para propósitos de comunicaciones. En efecto, diversos factores como la atenuación, el ruido y el efecto multicamino afectan en gran medida las comunicaciones por la red eléctrica. El primero es causado por las pérdidas de potencia en los cables, y se incrementa con la distancia y la frecuencia. Por su parte, el ruido en las líneas eléctricas resulta de la combinación de diversos tipos de ruido: ruido de fondo coloreado (*coloured background noise)*, ruido de banda estrecha y ruido impulsivo [1]. Por último, el efecto multicamino se basa en las múltiples reflexiones que una señal experimenta producto de las numerosas ramas y desacoples de impedancias de la red eléctrica.

La vulnerabilidad de la red eléctrica a los distintos tipos de ruido supone el empleo de técnicas de modulación que se adapten a las características del medio eléctrico, a fin de transmitir información de manera segura y eficiente. Los esquemas de modulación más utilizados en los dispositivos PLC actuales son DSSSM (*Direct Sequence Spread Spectrum Modulation*) y OFDM (*Orthogonal Frequency Division Multiplex*). El primero se caracteriza por su capacidad de operar con baja densidad

espectral de potencia, mientras que el segundo tiene la particularidad de utilizar un gran número de portadoras con anchos de banda estrechos.

1.3. Aplicaciones de la tecnología PLC

Una de las principales aplicaciones de la tecnología PLC es el acceso a Internet. Para ello, se emplea un dispositivo PLC (maestro) que introduce señales de datos provenientes de la red troncal de Internet en las líneas eléctricas de bajo voltaje. A fin de recibir la señal de Internet, cada suscriptor emplea otro dispositivo PLC (esclavo), el cual distribuye dicha señal en la red eléctrica interior del edificio. Esto se ilustra en la figura 1.1.

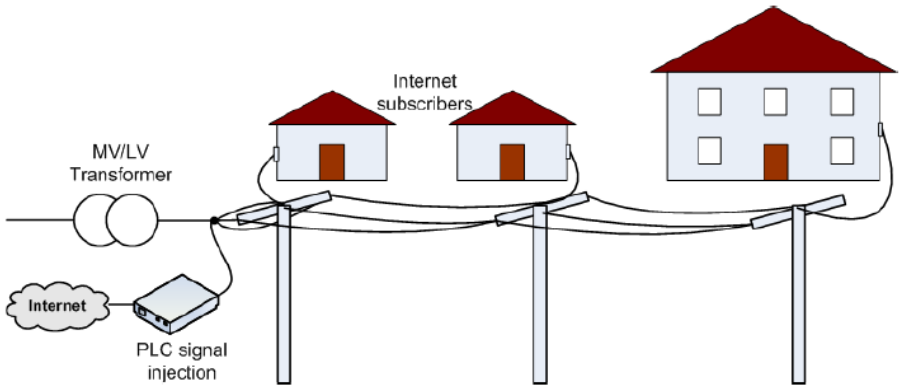


Figura 1.1: Uso de PLC para proporcionar acceso a Internet [2]

Asimismo, se emplea la tecnología PLC para implementar redes de área local (LAN) en el interior de casas y oficinas mediante el uso de las redes eléctricas interiores. La gran ventaja de PLC para esta aplicación es que las tomas de suministro eléctrico están presentes en gran número en el interior de los inmuebles, lo cual facilita la conexión de múltiples computadores a la red LAN. Además, permite compartir impresoras y dispositivos similares sin la necesidad de cableado adicional.

Adicionalmente, las altas tasas de datos ofrecidas por la tecnología PLC permiten su empleo para aplicaciones de voz y video. La transmisión de voz a través de PLC se realiza mediante la técnica de voz sobre IP (VoIP). En el caso de aplicaciones de video, se emplea para la transmisión de video en tiempo real (*video streaming)*, video vigilancia, video conferencia y televisión digital de alta definición.

CAPÍTULO 2:

MARCO TEÓRICO Y PRESENTACIÓN DEL ASUNTO DE ESTUDIO

En el presente capítulo, se aborda el estado del arte del trabajo realizado, y se describen los conceptos teóricos más importantes empleados en la elaboración del mismo. En primer lugar, se presenta el tema desarrollado, y se presentan trabajos relacionados al realizado. Luego, se da una breve descripción de la modulación OFDM, del estándar IEEE 1901, de los dispositivos FPGA y del lenguaje VHDL.

2.1. Estado del Arte

2.1.1. Presentación del asunto de estudio

Dado el importante despliegue de las redes eléctricas tanto a nivel urbano como rural, la tecnología *Power Line Communications* (PLC) se presenta como una alternativa de bajo costo y universal para proveer servicios de banda ancha. En particular, representa una posible solución para reducir la brecha de acceso a los servicios de telecomunicaciones en zonas rurales del Perú.

Sin embargo, a pesar de que la tecnología PLC tuvo un inicio muy prometedor cuando se empleó por primera vez para aplicaciones de banda ancha a inicios de la década de 1990 [3], aún no se ha consolidado a nivel mundial. El principal factor que ha limitado la implementación masiva de esta tecnología ha sido la ausencia de un estándar globalmente aceptado por la industria dedicada a la comercialización de PLC. Esto se refleja en el hecho de que los productos PLC disponibles actualmente están basados en diferentes estándares, y la mayoría de ellos son soluciones propietarias.

No obstante, en febrero del 2011 se publicó el estándar IEEE 1901, el primer estándar de la IEEE para PLC. A la fecha, este estándar cuenta con gran apoyo dentro de la industria PLC, principalmente debido a que asegura un alto nivel de fiabilidad y seguridad, y una alta velocidad en las comunicaciones por las redes eléctricas. Por tanto, es inminente su consolidación como único estándar PLC a nivel mundial.

El objetivo del presente trabajo es realizar el diseño de la arquitectura de un transmisor-receptor PLC de banda ancha basado en el esquema de modulación FFT-

OFDM e inspirado en el estándar IEEE 1901, para su implementación en un dispositivo FPGA.

2.1.2. Estado de la Investigación

A la fecha, se han desarrollado a nivel industrial los primeros productos PLC compatibles con el estándar IEEE 1901. El consorcio *HomePlug Powerline Alliance* publicó en el 2012 su propia especificación denominada *HomePlug AV2* para productos basados en el estándar IEEE 1901, donde garantiza velocidades de transmisión superiores a 200 Mbps [4]. Asimismo, el consorcio *HD-PLC Alliance* desarrolló en el mismo año sus primeros productos compatibles con el estándar en mención, en los cuales permite tasas de transmisión de hasta 240 Mbps en su capa física [5].

A nivel académico, no se han publicado, a la fecha, trabajos relacionados al diseño de transmisores-receptores PLC basados en el estándar IEEE 1901. Sin embargo, existe una serie de publicaciones sobre diseños de sistemas de transmisión-recepción PLC basados en OFDM e implementados sobre dispositivos FPGA. A continuación, se presentan aquéllos que se han empleado como referencia para el desarrollo de este trabajo.

El diseño de módem PLC propuesto por Shih-Chieh Hsu [6] cumple con las especificaciones del estándar *HomePlug 1.0* de *HomePlug Powerline Alliance*. Este transmisor-receptor emplea modulación OFDM de 128 sub-portadoras y está basado en una FFT de radix-4 de 256 puntos. Además, utiliza codificación DBPSK para cada una de las sub-portadoras.

El diseño de transmisor-receptor PLC propuesto por Sánchez [7] está basado en el estándar PRIME. Se caracteriza por utilizar modulación OFDM de 96 sub-portadoras y se basa en una FFT de 512 puntos. Asimismo, emplea alternadamente los esquemas DBPSK, DQPSK y D8PSK para la codificación de las sub-portadoras.

El transmisor-receptor PLC implementado por García-Baleón [8] utiliza un esquema de modulación OFDM de 31 sub-portadoras y está basado en una FFT de 64 puntos. Adicionalmente, utiliza codificación 4-QAM para cada una de las sub-portadoras.

2.2. Modulación OFDM

La modulación OFDM (*Orthogonal Frequency Division Multiplexing*) es una técnica de modulación multiportadora en la cual el espectro de frecuencia disponible es dividido en subportadoras ortogonales, cada una modulada por un flujo de datos de baja velocidad [9]. La ortogonalidad entre las subportadoras permite que éstas sean espaciadas más cerca una de otra, de modo que se logra una alta eficiencia espectral y se reduce el ancho de banda empleado. A continuación, se muestra el espectro de frecuencia de un símbolo OFDM (figura 2.1).

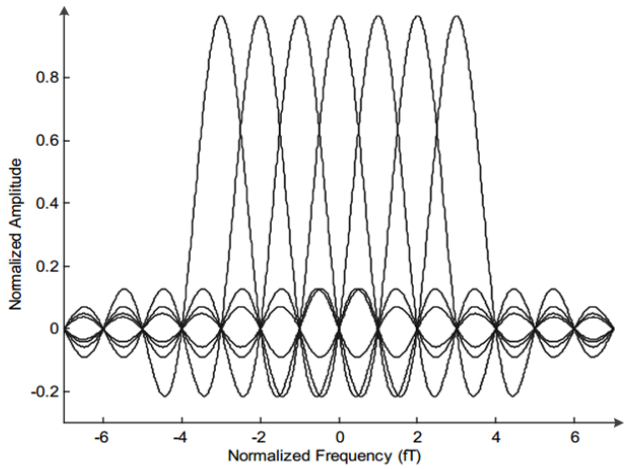


Figura 2.1: Espectro de frecuencia de un símbolo OFDM [10]

Asimismo, la modulación OFDM es efectiva frente a la interferencia inter-símbolo (ISI), causada por el efecto multicamino, y al desvanecimiento selectivo en frecuencia [9]. Sin embargo, su principal desventaja es que presenta alta sensibilidad a desviaciones de frecuencia y fase, las cuales afectan la ortogonalidad de las subportadoras.

Los sistemas de transmisión-recepción digitales basados en OFDM realizan la modulación/demodulación multiportadora por medio del empleo de la Transformada Inversa Discreta de Fourier (IDFT) y la Transformada Discreta de Fourier (DFT), respectivamente [11].

2.2.1. Transformada Discreta de Fourier: IDFT y DFT

La Transformada Inversa Discreta de Fourier (IDFT) se emplea para convertir las muestras del espectro de una señal en un igual número de muestras de dicha señal

en el dominio del tiempo. La IDFT $x(n)$, $n = 0, \ldots, N-1$, de una secuencia $X(k)$, $k = 0, \ldots, N-1$, está descrita por la ecuación 2.1.

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{\frac{jnk2\pi}{N}} \quad , \quad n = 0, \ldots, N-1 \qquad (\text{Ec. } 2.1)$$

La Transformada Discreta de Fourier (DFT) realiza el proceso inverso, es decir, convierte las muestras de una señal en el dominio del tiempo en un igual número de muestras de dicha señal en el dominio de la frecuencia. La DFT $X(k)$, $k = 0, \ldots, N-1$, de una secuencia $x(n)$, $n = 0, \ldots, N-1$, está descrita por la ecuación 2.2.

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-\frac{jnk2\pi}{N}} \quad , \quad k = 0, \ldots, N-1 \qquad (\text{Ec. } 2.2)$$

La implementación computacional de la IDFT y la DFT se realiza de forma eficiente por medio de los algoritmos de la Transformada Inversa Rápida de Fourier (IFFT) y la Transformada Rápida de Fourier (FFT) [12].

2.3. Estándar IEEE 1901

El estándar IEEE 1901 está definido para dispositivos PLC de banda ancha, o también llamados dispositivos BPL (*broadband over power line)*. Este estándar permite velocidades de transmisión mayores a 100 Mbps, y emplea frecuencias de transmisión menores a 100 MHz [13]. Asimismo, está enfocado en proveer seguridad y calidad de servicio (QoS) en las comunicaciones por las redes eléctricas.

El estándar establece especificaciones para la capa física (PHY) y la subcapa de control de acceso al medio (MAC) de la capa de enlace de datos (*data link layer)*, definidas por el modelo de referencia OSI (*Open Systems Interconnection*) de la Organización Internacional para la Estandarización (*International Organization for Standardization, ISO)*.

La capa física (PHY) del estándar soporta dos tipos: una basada en la modulación FFT-OFDM y otra basada en la modulación wavelet-OFDM. Respecto a la primera, sobre la cual se basa el presente trabajo, utiliza núcleos de IFFT y FFT como elementos principales de la modulación y demodulación, respectivamente. Asimismo, emplea un rango de frecuencias para las subportadoras de 1.8 MHz a 50 MHz.

En la figura 2.2 se presenta el modelo de capa física propuesto por el estándar. Se observa que incluye, tanto en el transmisor como en el receptor, tres etapas separadas de procesamiento: la primera para los datos de control de trama del estándar TIA-1113 (para compatibilidad con dicho estándar), la segunda para los datos de control de trama del estándar IEEE 1901, y la tercera para los datos de carga útil (*payload data)* del estándar IEEE 1901. Además, el núcleo de IFFT en el transmisor es de 512 puntos para los datos de control de trama del estándar TIA-1113, y de 4096 puntos para los datos de control y de carga útil del estándar IEEE 1901.

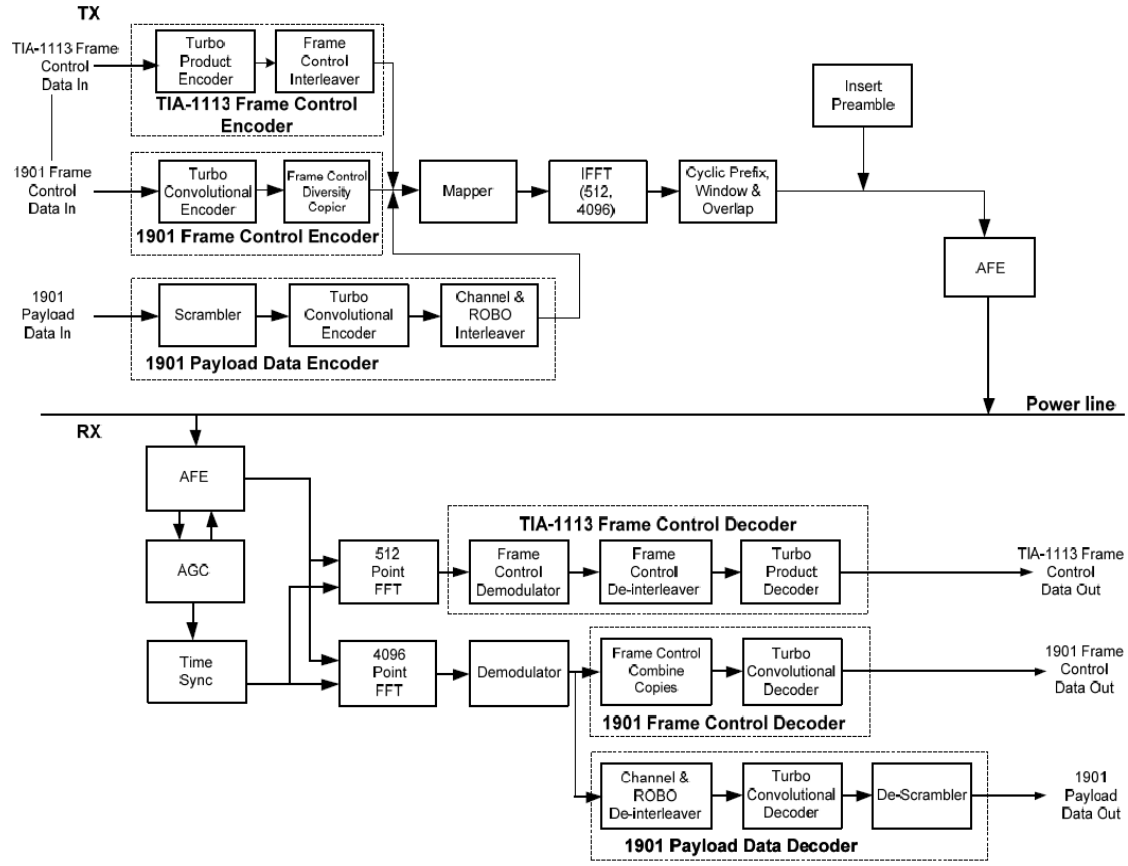


Figura 2.2: Modelo de capa física (PHY) propuesto por el estándar IEEE 1901 [13]

2.4. Dispositivo Lógico Programable FPGA

Un *field-programmable gate array* (FPGA) es un dispositivo lógico que contiene un arreglo bidimensional de celdas lógicas genéricas e interruptores programables [14], como se muestra en la figura 2.3.

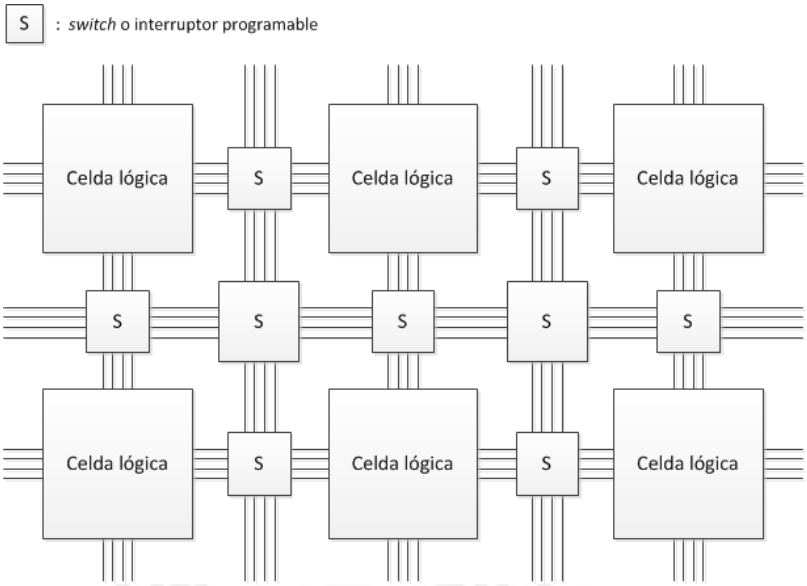


Figura 2.3: Estructura conceptual de un dispositivo FPGA

Una celda lógica puede ser configurada para realizar una determinada función, y un interruptor programable puede ser empleado para proveer interconexiones entre las celdas lógicas. Para la implementación de un diseño determinado, se especifican las funciones de cada celda lógica y se activan o desactivan las conexiones de cada interruptor programable. Dado que la configuración del dispositivo es realizada por el usuario, se dice que es "programable en el campo" o *field-programmable*.

2.5. Lenguaje de Descripción de Hardware VHDL

Los lenguajes de descripción de hardware (*hardware description languages* o HDLs) son empleados para modelar o describir sistemas digitales en un nivel de abstracción deseado, ya sea desde el punto de vista estructural o comportamental [14]. A diferencia de los lenguajes de programación, donde las instrucciones se ejecutan en forma secuencial, los HDLs cuentan con sentencias concurrentes, es decir, que se ejecutan en simultáneo.

Los HDLs más empleados a nivel mundial son VHDL y Verilog, los cuales se han establecido como estándares industriales. Ambos son utilizados principalmente para implementar circuitos digitales en dispositivos lógicos programables como FPGAs o CPLDs, o en circuitos integrados de aplicación específica (ASICs).

CAPÍTULO 3:

DISEÑO DE LA ARQUITECTURA DEL SISTEMA DE TRANSMISIÓN-RECEPCIÓN PLC BASADO EN FFT-OFDM

En el presente capítulo, se describe de forma detallada la arquitectura propuesta para la implementación del sistema de transmisión-recepción PLC. En primer lugar, se presentan los objetivos generales y específicos del diseño. Luego, se realiza una descripción general del sistema diseñado, y se describe detalladamente la arquitectura de cada uno de los módulos que lo componen. Finalmente, se presentan las consideraciones tomadas en la descripción en VHDL de las arquitecturas diseñadas.

3.1. Objetivos

3.1.1. Objetivo General

Realizar el diseño de un sistema de transmisión-recepción basado en el esquema de modulación FFT-OFDM, orientado a comunicaciones por la red eléctrica (PLC) e inspirado en el estándar IEEE 1901.

3.1.2. Objetivos Específicos

- Diseñar a nivel de bloques funcionales la arquitectura del sistema de transmisión-recepción.
- Describir la arquitectura diseñada por medio del lenguaje de descripción de hardware VHDL.
- Realizar una arquitectura modular, de modo que cada módulo diseñado, tanto en el transmisor como en el receptor, sea independiente de los otros.
- Realizar una verificación funcional del sistema descrito, por medio de bancos de prueba o *test benches*, a fin de comprobar su correcto funcionamiento.
- Realizar simulaciones de distintos tipos de ruido típicos del canal eléctrico, y evaluar la eficiencia del sistema de transmisión-recepción diseñado frente a estos tipos de ruido, en términos de tasa de errores de bit (BER).

3.2. Descripción de la arquitectura del sistema de transmisión-recepción PLC basado en FFT-OFDM

En la figura 3.1 se muestra la estructura general de la arquitectura propuesta para el sistema de transmisión-recepción PLC.

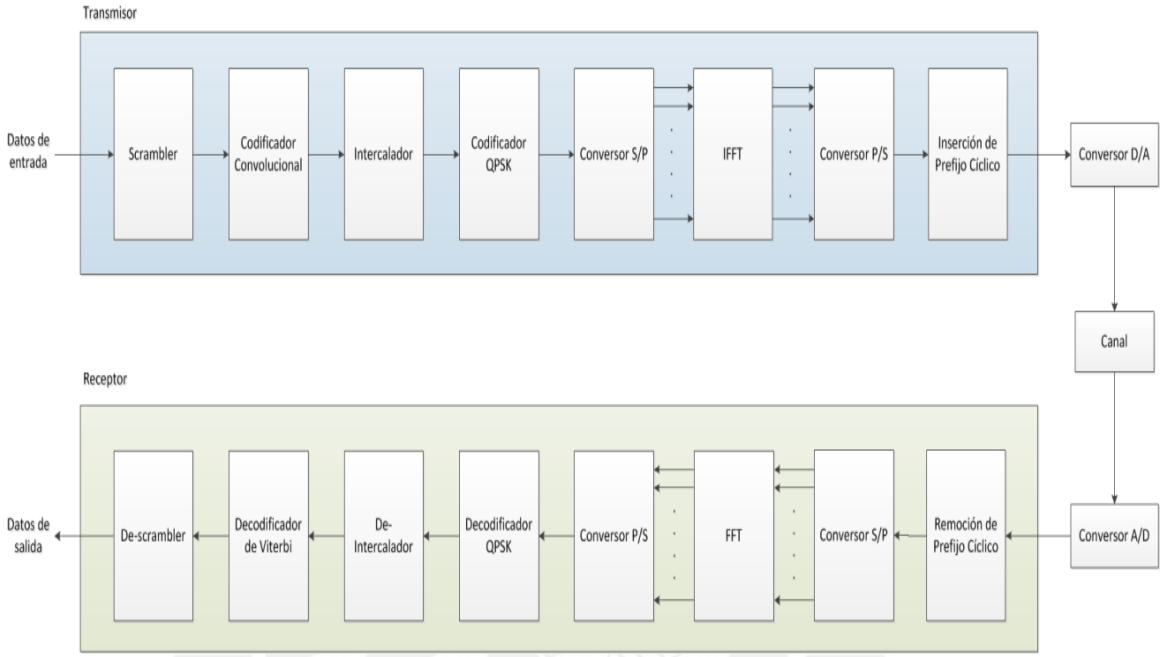


Figura 3.1: Arquitectura propuesta para el sistema de transmisión-recepción PLC

La arquitectura presentada está basada en el modelo que propone el estándar IEEE 1901 para la capa física de un sistema de transmisión-recepción basado en FFT-OFDM (figura 2.2). Específicamente, se basa en las etapas de procesamiento de datos que recomienda el estándar para la transmisión-recepción de datos de carga útil (*payload data*).

Sin embargo, la arquitectura diseñada presenta ciertas diferencias con respecto al modelo presentado por el estándar IEEE 1901. La distinción principal consiste en el empleo de un mecanismo de corrección de errores de menor complejidad. En lugar del Codificador-Decodificador Turbo Convolucional que se especifica en el estándar, se utiliza un Codificador Convolucional en el transmisor y un Decodificador de Viterbi en el receptor. Asimismo, los bloques de intercalado de datos presentan una estructura distinta, pues trabajan con una menor cantidad de bits de paridad, que son generados por el Codificador Convolucional. Por último, los bloques

IFFT/FFT no emplean los ángulos de fase que especifica el estándar para cada una de las subportadoras. Estas diferencias serán detalladas en los subcapítulos posteriores.

A continuación, se explica de manera general las operaciones llevadas a cabo tanto en el transmisor como en el receptor, y se describe cada uno de sus bloques componentes.

3.2.1. Descripción de la arquitectura del transmisor

En la figura 3.2 se muestra un esquema del transmisor implementado, donde se especifica el número de bits de datos que emplea cada módulo que lo compone.

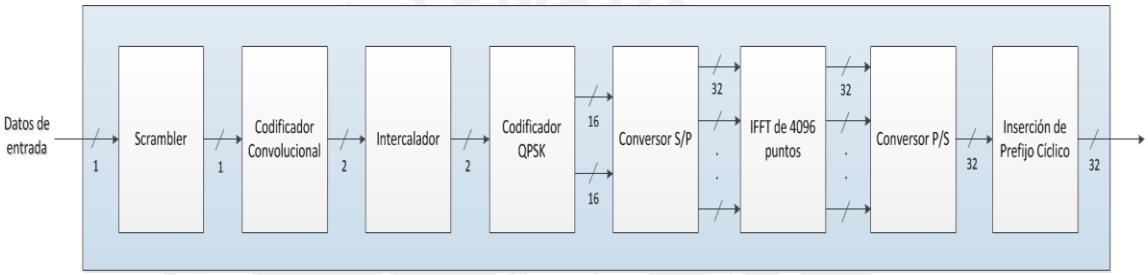


Figura 3.2: Arquitectura del transmisor

Para la transmisión de un símbolo OFDM, se ingresa un grupo de 4096 bits de forma serial. Cada uno de estos bits es procesado por el bloque *scrambler*, el cual modifica los valores de los bits de forma pseudo-aleatoria, con la finalidad de evitar secuencias repetitivas de datos.

Luego, el codificador convolucional genera un bit adicional, denominado bit de paridad, por cada bit de información que recibe. Este bit adicional de redundancia permite la detección y corrección de errores en el receptor.

A continuación, el bloque intercalador realiza un reordenamiento de los bits que recibe, a fin de separar bits o bloques de bits consecutivos. De esta manera, permite que los errores de ráfaga puedan ser detectados y corregidos en el receptor.

Seguidamente, el codificador QPSK recibe datos en grupos de dos bits y los asigna a un punto complejo de la constelación QPSK. Cada punto tiene una correspondencia en componente en fase (I) y componente en cuadratura (Q), que son codificados en palabras de 16 bits.

Luego, un conversor serial-paralelo agrupa las dos palabras de 16 bits recibidas y forma palabras de 32 bits expresadas de forma paralela. De este modo, se forman las 4096 entradas paralelas que requiere el módulo IFFT.

A continuación, el módulo de Transformada Inversa Rápida de Fourier (IFFT) realiza la modulación OFDM propiamente dicha, dando como resultado un símbolo OFDM compuesto de 4096 muestras en el tiempo, cada una con valores de 32 bits.

En seguida, un conversor paralelo-serial convierte los datos paralelos en un flujo serial de palabras de 32 bits, y finalmente, se realiza la inserción de prefijo cíclico, que consiste en una réplica de la parte final del símbolo OFDM al principio del mismo. Para el transmisor diseñado, se emplea un prefijo cíclico de 1252 muestras, de modo que se genera un símbolo extendido de 5348 muestras. Dicho símbolo es enviado a un conversor digital-analógico para su transmisión.

3.2.1.1. Bloque Scrambler

El bloque *scrambler* o aleatorizador modifica de forma pseudo-aleatoria los bits de datos que ingresan al transmisor. Su objetivo es evitar la repetición periódica de secuencias de datos, las cuales pueden ocasionar que el espectro de potencia presente una concentración indeseable de potencia en torno a ciertas frecuencias discretas, y por tanto reduce la interferencia electromagnética que el transmisor puede producir [7].

El proceso de aleatorización del *scrambler* consiste en realizar una suma de módulo 2 (operación *xor*) entre el flujo de datos de entrada y una secuencia pseudo-aleatoria generada a partir de un registro de desplazamiento con retroalimentación lineal (LFSR). Este último define su operación por medio de un polinomio característico, el cual establece la secuencia pseudo-aleatoria que se genera.

El circuito diseñado está basado en un LFSR de 10 bits que está definido por el polinomio generador $S(x)$ (ecuación 3.1). Su estado inicial es el valor ”1111111111” en binario, es decir, toma dicho valor cada vez que recibe el primer bit de una nueva trama de datos.

$$S(x) = x^{10} + x^3 + 1 \qquad \text{(Ec. 3.1)}$$

En la figura 3.3 se observa el circuito *scrambler* diseñado, según propone el estándar IEEE 1901, donde *x(n)* representa el bit de entrada (en el tiempo *n*), *k(n)* representa el bit generado por el LFSR, y *c(n)* representa el bit de salida del *scrambler*.

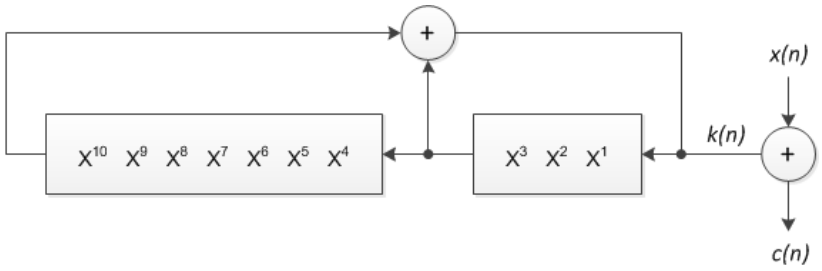


Figura 3.3: Circuito lógico de *scrambler*

A continuación, se presenta una vista esquemática general del bloque *scrambler* diseñado (figura 3.4).

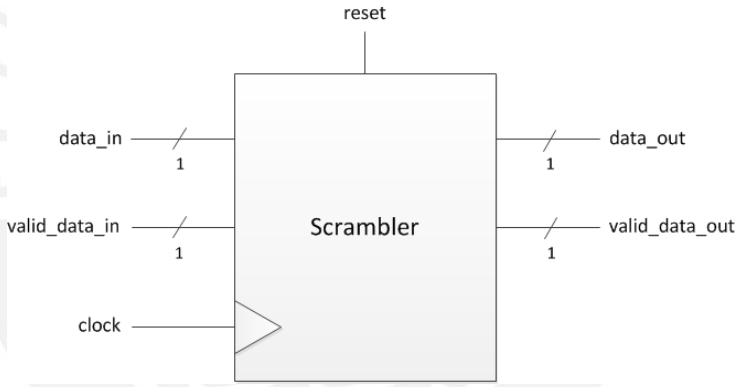


Figura 3.4: Esquema general del bloque *scrambler*

Las entradas y salidas del *scrambler* son:

- *Reset*: señal que reinicia el circuito a sus condiciones iniciales.
- *Clock*: señal de reloj, común a todo el sistema.
- *Data_in*: flujo de datos de entrada, el cual es procesado en el scrambler.
- *Valid_data_in*: señal que indica cuándo se recibe una trama válida de datos. Esta señal, además, reinicializa el LFSR cada vez que se activa nuevamente luego de permanecer inactiva.
- *Data_out*: señal de datos de salida, generada por el circuito lógico presentado en la figura 3.3.
- *Valid_data_out*: señal que indica cuándo los datos en la salida son válidos.

3.2.1.2. Bloque Codificador Convolucional

El bloque Codificador Convolucional, en conjunto con el bloque Decodificador de Viterbi, forma un mecanismo de detección y corrección de errores. En general, un Codificador Convolucional con parámetros (*n, k, K*) convierte *k* bits de información en *n* bits de canal, donde $m = n - k$ es la cantidad de bits de redundancia o de paridad. El tercer parámetro, *K*, es llamado longitud de limitante (*constraint length)*, y especifica el número de bits del flujo de datos de entrada que son usados en los cálculos de los bits de paridad [15].

La arquitectura general de un Codificador Convolucional consiste de un registro de desplazamiento que almacena los últimos *K* valores del flujo de datos de entrada, y de un circuito combinacional, basado en sumadores de módulo 2 (compuertas *xor*), que genera los *n* bits de salida a partir de combinaciones lógicas entre los bits contenidos en el registro de desplazamiento.

El Codificador Convolucional diseñado está basado en el modelo propuesto en [16], y tiene parámetros $k = 1$, $n = 2$ y $K = 3$. Esto es, genera dos bits en su salida por cada bit que recibe en la entrada, y está compuesto por un registro de desplazamiento de 3 bits. Además, es de tipo recursivo, puesto que una de las salidas se conecta directamente a la entrada, tal como se observa en el diagrama esquemático presentado en la figura 3.5.

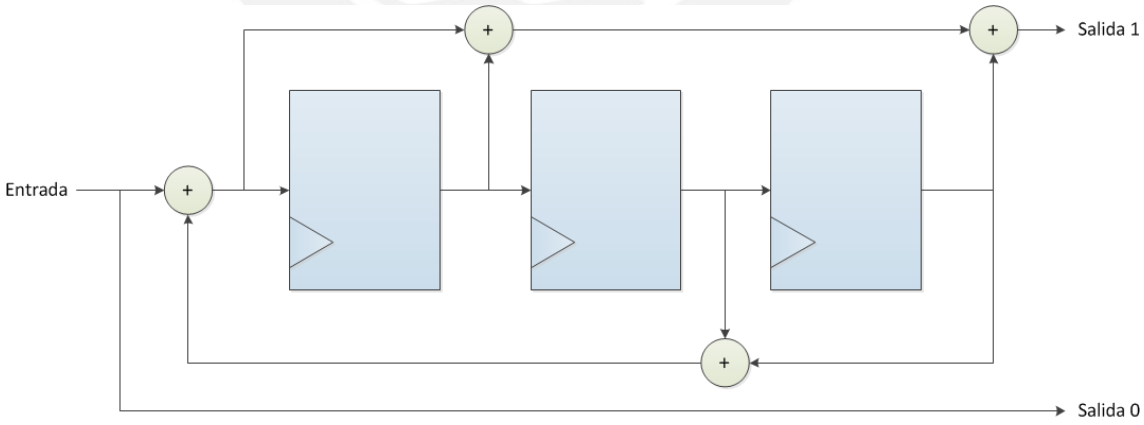


Figura 3.5: Circuito esquemático de Codificador Convolucional

El funcionamiento del Codificador Convolucional se puede expresar mediante el diagrama de estados presentado en la figura 3.6. Este diagrama presenta los posibles

estados del codificador (valores del registro de desplazamiento) y las posibles transiciones entre ellos. Se observa que en total se presentan ocho estados, y que el estado inicial es “000”, es decir, el codificador toma dicho estado cada vez que se reinicializa. En la figura, la señal *x* representa el bit de entrada, mientras que la señal *y* representa la señal de salida de dos bits, la cual resulta de la concatenación de los bits de las salidas 1 y 0 mostradas en la figura 3.5.

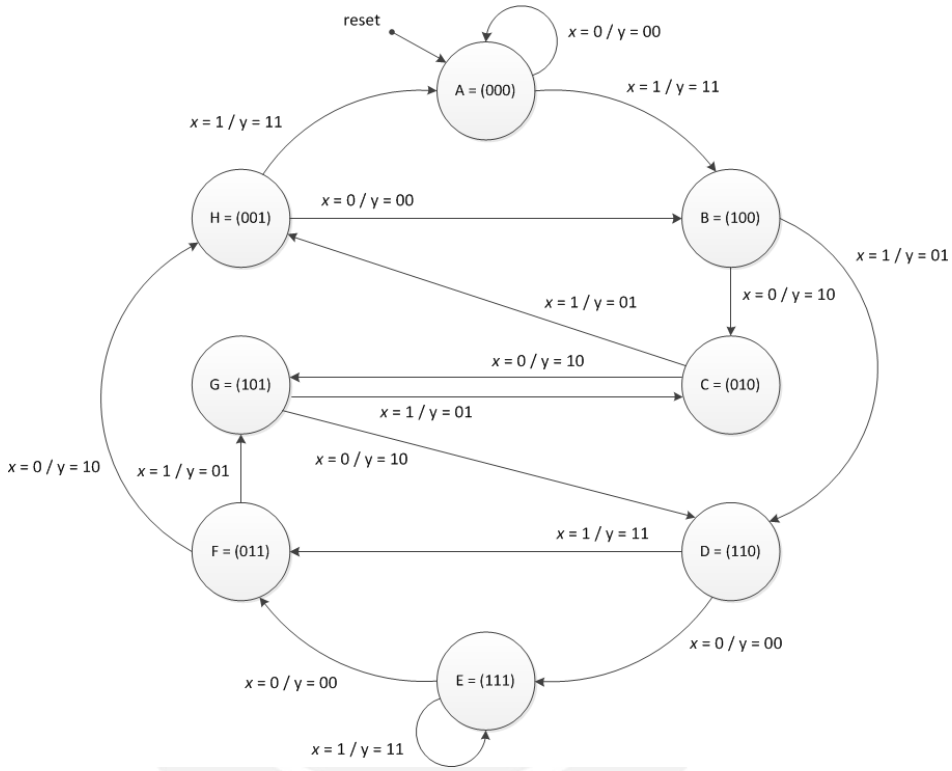


Figura 3.6: Diagrama de estados de Codificador Convolucional

En la figura 3.7 se muestra el diagrama de Trellis del Codificador Convolucional, el cual es una representación gráfica de las transiciones de estado que pueden ocurrir en el codificador desde su inicialización. En la dirección horizontal se tienen los instantes de tiempo discreto, en tanto que en el eje vertical se representan todos los posibles estados del registro de desplazamiento. Como se observa, se parte del estado inicial “000”, desde el cual se realiza, en el siguiente instante de tiempo, una transición hacia uno de dos posibles estados, según el bit de entrada sea ‘0’ ó ‘1’. En adelante, se muestran las posibles transiciones de estados en los primeros seis instantes de tiempo, luego de los cuales se repite el mismo patrón de transiciones dado desde el instante t_3. En el diagrama, se indica mediante líneas continuas las transiciones ocurridas cuando

el bit de entrada es ‘0’, y mediante líneas punteadas aquellas transiciones producidas cuando el bit de entrada es ‘1’. Además, se indican las salidas generadas en cada caso.

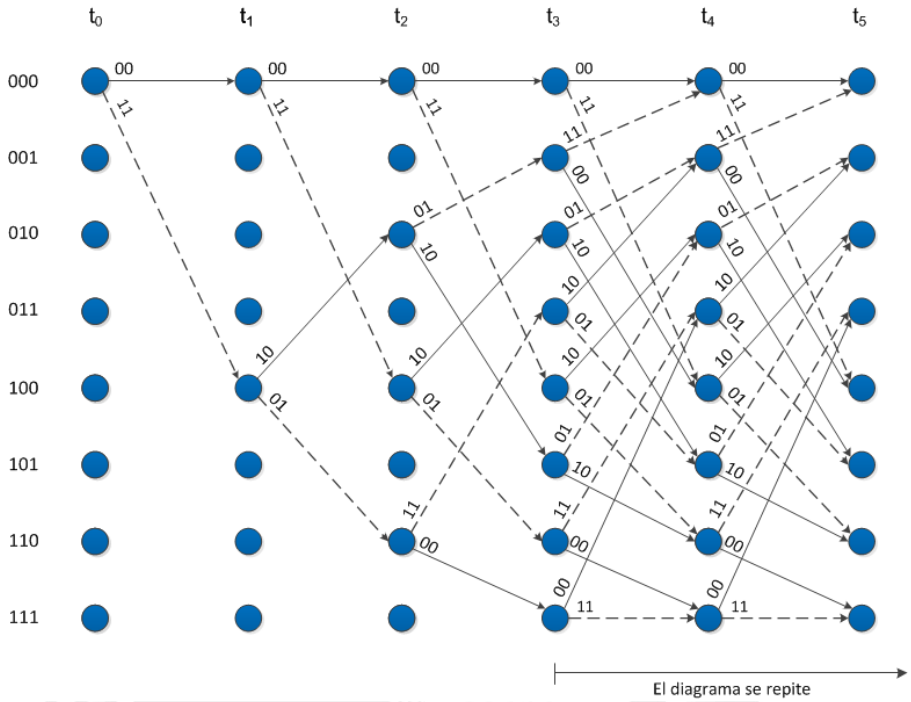


Figura 3.7: Diagrama de Trellis de Codificador Convolucional

El circuito diseñado para el Codificador Convolucional consta fundamentalmente de una máquina de estados finita (FSM) de tipo Mealy, basada en el diagrama de estados presentado en la figura 3.6. Las entradas y salidas del circuito se muestran en la figura 3.8.

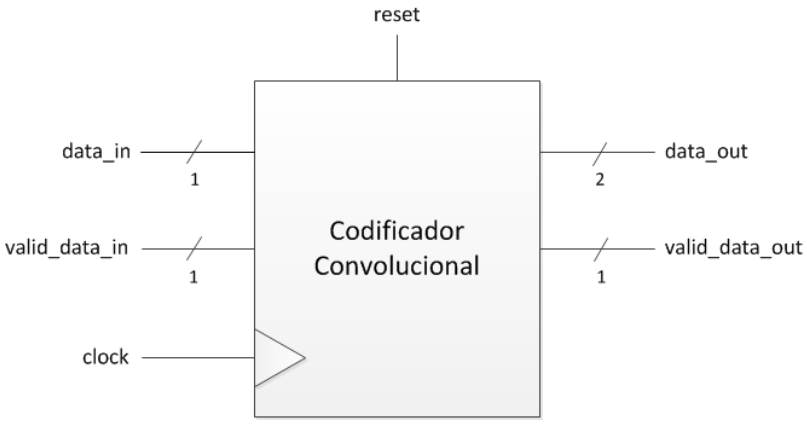


Figura 3.8: Esquema general del bloque Codificador Convolucional

Se pueden observar en la figura las siguientes señales de entrada y salida:

- *Reset*: señal que reinicializa el circuito a sus condiciones iniciales.
- *Clock*: señal de reloj, común a todo el sistema.
- *Data_in*: flujo de datos de entrada, que representa a los bits de información.
- *Valid_data_in*: señal que indica cuándo se recibe una trama válida de datos. Esta señal, además, reinicializa la máquina de estados finita (FSM) cada vez que se activa nuevamente luego de permanecer inactiva.
- *Data_out*: señal de datos de salida, conformada por los dos bits codificados por la FSM.
- *Valid_data_out*: señal que indica cuándo los datos en la salida son válidos.

3.2.1.3. Bloque Intercalador

El bloque Intercalador tiene como función realizar un reordenamiento de los bits que recibe, a fin de proporcionar un método para reducir los errores de ráfaga, los cuales se producen en los datos transmitidos en canales con presencia de ruido.

El Intercalador diseñado emplea dos memorias organizadas como matrices de 1024x4 bits, es decir, cada memoria cuenta con cuatro sub-bancos de memoria de 1024x1 bit. La primera memoria está destinada para el almacenamiento de bits de información (bit 0 ó LSB de los dos bits recibidos), mientras que la segunda almacena los bits de paridad (bit 1 ó MSB de los dos bits recibidos).

El procedimiento de reordenamiento de datos se realiza en dos etapas. La primera consiste en la escritura de los bits en los sub-bancos de memoria. Ésta se produce de forma vertical, tal como se observa en la figura 3.9, donde el índice en cada celda indica el orden en que se escriben los bits. Se puede notar que los primeros 1024 bits de información se almacenan uno por uno en el primer sub-banco de memoria, los siguientes 1024 bits en el segundo sub-banco, y así sucesivamente hasta el cuarto y último sub-banco de memoria. El mismo procedimiento se realiza de manera paralela en el caso de los bits de paridad.

La segunda etapa consta de la lectura de los bits de los sub-bancos de memoria, y es la que determina la manera en cómo se intercalan los datos. Ésta se realiza de forma horizontal y en grupos de dos bits, como se puede apreciar en la figura 3.10, donde el índice en cada celda indica el orden en que se leen los pares de

bits. Se observa que los ocho primeros bits en ser leídos corresponden a aquéllos situados en la fila 1 de los sub-bancos de memoria, los siguientes ocho corresponden a la fila 2 de los sub-bancos, y así sucesivamente hasta la fila 1024 de los sub-bancos de memoria.

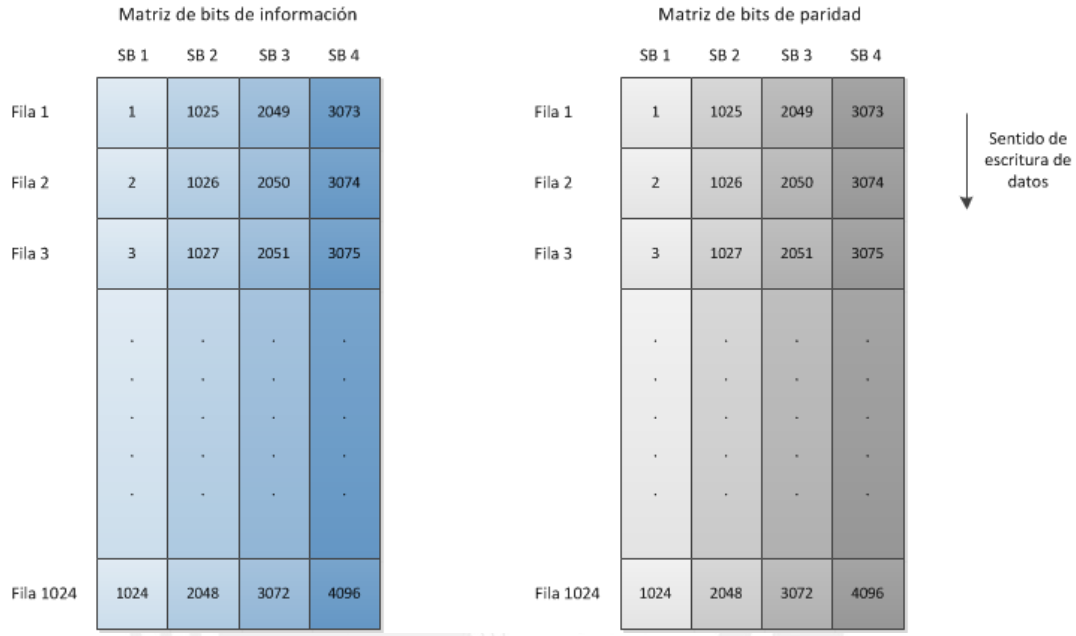


Figura 3.9: Procedimiento de escritura de datos de Bloque Intercalador

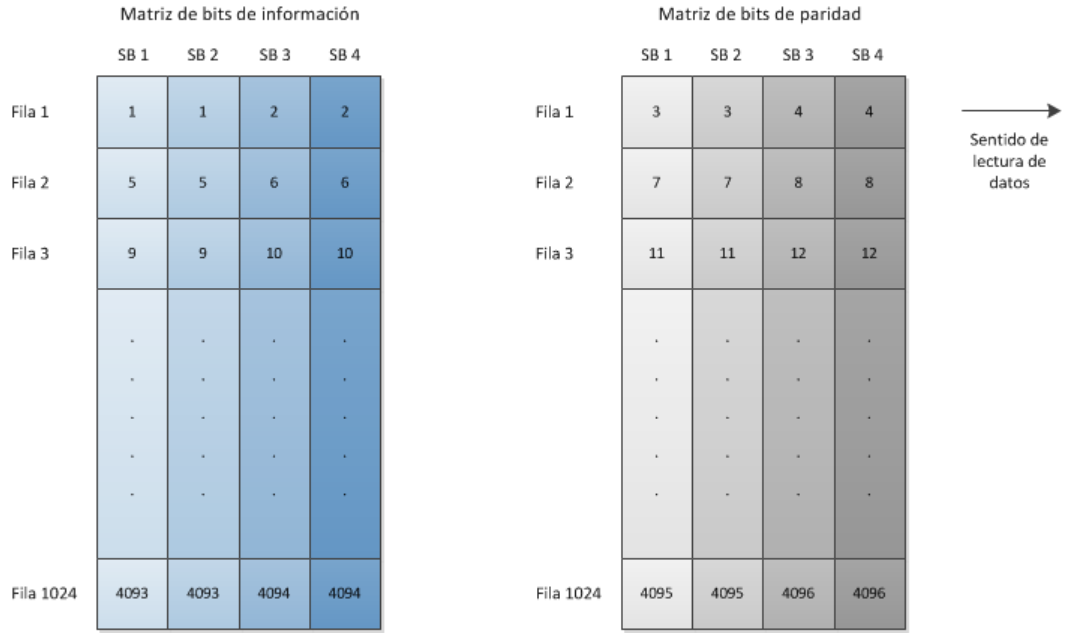


Figura 3.10: Procedimiento de lectura de datos de Bloque Intercalador

El circuito de control diseñado para el Intercalador está basado en una FSM de tres estados, cuyo diagrama de estados se muestra en la figura 3.11. El primer estado, denominado *inactivo*, es aquél donde se inicializa la FSM. Se permanece en este estado hasta que se recibe una nueva trama válida de datos, es decir, cuando se activa la señal *inicio*. El estado *escritura* es aquél donde se realiza la escritura de los bits en los sub-bancos de memoria. En este estado se generan las señales de activación de escritura de los sub-bancos de memoria, para lo cual se emplean contadores de fila y columna. Se permanece en este estado hasta que se completa la escritura de los sub-bancos, esto es, cuando se activa la señal *fin_escritura*. El tercer estado, denominado *lectura*, realiza la lectura, en grupos de dos bits, de los datos almacenados en los sub-bancos de memoria. Para ello, utiliza también contadores de fila y columna para determinar los bits que se leen en cada ciclo. Una vez que se completa la lectura de datos, se activa la señal *fin_lectura*, y se retorna al estado *inactivo*.

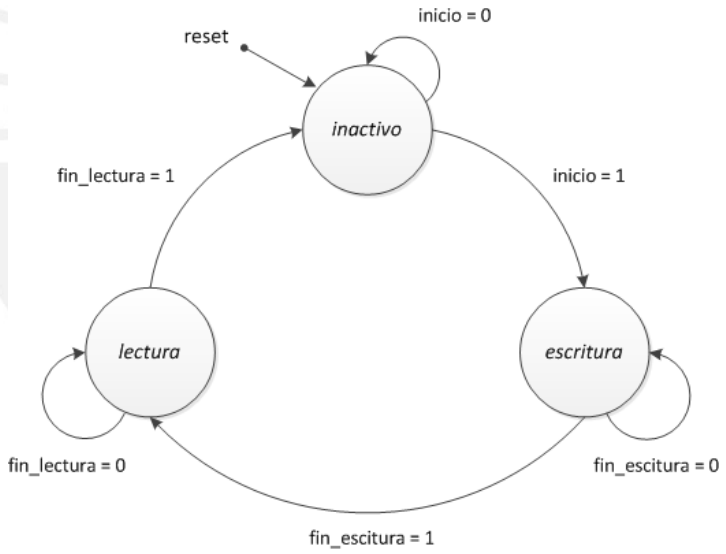


Figura 3.11: Diagrama de estados de bloque Intercalador

En la figura 3.12 se presenta una vista esquemática general del bloque Intercalador diseñado, en el cual se observan las siguientes entradas y salidas:

- *Reset*: señal que reinicializa el circuito a sus condiciones iniciales.
- *Clock*: señal de reloj, común a todo el sistema.
- *Data_in*: señal de datos de entrada de dos bits, donde el primer bit (MSB) representa al bit de paridad y el segundo bit (LSB) al bit de información.

- *Valid_data_in*: señal que indica cuándo se recibe una trama válida de datos. Esta señal, además, reinicializa la máquina de estados finita (FSM) cada vez que se activa nuevamente luego de permanecer inactiva.
- *Data_out*: señal de datos de salida de dos bits, conformada por los pares de bits leídos de los sub-bancos de memoria.
- *Valid_data_out*: señal que indica cuándo los datos en la salida son válidos.

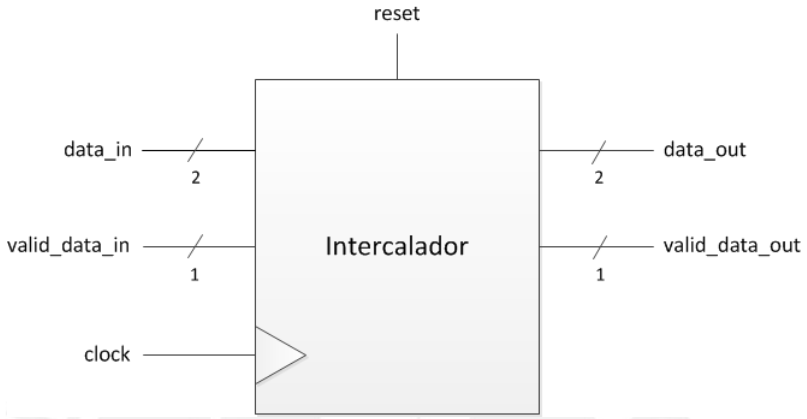


Figura 3.12: Esquema general del bloque Intercalador

Es importante mencionar que el bloque diseñado difiere parcialmente de las especificaciones del estándar IEEE 1901, puesto que el bloque Intercalador del estándar recibe cuatro bits de un Codificador Turbo Convolucional, en lugar de los dos bits que se recibe del Codificador Convolucional en el diseño propuesto. Sin embargo, el modo de escritura y lectura de datos es similar, con la diferencia de que la lectura no se realiza en un orden consecutivo de filas, sino en un orden predeterminado.

3.2.1.4. Bloque Codificador QPSK

El bloque Codificador QPSK (*Quadrature Phase Shift Keying)* se encarga de generar los símbolos de modulación para cada una de las subportadoras OFDM. Estos símbolos son de la forma $s_k = I_k + jQ_k$, donde I_k es denominado componente en fase y Q_k componente en cuadratura, y son generados a partir de los grupos de dos bits que se reciben del bloque Intercalador.

La correspondencia entre los bits de entrada y los símbolos generados, expresados en formato decimal, se presenta en la tabla 3.1.

Tabla 3.1: Correspondencia de símbolos de modulación QPSK

Bits de entrada		Símbolo de modulación	
b_1	b_0	I_k	Q_k
0	0	-1	-1
0	1	-1	$+1$
1	0	$+1$	-1
1	1	$+1$	$+1$

De forma gráfica, esta relación de equivalencia entre los símbolos de modulación y los bits de entrada se expresa mediante el diagrama de constelación presentado en la figura 3.13. Como se observa en el gráfico, los puntos en el plano complejo I-Q equidistan del origen de coordenadas, y presentan un desfase de 90° respecto de sus puntos vecinos.

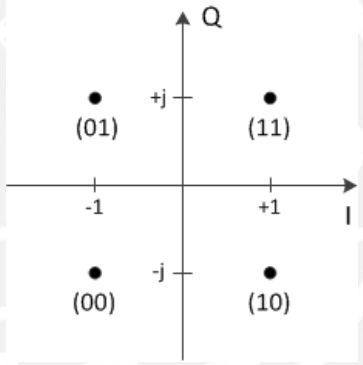


Figura 3.13: Diagrama de constelación QPSK

El circuito diseñado para el Codificador QPSK se muestra en la figura 3.14, donde se observa que los componentes de fase y cuadratura son codificados en palabras de 16 bits, expresadas en complemento a 2.

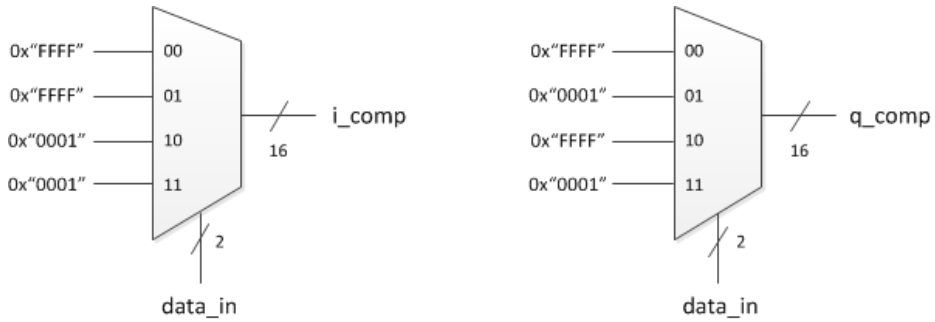


Figura 3.14: Circuito lógico de Codificador QPSK

A continuación, se presenta una vista esquemática general del bloque Codificador QPSK diseñado (figura 3.15).

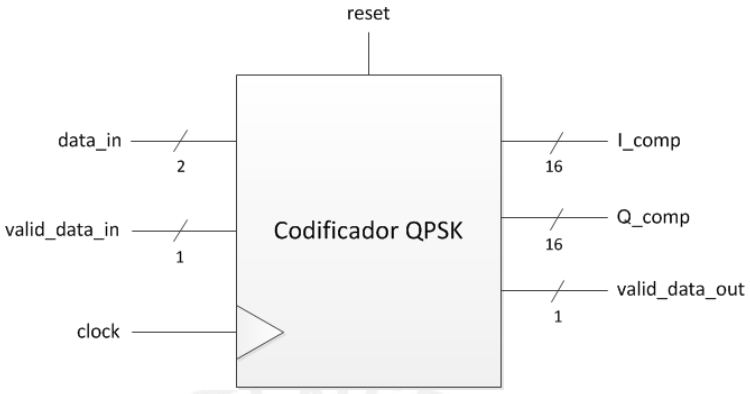


Figura 3.15: Esquema general del bloque Codificador QPSK

Las entradas y salidas del circuito son:

- *Reset*: señal que reinicializa el circuito.
- *Clock*: señal de reloj, común a todo el sistema, que sincroniza los registros ubicados en las salidas del circuito.
- *Data_in*: señal de datos de entrada de dos bits, la cual se codifica según la correspondencia mostrada en la tabla 3.1.
- *Valid_data_in*: señal que indica cuándo se recibe una trama válida de datos.
- *I_comp*: señal de salida de 16 bits, correspondiente a la componente en fase del símbolo de modulación QPSK.
- *Q_comp*: señal de salida de 16 bits, correspondiente a la componente en cuadratura del símbolo de modulación QPSK.
- *Valid_data_out*: señal que indica cuándo los datos en la salida son válidos.

Es importante precisar que la codificación QPSK es sólo uno de los tipos de mapeo o codificación que propone el estándar IEEE 1901 para los datos de carga útil (*payload data*). En comparación con otros métodos de codificación, éste es uno de los más robustos e inmunes al ruido, puesto que los símbolos en la constelación presentan un mayor espaciamiento y un mayor desfase. Sin embargo, presenta la desventaja de ofrecer una menor tasa de transmisión en comparación con otros formatos de codificación de mayor orden, dado que transmite una menor cantidad de bits por cada símbolo generado.

3.2.1.5. Bloque Conversor Serial-Paralelo

El bloque Conversor Serial-Paralelo se encarga de expresar en forma paralela el flujo de 4096 datos que recibe en cada trama válida de datos, esto es, genera 4096 salidas paralelas a partir de un flujo de datos de entrada con la misma cantidad de muestras. Este bloque es parte del núcleo de propiedad intelectual (*IP core*) Fast Fourier Transform v8.0 de Xilinx® que se utiliza en el transmisor-receptor, y se emplea internamente para el ingreso de los datos que son procesados en dicho *IP core*.

3.2.1.6. Bloque Transformada Inversa Rápida de Fourier (IFFT)

El bloque Transformada Inversa Rápida de Fourier (IFFT) constituye el núcleo principal de la modulación OFDM. Su función es generar la señal OFDM en el tiempo a partir de las muestras espectrales de los símbolos QPSK que recibe, esto es, convierte una señal frecuencial compuesta por símbolos QPSK en una secuencia temporal.

La IFFT empleada es de 4096 puntos, lo que significa que recibe 4096 componentes de frecuencia, y genera la misma cantidad de muestras en el tiempo. Por tanto, el término N toma el valor de 4096 en la ecuación 2.1 presentada en el capítulo 2. De este modo, la operación realizada por la IFFT implementada ($x(n)$) está descrita por la ecuación 3.2, donde n es el número de muestra en el tiempo, k el número del componente frecuencial y $X(k)$ el valor del k-ésimo componente frecuencial.

$$x(n) = \frac{1}{4096} \sum_{k=0}^{4095} X(k) e^{\frac{jnk2\pi}{4096}} \quad , \; n = 0, \dots, 4095 \qquad (\text{Ec. } 3.2)$$

Para la implementación de la IFFT se emplea el núcleo de propiedad intelectual (*IP core*) Fast Fourier Transform v8.0 de Xilinx®, cuya hoja de datos se presenta en [17]. Este *IP* core no sólo efectúa el cálculo de la IFFT, sino que también realiza la conversión serial-paralela para los datos de entrada y la conversión paralela-serial para los datos de salida. A continuación, se presentan los parámetros seleccionados para su configuración:

- Tipo de implementación: *Pipelined Streaming I/O*. Esta arquitectura permite un procesamiento continuo de datos, y por tanto es la que permite la mayor tasa de

datos entre los diferentes modos de implementación. Sin embargo, es la arquitectura que requiere la mayor cantidad de recursos.

- Tamaño de la transformada: 4096 puntos.
- Formato de datos: punto fijo. Tanto los datos de entrada como los de salida se expresan en formato de punto fijo.
- Tamaño de bus de datos de entrada: 16. Tanto los componentes de fase como los de cuadratura son de 16 bits.
- Tamaño de factor de fase: 16. A mayor tamaño del factor de fase, mayor inmunidad al ruido y mayor cantidad de recursos utilizados.
- Tipo de escalamiento: *unscaled* (no escalado). Permite una mayor precisión de los datos en la salida que el modo *scaled* (escalado).
- Modo de redondeo: *convergent rounding* (redondeo convergente). Permite una mayor precisión en los datos que el modo *truncation* (truncado).
- Ordenamiento de las salidas: orden natural. Esto implica que los datos en la salida son mostrados en orden consecutivo desde el primero hasta el último.
- Inserción de prefijo cíclico: sí.
- Cantidad de etapas que usan *Block RAM*: 7. Se utiliza dicha cantidad de bloques de memoria dedicados.

En las siguientes líneas, se presentan las entradas y salidas empleadas del núcleo de propiedad intelectual FFT v8.0 (no se muestran las señales que no han sido utilizadas), así como los valores o señales que se les asignan.

- *s_axis_config_tdata*: especifica el tamaño del prefijo cíclico y el tipo de transformada (FFT o IFFT). Se le asigna el valor decimal de 1252, el cual indica que el prefijo cíclico es de dicha longitud y que se selecciona el modo IFFT.
- *s_axis_config_tvalid*: indica que la señal *s_axis_config_tdata* es válida. Se le asigna el valor constante ‘1’.
- *s_axis_data_tdata*: señal de datos de entrada. Se le asigna la señal mostrada en la figura 3.16.
- *s_axis_data_tvalid*: señal de entrada que indica que la trama de datos ingresada es válida. Se le asigna la señal *valid_data_in*, como se observa en la figura 3.16.
- *s_axis_data_tlast*: señal de entrada que se activa externamente cuando se envía la última muestra de una trama. Esta señal no se emplea, y por tanto se le asigna el valor constante ‘0’.

- *m_axis_data_tready*: señal de entrada que indica que el circuito conectado en la salida (circuito esclavo) está listo para recibir datos. Se le asigna el valor constante '1'.
- *aclk*: señal de entrada de reloj. Se le asigna la señal *clock*, la cual es la señal de reloj del sistema.
- *m_axis_data_tdata*: señal de datos de salida. Se asigna a la señal mostrada en la figura 3.16.
- *m_axis_data_tvalid*: señal que indica que la trama de datos de salida es válida. Se asigna a la señal *valid_data_out*, como se observa en la figura 3.16.

En la figura 3.16 se muestra un diagrama esquemático del bloque IFFT implementado. Se observa que las señales de entrada *I_comp* y *Q_comp* ingresan inicialmente a bloques denominados *Shift*, para luego ser concatenadas a una sola señal que es ingresada a la entrada *s_axis_data_tdata* del *IP Core* FFT v8.0. Estos bloques *Shift* realizan un desplazamiento de 10 bits hacia la izquierda de sus valores de entrada, a fin de que los últimos bits que contienen información se desplacen a posiciones más significativas. Además, se aprecia que la señal de salida *m_axis_data_tdata* del *IP Core* es inicialmente dividida en dos señales, las cuales representan las componentes real e imaginaria de la muestra de salida. Estas señales son procesadas por bloques denominados *Division*, los cuales dividen sus valores de entrada entre 4096 (desplazamiento de 12 bits a la izquierda) y toman los 16 bits menos significativos del resultado. La finalidad de este bloque es realizar la división de las muestras entre el número 4096, como se muestra en la ecuación 3.2, puesto que el *IP Core* no realiza la división internamente. Finalmente, las salidas de los bloques *Division* son concatenadas para formar la señal *data_out* de salida.

Las entradas y salidas generales del circuito de la figura 3.16 son:

- *Clock*: señal de reloj, común a todo el sistema.
- *I_comp*: señal de datos de entrada de 16 bits, correspondiente a las componentes en fase de los símbolos QPSK.
- *Q_comp*: señal de datos de entrada de 16 bits, correspondiente a las componentes en cuadratura de los símbolos QPSK.
- *Valid_data_in*: señal que indica cuándo se recibe una trama válida de datos.
- *Data_out*: señal de datos de salida de 32 bits, correspondiente a las muestras temporales que genera la IFFT.

- *Valid_data_out*: señal que indica cuándo los datos en la salida son válidos.

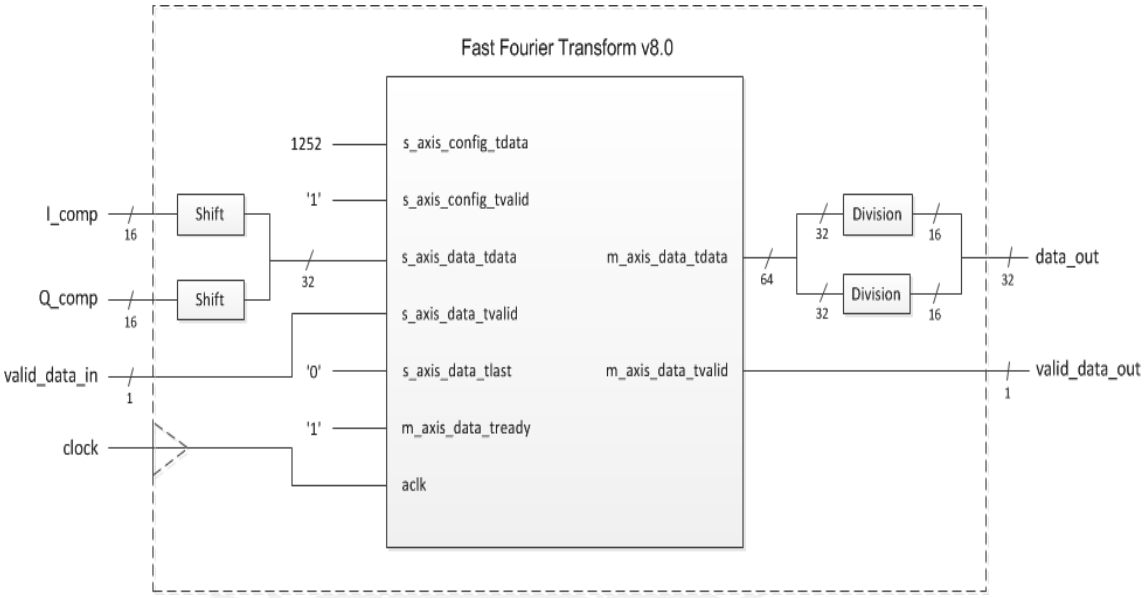


Figura 3.16: Diagrama esquemático del bloque IFFT

Es importante mencionar que el bloque IFFT no emplea los ángulos de fase que propone el estándar IEEE 1901 para cada una de las subportadoras durante la transmisión de un símbolo OFDM, debido a que el *IP Core* empleado no permite la introducción de los mismos. Por tanto, todas las subportadoras se consideran en fase, es decir, presentan un ángulo de fase de 0°.

3.2.1.7. Bloque Conversor Paralelo-Serial

El bloque Conversor Paralelo-Serial se encarga de expresar los 4096 datos que recibe de forma paralela en un flujo serial con la misma cantidad de muestras, esto es, construye una trama serial de 4096 muestras a partir de los valores de sus 4096 entradas. Este bloque es parte del núcleo de propiedad intelectual (*IP core*) Fast Fourier Transform v8.0 de Xilinx® que se utiliza en el transmisor-receptor, y se emplea internamente para la salida de los datos que han sido procesados en dicho *IP core*.

3.2.1.8. Bloque de Inserción de Prefijo Cíclico

El bloque de Inserción de Prefijo Cíclico es el encargado de copiar un grupo de muestras que se encuentran al final del símbolo OFDM y colocarlas al inicio del mismo.

Su finalidad es generar símbolos OFDM periódicos, de modo que la convolución lineal entre la señal transmitida y el canal se convierta en una convolución circular, para así facilitar la recuperación de la señal en el receptor [18].

La longitud del prefijo cíclico se determina de tal modo que sea mayor a la máxima dispersión de retardo de canal (*maximum delay spread*) [18], con la finalidad de que los retardos presentados en el canal no provoquen sucesivas interferencias en los símbolos OFDM. El estándar IEEE 1901 define un prefijo cíclico de 1252 muestras, de tal manera que el símbolo extendido generado es de 5348 muestras, como se observa en la figura 3.17.

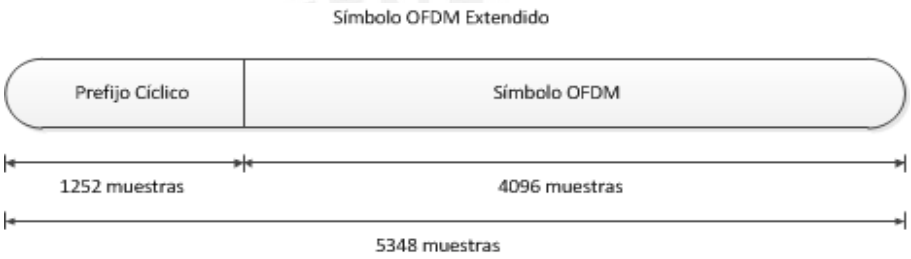


Figura 3.17: Diagrama de símbolo OFDM extendido

La inserción del prefijo cíclico es realizada internamente por el núcleo de propiedad intelectual (*IP core*) Fast Fourier Transform v8.0 de Xilinx® que se emplea para el cálculo de la IFFT.

3.2.2. Descripción de la arquitectura del receptor

En la figura 3.18 se muestra un esquema del receptor implementado, donde se especifica el número de bits de datos que emplea cada módulo que lo compone.

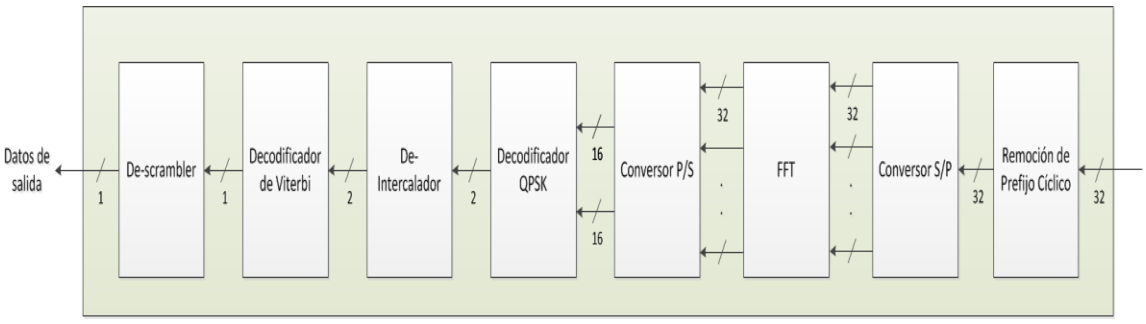


Figura 3.18: Arquitectura del receptor

Cada símbolo OFDM recibido consta de un flujo de 5348 muestras de 32 bits, y es inicialmente procesado por el bloque de Remoción de prefijo cíclico, el cual elimina las primeras 1252 muestras correspondientes al prefijo cíclico.

Luego, un conversor serial-paralelo expresa en forma paralela el flujo de 4096 muestras que recibe, de modo que forma las 4096 entradas paralelas que requiere el módulo FFT.

A continuación, el módulo de Transformada Rápida de Fourier (FFT) realiza la demodulación OFDM propiamente dicha, dando como resultado una señal frecuencial compuesta por 4096 símbolos QPSK.

En seguida, un conversor paralelo-serial convierte los datos paralelos de la salida de la FFT en dos flujos de palabras de 16 bits, correspondientes a los componentes en fase (I) y componentes en cuadratura (Q) de los símbolos QPSK generados por la FFT.

Luego, el decodificador QPSK efectúa la conversión de los símbolos que recibe en dos bits de información, según la relación de correspondencia mostrada anteriormente en la tabla 3.1.

A continuación, el bloque De-intercalador realiza un reordenamiento de los datos, de modo inverso al efectuado por el bloque Intercalador en el transmisor.

Seguidamente, el Decodificador de Viterbi analiza los dos bits recibidos, de tal manera que detecta y corrige bits erróneos, y genera un único flujo de bits de información.

Finalmente, el *de-scrambler* modifica los valores de los bits de forma pseudo-aleatoria, al igual que el *scrambler* en el transmisor, de modo que permite la recuperación de la información inicialmente transmitida.

3.2.2.1. Bloque de Remoción de Prefijo Cíclico

El bloque de Remoción de Prefijo Cíclico es el encargado de descartar las primeras 1252 muestras del símbolo OFDM recibido, correspondientes al prefijo cíclico añadido en el transmisor. Para ello, desactiva de señal de datos válidos (*valid_out*)

durante la duración del prefijo cíclico, lo cual se efectúa mediante una FSM cuyo diagrama de estados se presenta en la figura 3.19.

El primer estado en el diagrama mostrado, denominado *inactivo*, es aquél donde se inicializa la FSM. Se permanece en este estado hasta que se recibe una nueva trama válida de datos, es decir, cuando se activa la señal *valid_in*. El estado *remociónCP* es aquél donde se elimina el prefijo cíclico, y por tanto desactiva la señal de datos válidos (*valid_out*). Este estado emplea un contador de muestras recibidas, que activa la señal *fin_CP* cuando se iguala a 1252, es decir, cuando recibe la última muestra correspondiente al prefijo cíclico. El tercer estado, denominado *data_válida*, activa la señal *valid_out* hasta que recibe la última muestra del símbolo OFDM, para luego retornar al estado *inactivo*.

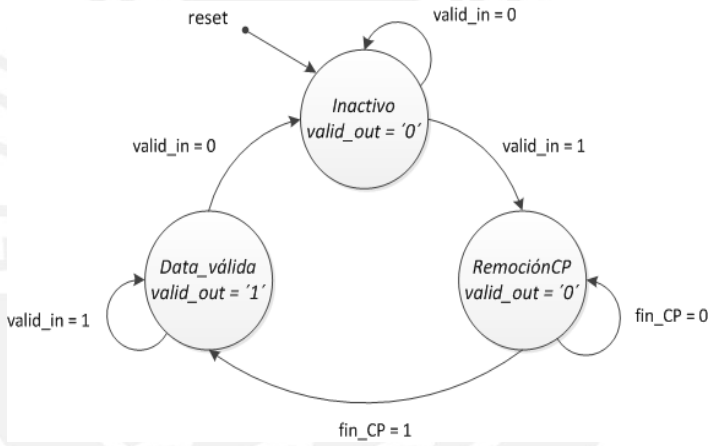


Figura 3.19: Diagrama de estados de bloque de Remoción de Prefijo Cíclico

En la figura 3.20 se presenta una vista esquemática general del bloque de Remoción de Prefijo Cíclico diseñado, en el cual se observan las siguientes entradas y salidas:

- *Reset*: señal que reinicializa el circuito a sus condiciones iniciales.
- *Clock*: señal de reloj, común a todo el sistema.
- *Data_in*: señal de datos de entrada de 32 bits, que conforman el símbolo OFDM extendido que se recibe.
- *Valid_in*: señal que indica cuándo se recibe una trama válida de datos.
- *Data_out*: señal de datos de salida de 32 bits, que conforman el símbolo OFDM recuperado.
- *Valid_out*: señal que indica cuándo los datos en la salida son válidos.

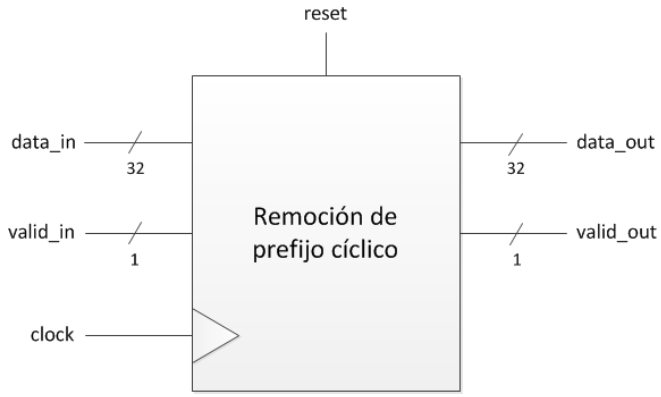


Figura 3.20: Esquema general del bloque de Remoción de Prefijo Cíclico

3.2.2.2. Bloque Conversor Serial-Paralelo

El bloque Conversor Serial-Paralelo del receptor es similar al empleado en el transmisor, descrito en la sección 3.2.1.5.

3.2.2.3. Bloque Transformada Rápida de Fourier (FFT)

El bloque Transformada Rápida de Fourier (FFT) constituye el núcleo principal de la demodulación OFDM. Su función es generar, a partir de las muestras de la señal OFDM en el tiempo, los símbolos QPSK para cada una de las subportadoras que conforman la señal en frecuencia. Dicho de otro modo, convierte una secuencia temporal en una señal frecuencial compuesta por símbolos QPSK.

La FFT empleada es de 4096 puntos, lo que significa que recibe 4096 muestras en el tiempo, y genera la misma cantidad de componentes de frecuencia. Por tanto, el término N toma el valor de 4096 en la ecuación 2.2 presentada en el capítulo 2. De esta manera, la operación realizada por la FFT implementada ($X(k)$) está descrita por la ecuación 3.3, donde k es el número del componente frecuencial, n el número de muestra en el tiempo y $x(n)$ el valor de la n-ésima muestra en el tiempo.

$$X(k) = \sum_{n=0}^{4095} x(n) e^{-\frac{jnk2\pi}{4096}} \quad , \; k = 0, \dots, 4095 \qquad (\text{Ec. } 3.3)$$

Para la implementación de la FFT se emplea el núcleo de propiedad intelectual (*IP core*) Fast Fourier Transform v8.0 de Xilinx®, cuya hoja de datos se presenta en

[17]. Este *IP core* no sólo efectúa el cálculo de la FFT, sino que también realiza la conversión serial-paralela para los datos de entrada y la conversión paralela-serial para los datos de salida.

Los parámetros seleccionados para la configuración del *IP core* son los mismos que los empleados para el bloque IFFT en el transmisor (sección 3.2.1.6), con la excepción de que no se habilita la inserción de prefijo cíclico.

A continuación, se presentan las entradas y salidas empleadas del núcleo de propiedad intelectual FFT v8.0 (no se muestran las señales que no han sido utilizadas), así como los valores o señales que se les asignan.

- *s_axis_config_tdata*: especifica el tipo de transformada (FFT o IFFT). Se le asigna el valor decimal de 1, el cual indica que se selecciona el modo FFT.
- *s_axis_config_tvalid*: indica que la señal *s_axis_config_tdata* es válida. Se le asigna el valor constante '1'.
- *s_axis_data_tdata*: señal de datos de entrada. Se le asigna la señal *data_in,* como se muestra en la figura 3.21.
- *s_axis_data_tvalid*: señal de entrada que indica que la trama de datos ingresada es válida. Se le asigna la señal *valid_data_in*, como se observa en la figura 3.21.
- *s_axis_data_tlast*: señal de entrada que se activa externamente cuando se envía la última muestra de una trama. Esta señal no se emplea, y por tanto se le asigna el valor constante '0'.
- *m_axis_data_tready*: señal de entrada que indica que el circuito conectado en la salida (circuito esclavo) está listo para recibir datos. Se le asigna el valor constante '1'.
- *aclk*: señal de entrada de reloj. Se le asigna la señal *clock*, la cual es la señal de reloj del sistema.
- *m_axis_data_tdata*: señal de datos de salida. Se asigna a la señal mostrada en la figura 3.21.
- *m_axis_data_tvalid*: señal que indica que la trama de datos de salida es válida. Se asigna a la señal *valid_data_out*, como se observa en la figura 3.21.

En la figura 3.21 se muestra un diagrama esquemático del bloque FFT implementado. Se observa que la señal de salida *m_axis_data_tdata* del *IP Core* es inicialmente dividida en dos señales, las cuales representan las componentes en fase

(I) y en cuadratura (Q) del símbolo QPSK generado. Estas señales son procesadas con el fin de reducir su longitud de 32 bits a 16 bits, de modo inverso al procedimiento que se realizó en el bloque IFFT del transmisor para la adaptación de las señales *I_comp* y *Q_comp* de 16 a 32 bits. Para ello, se emplean inicialmente los bloques denominados *Redondeo*, los cuales toman los 16 bits menos significativos y redondean la palabra resultante a sus 6 bits más significativos, de modo que los bits restantes son ceros. Luego, los bloques *Shift* realizan un desplazamiento de 10 bits hacia la derecha de los valores de cada señal, de modo que finalmente genera las salidas *I_comp* y *Q_comp* de 16 bits.

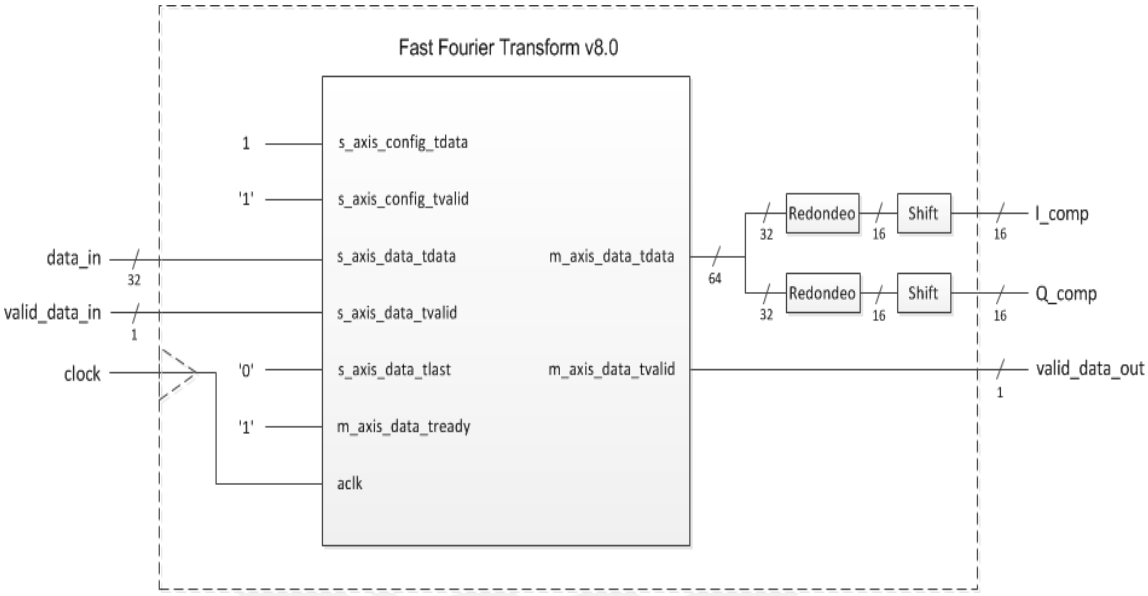


Figura 3.21: Diagrama esquemático del bloque FFT

Las entradas y salidas generales del circuito mostrado son:

- *Clock*: señal de reloj, común a todo el sistema.
- *Data_in*: señal de datos de entrada de 32 bits, correspondiente a las muestras temporales del símbolo OFDM recibido.
- *Valid_data_in*: señal que indica cuándo se recibe una trama válida de datos.
- *I_comp*: señal de datos de salida de 16 bits, correspondiente a las componentes en fase de los símbolos QPSK generados.
- *Q_comp*: señal de datos de salida de 16 bits, correspondiente a las componentes en cuadratura de los símbolos QPSK generados.
- *Valid_data_out*: señal que indica cuándo los datos en la salida son válidos.

3.2.2.4. Bloque Conversor Paralelo-Serial

El bloque Conversor Paralelo-Serial del receptor es similar al empleado en el transmisor, descrito en la sección 3.2.1.7.

3.2.2.5. Bloque Decodificador QPSK

El bloque Decodificador QPSK (*Quadrature Phase Shift Keying)* realiza el proceso inverso al efectuado por el Codificador QPSK en el transmisor, esto es, convierte los símbolos QPSK recibidos en dos bits de información. La relación de correspondencia entre los bits generados y los símbolos recibidos es la misma que la presentada en la tabla 3.1.

A continuación, se presenta una vista esquemática general del bloque Decodificador QPSK diseñado (figura 3.22).

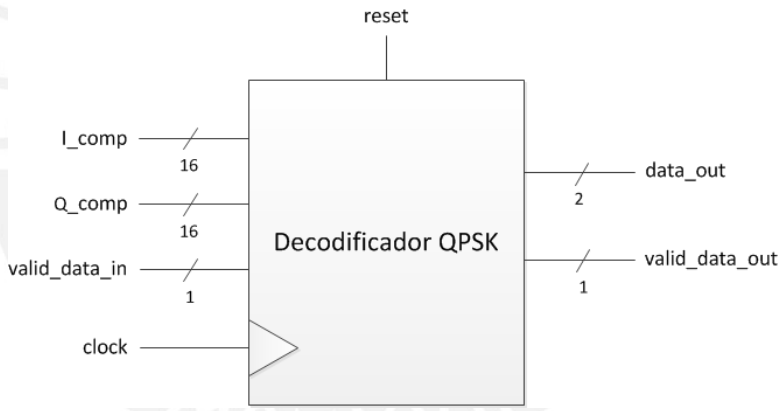


Figura 3.22: Esquema general del bloque Decodificador QPSK

Se observa que las entradas y salidas del bloque son:

- *Reset*: señal que reinicializa el circuito.
- *Clock*: señal de reloj, común a todo el sistema, que sincroniza los registros ubicados en las salidas del circuito.
- *I_comp*: señal de entrada de 16 bits, correspondiente a la componente en fase del símbolo QPSK recibido.
- *Q_comp*: señal de entrada de 16 bits, correspondiente a la componente en cuadratura del símbolo QPSK recibido.
- *Valid_data_in*: señal que indica cuándo se recibe una trama válida de datos.

- *Data_out*: señal de datos de salida de dos bits, la cual se genera según la correspondencia mostrada en la tabla 3.1.
- *Valid_data_out*: señal que indica cuándo los datos en la salida son válidos.

3.2.2.6. Bloque De-Intercalador

El bloque De-Intercalador efectúa un reordenamiento de los datos de modo inverso al realizado por el Intercalador en el transmisor. Esto significa que la primera etapa de escritura de los bits en los sub-bancos de memoria se produce de forma horizontal, como se muestra en la figura 3.10; mientras que la etapa de lectura de los bits de los sub-bancos se realiza de forma vertical, como se indica en la figura 3.9.

El circuito de control empleado se basa en una FSM similar a la utilizada en el Intercalador (figura 3.11), con la diferencia que el estado *escritura* genera señales para una escritura de datos en sentido horizontal, y el estado *lectura* produce señales para una lectura de datos en sentido vertical.

En la figura 3.23 se presenta una vista esquemática general del bloque De-intercalador diseñado.

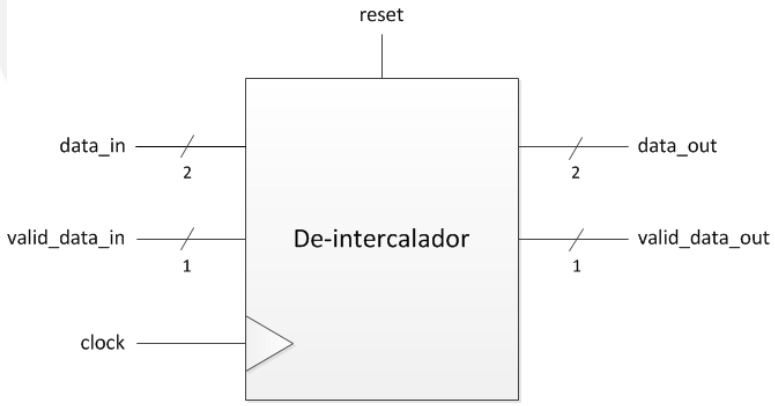


Figura 3.23: Esquema general del bloque De-intercalador

Se observan las siguientes entradas y salidas:

- *Reset*: señal que reinicializa el circuito a sus condiciones iniciales.
- *Clock*: señal de reloj, común a todo el sistema.
- *Data_in*: señal de datos de entrada de dos bits, que corresponde a los bits generados por el Decodificador QPSK.

- *Valid_data_in*: señal que indica cuándo se recibe una trama válida de datos. Esta señal, además, reinicializa la FSM cada vez que se activa nuevamente luego de permanecer inactiva.
- *Data_out*: señal de datos de salida de dos bits, conformada por los pares de bits leídos de los sub-bancos de memoria.
- *Valid_data_out*: señal que indica cuándo los datos en la salida son válidos.

3.2.2.7. Bloque Decodificador de Viterbi

El bloque Decodificador de Viterbi es el encargado de la detección y corrección de errores de los datos recibidos. Para ello, emplea el algoritmo de Viterbi para decodificar la secuencia de bits recibida, de modo que recupera los bits de información inicialmente transmitidos. Este algoritmo consiste en localizar el camino de máxima probabilidad (*maximum likelihood*) dentro del diagrama de Trellis generado por el Codificador Convolucional en el transmisor, de manera que selecciona la secuencia de datos más probable [15].

El Decodificador de Viterbi implementado es un decodificador de decisión dura (*hard decision decoder)*, lo que significa que emplea la distancia de *Hamming*[1] como medida de comparación entre la secuencia recibida y la secuencia generada en el diagrama de Trellis.

El diseño realizado para este bloque está basado en el modelo propuesto en [16]. En la figura 3.24 se muestra un diagrama general de la arquitectura diseñada. A continuación, se describen cada uno de los módulos empleados en la arquitectura.

Los módulos *Branch Metric Calculation* (BMC0, …, BMC7) calculan la diferencia entre los dos bits recibidos y los probables dos bits que debieron recibirse en cada estado del diagrama de Trellis de la figura 3.7, por cada posible transición de estado. Dicha diferencia se codifica asignando los siguientes valores o ‘costos’:

“00”: No hay diferencia entre los dos bits recibidos y los que debieron recibirse
“01”: Hay un bit de diferencia entre dichos pares de bits
“10”: Hay dos bits de diferencia entre dichos pares de bits

[1] La distancia de *Hamming* entre dos palabras es el número de bits en que difieren una de la otra.

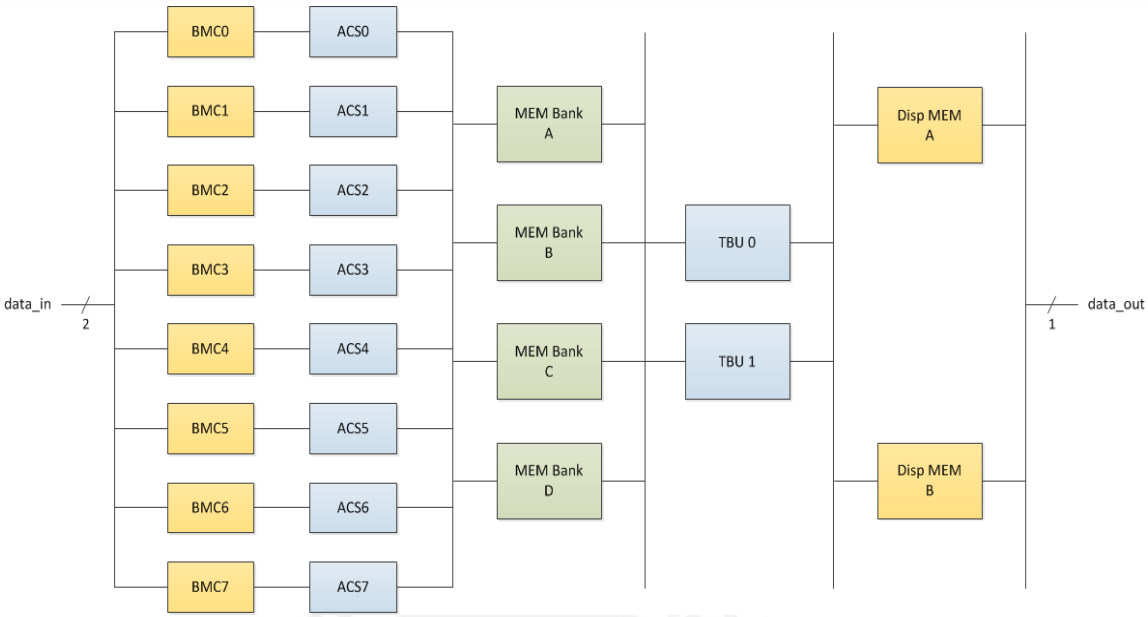


Figura 3.24: Arquitectura del Decodificador de Viterbi

Por ejemplo, el módulo BMC1 se encarga del cálculo de las diferencias entre los bits recibidos y los que debieron recibirse en el estado “001” del diagrama de Trellis. Estos últimos se muestran nuevamente en la figura 3.25, donde la línea continua indica la transición dada por el bit ‘0’ en el diagrama de Trellis, y las líneas punteadas indican la transición dada por el bit ‘1’. Por lo tanto, si el dato recibido es “10”, se asigna el costo de “00” para la transición dada por el bit ‘0’, puesto que los bits recibidos y los bits que debieron recibirse son iguales; del mismo modo, se asigna el costo de “10” para la transición dada por el bit ’1’, pues los bits son totalmente distintos.

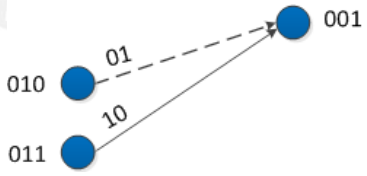


Figura 3.25: Probables bits recibidos para el estado “001” del diagrama de Trellis

Los módulos *Add-Compare-Select* (ACS0, …, ACS7) realizan, en el orden mostrado, las siguientes funciones:

- Adicionan los costos por transición calculados por los módulos BMC, llamados *branch cost* o BC, a los costos acumulados por transición, llamados *path cost* o PC, en cada estado del diagrama de Trellis.

- Comparan los nuevos costos acumulados por transición (BC + PC), en cada estado.
- Seleccionan la transición que produce el menor costo acumulado en cada estado.

Considerando el ejemplo anterior, si el costo acumulado en el estado anterior a la transición dada por el bit '1' es 5, es decir, el costo del camino del bit '1' (*path cost 1 ó PC1)* es 5, y el costo del camino del bit '0' ó PC0 es 6; entonces el módulo ACS1 selecciona la transición dada por el bit '0', puesto que su nuevo costo acumulado es 6 (PC0 + BC0 = 6), y es menor al nuevo costo acumulado del camino del bit '1', el cual es 7 (PC1 + BC1 = 7). Esto se ilustra en la figura 3.26.

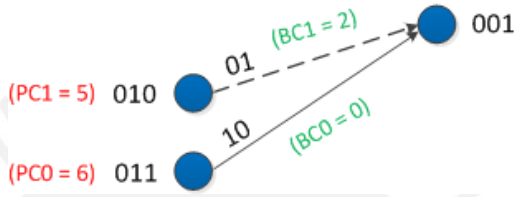


Figura 3.26: Ejemplo de cálculo de costos en el módulo ACS1

Los módulos *MEM Bank* son bancos de memoria en los cuales se almacenan los resultados de las transiciones seleccionadas en cada uno de los módulos ACS. Esto significa que los bancos de memoria son de 8 bits, puesto que cada banco debe almacenar los resultados generados por los ocho módulos ACS en cada ciclo de reloj. Los cuatro bancos (MEM Bank A, …, MEM Bank D) son de 8 registros, y su escritura se realiza en modo *round robin*, esto es, se emplea en primer lugar el banco A, en segundo lugar el B, en tercer lugar el C, en cuarto lugar el D, luego nuevamente el banco A, y así sucesivamente.

Los módulos *Trace Back Unit* (TBU 0 y TBU 1) realizan un recorrido en sentido inverso en el diagrama de Trellis, de manera que seleccionan el camino de máxima probabilidad. Para ello, realizan lecturas de los bancos de memoria de modo inverso al que fueron escritos, es decir, del último al primer valor almacenado.

En la figura 3.27 se muestra el modo de operación de los módulos TBU. Se observa que el primer TBU, el cual hace la operación de rastreo o *trace back* en el banco de memoria Y, inicia su trayectoria en el estado "000" del diagrama de Trellis. Luego, continúa su recorrido en base a los datos que lee del banco de memoria, los cuales indican qué transición debe seleccionar en cada estado. Su trayecto culmina cuando lee el octavo y último dato del banco de memoria que emplea. A continuación,

el segundo TBU, el cual hace la operación de rastreo o *trace back* en el banco de memoria X, continúa el recorrido en el diagrama de Trellis desde el estado en que el primer TBU culminó su trayectoria, hasta que lee el octavo y último dato del banco de memoria. Sin embargo, sólo el recorrido realizado por el segundo TBU es considerado como válido, puesto que según el algoritmo de Viterbi se asume que sólo los estados siguientes a la primera operación de *trace back* son válidos [15].

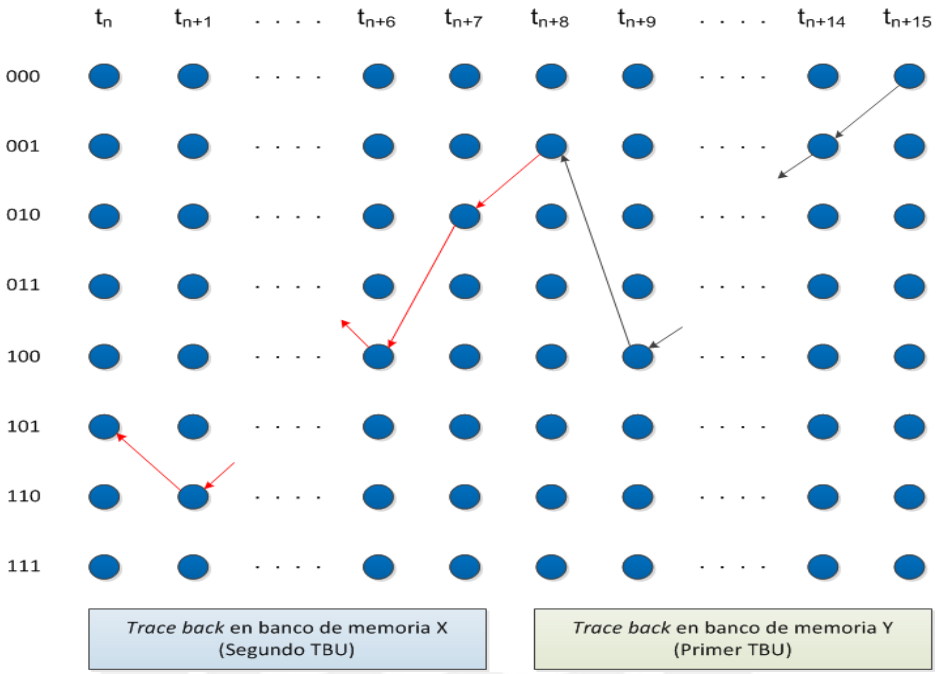


Figura 3.27: Modo de operación de los módulos TBU

Las transiciones válidas de los módulos TBU permiten detectar la secuencia de bits de información de máxima probabilidad. Estos bits son almacenados temporalmente en bancos de memoria denominados *Disp MEM* antes de ser enviados a la salida *data_out* del circuito.

En la figura 3.28 se muestra un diagrama de tiempos general de todos los procesos realizados en el Decodificador de Viterbi, donde las letras hacen referencia a los cuatro bancos de memoria empleados. Además, las flechas indican el sentido que se emplea para el direccionamiento de los bancos de memoria, es decir, si se realiza en sentido directo (del primer hasta el último registro del banco) o en sentido inverso (del último hasta el primer registro del banco). Se observa que los módulos TBU operan de forma paralela, de manera que alternan su función como primer o segundo

TBU para las operaciones de rastreo o *trace back*.

		t_n	t_{n+1}	t_{n+2}	t_{n+3}	t_{n+4}	t_{n+5}	t_{n+6}	t_{n+7}	t_{n+8}	t_{n+9}	t_{n+10}	t_{n+11}	...
Escritura en banco de memoria	→	A	B	C	D	A	B	C	D	A	B	C	D	...
Operación de TBU 0	←			B	A	D	C	B	A	D	C	B	A	...
Operación de TBU 1	←				C	B	A	D	C	B	A	D	C	...
Escritura en Disp MEM 0	←				A		C		A		C		A	...
Escritura en Disp MEM 1	←					B		D		B		D		...
Lectura de Disp MEM 0	→					A		C		A		C		...
Lectura de Disp MEM 1	→						B		D		B		D	...

Figura 3.28: Diagrama de tiempos de Decodificador de Viterbi

A continuación, se presenta una vista esquemática general del Decodificador de Viterbi (figura 3.29).

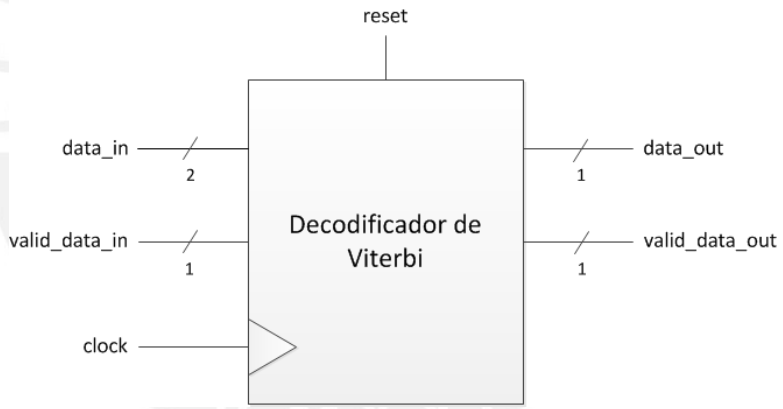


Figura 3.29: Esquema general del bloque Decodificador de Viterbi

Se pueden observar en la figura las siguientes señales de entrada y salida:

- *Reset*: señal que reinicializa el circuito a sus condiciones iniciales.
- *Clock*: señal de reloj, común a todo el sistema.
- *Data_in*: datos de entrada de dos bits, que se recibe del bloque De-Intercalador.
- *Valid_data_in*: señal que indica cuándo se recibe una trama válida de datos.
- *Data_out*: señal de datos de salida, correspondiente a los bits de información decodificados.
- *Valid_data_out*: señal que indica cuándo los datos en la salida son válidos.

3.2.2.8. Bloque De-scrambler

El bloque de-*scrambler* realiza el mismo proceso de aleatorización de datos que el realizado por el *scrambler* en el transmisor, y su circuito lógico es igual al presentado en la figura 3.3. Esto se debe a que la inversa del operador *xor* (o suma de módulo 2) empleado para la aleatorización es el mismo operador *xor*. Por tanto, el esquema general del *scrambler* presentado en la figura 3.4 es el mismo para el de-*scrambler*.

3.3. Diseño en VHDL de la arquitectura

En primer lugar, todos los bloques y diagramas presentados anteriormente fueron descritos mediante el lenguaje de descripción de hardware VHDL. El diseño de los circuitos combinacionales y secuenciales empleados fue realizado a nivel RTL (*register-transfer level*), y la metodología utilizada para la descripción de los mismos se basa en la presentada en [14].

En segundo lugar, la arquitectura descrita es modular, de modo que cada bloque diseñado es independiente de los otros, y puede ser modificado sin afectar el funcionamiento general del sistema implementado. Por ello, se incluyeron las señales *valid_data_in* y *valid_data_out* en cada módulo, las cuales permiten indicar cuándo se reciben o generan datos válidos, y por tanto hacen posible que los módulos contiguos se conecten de forma directa.

Asimismo, los bloques diseñados son portables, a excepción de los utilizados para el cálculo de la IFFT y FFT. Esto significa que pueden ser implementados en dispositivos FPGA de distintas familias y fabricantes.

Por último, se ha tenido especial consideración en la descripción para utilizar la menor cantidad posible de elementos combinacionales entre dos registros, pues ello limita la frecuencia máxima de operación del sistema. Así, entre operaciones que implicaban gran cantidad de recursos combinacionales, se han colocado registros intermedios luego de cada sub-etapa de dichas operaciones.

CAPÍTULO 4:

SIMULACIONES Y RESULTADOS

En el presente capítulo, se presentan las simulaciones que permiten verificar el funcionamiento del sistema diseñado, así como también los resultados obtenidos. En primer lugar, cada bloque que compone el sistema es simulado por separado para validar su funcionamiento frente a diferentes señales de entrada. Luego, los bloques se unen en entidades mayores, y son simulados para verificar el funcionamiento en conjunto. Este procedimiento continúa hasta formar el sistema completo. Posteriormente, se simula la operación del sistema en presencia de distintos modelos del ruido presente en el canal eléctrico. Los resultados obtenidos a partir de las simulaciones realizadas son mostrados en la parte final del capítulo.

4.1. Simulaciones

Las simulaciones de los bloques diseñados fueron realizadas mediante la herramienta ISim del software ISE 14.4 de la compañía Xilinx®. Para ello, se elaboraron bancos de prueba o *test benches* donde se incluyeron estímulos aleatorios para los datos de entrada.

Una de las principales simulaciones efectuadas corresponde a aquélla del sistema completo de transmisión-recepción. Ésta consistió en conectar directamente la salida del transmisor a la entrada del receptor, de modo que la trama ingresada al transmisor debía ser igual a la trama recuperada en el receptor.

El banco de prueba codificado para esta simulación consistió en la generación de una trama aleatoria de 4096 bits para la señal de datos de entrada, y de una señal de datos válidos (*data_in_valid)* activa por un período de 4096 ciclos de reloj, esto es, de igual duración que la trama de entrada generada. Además, se generó el reloj del sistema a una frecuencia de 100 MHz, y una señal de *reset* activa en el primer ciclo de reloj para la inicialización del sistema.

En la figura 4.1 se muestra el resultado obtenido de la simulación. Las señales de entrada *data_in* y *data_in_valid* se indican en amarillo, mientras que las señales de salida *data_out* y *valid_out* se indican en celeste. Asimismo, se observan las diferentes

señales generadas por los bloques componentes del transmisor y receptor, en el orden en que son producidas.

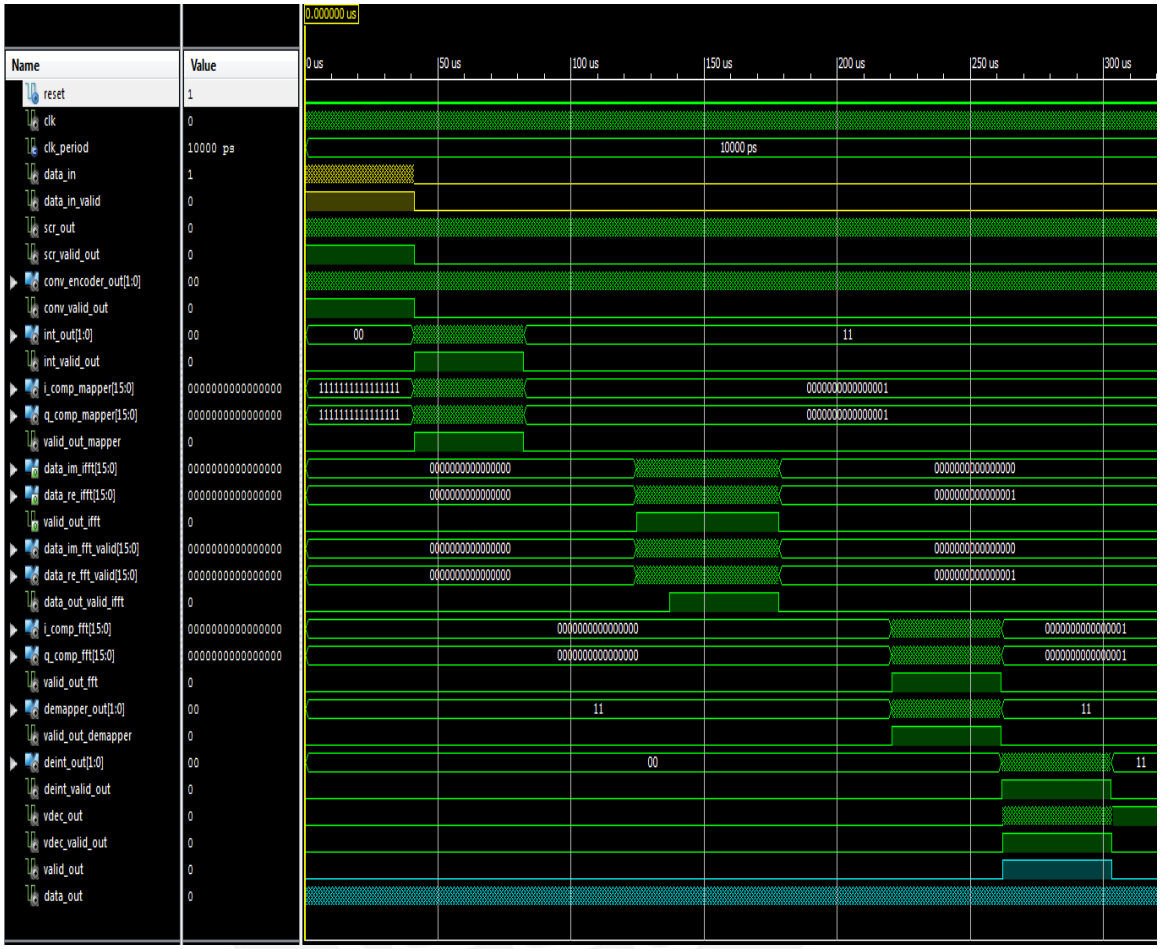


Figura 4.1: Simulación general del sistema de transmisión-recepción

En la figura 4.2 se comparan las primeras muestras válidas de la trama serial ingresada (*data_in*) y la trama serial recibida (*data_out*). Como se puede apreciar, los valores de las muestras presentadas coinciden.

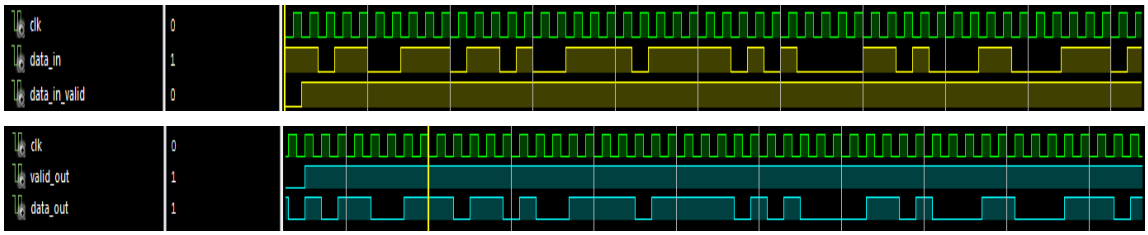


Figura 4.2: Comparación de primeras muestras de trama ingresada y trama recibida

Para la validación de la totalidad de la trama recibida, se exportaron los valores de dicha trama al software MATLAB®, y se comparó cada muestra válida con su correspondiente en la trama ingresada. Como resultado, se obtuvo que los valores de todas las muestras coincidieron, y por tanto las tramas eran iguales.

Asimismo, a partir de los valores de la señal OFDM generada en el transmisor, se elaboró un gráfico de dicha señal (figura 4.3). Se observa en la figura que la envolvente de la señal presenta un comportamiento fluctuante. Esto se debe a que la señal se compone de la suma de múltiples subportadoras con fases idénticas, lo cual produce que la potencia instantánea de la señal en determinados puntos sea mucho mayor a la potencia promedio de la misma, esto es, hace que la relación de potencia pico a promedio (*peak to average power ratio, PAPR*) sea elevada.

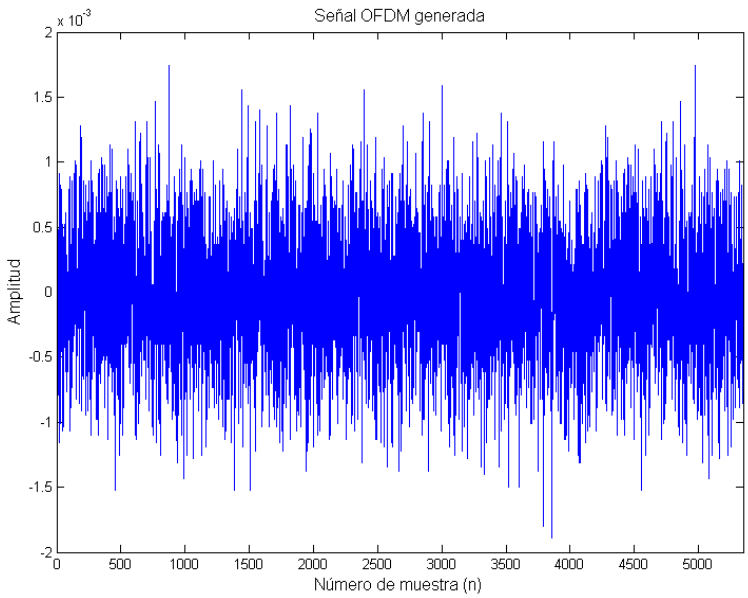
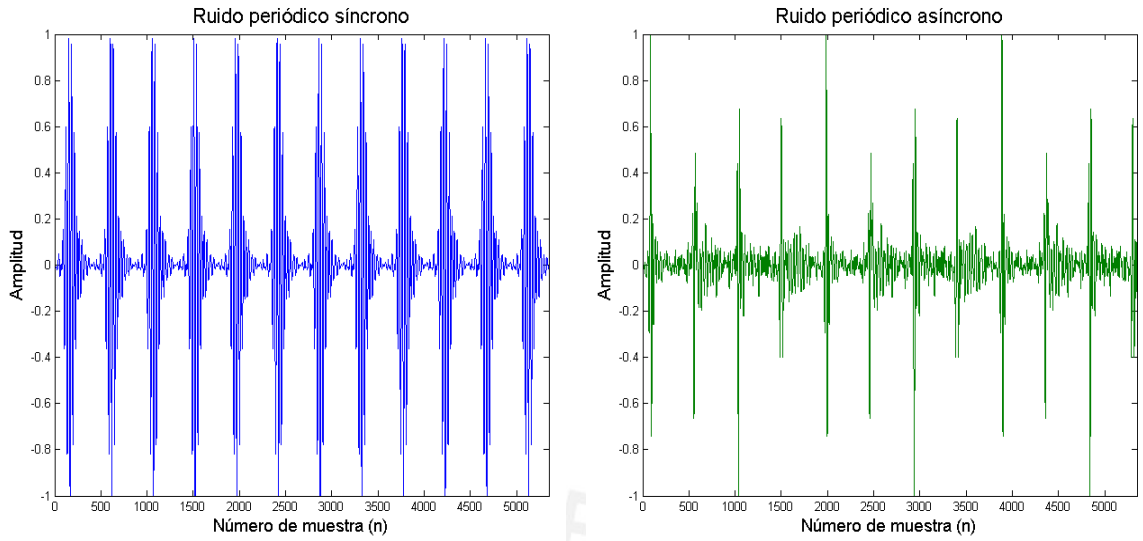


Figura 4.3: Señal OFDM generada en el transmisor

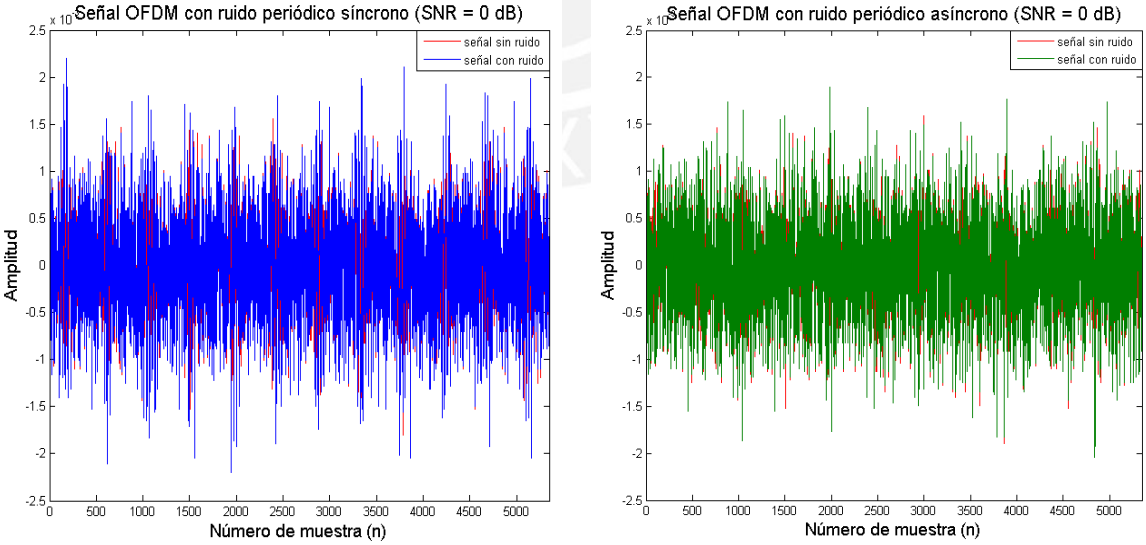
Otra de las principales simulaciones realizadas corresponde a aquélla de la operación del sistema en presencia de ruido. Para ello, se emplearon los modelos de distintos tipos de ruido de canal eléctrico presentados en [19]. En la figura 4.4(a) se presenta el modelo empleado del ruido periódico síncrono, mientras que en la figura 4.4(b) se presenta el modelo utilizado del ruido periódico asíncrono. Cabe mencionar que ambos son modelos ideales de estos tipos de ruido, y por tanto no han sido obtenidos a partir del análisis de una red eléctrica en particular.



(a) (b)

Figura 4.4: (a) Modelo de ruido periódico síncrono, (b) Modelo de ruido periódico asíncrono

Cada tipo de ruido se introdujo en la señal generada en el transmisor por medio de una suma entre la señal OFDM y la señal de ruido, empleando diferentes valores de relación señal a ruido (SNR). En la figura 4.5(a) se muestra la señal OFDM con ruido periódico síncrono para una SNR de 0 dB, mientras que en la figura 4.5(b) se muestra la señal OFDM con ruido periódico asíncrono también para una SNR de 0 dB.



(a) (b)

Figura 4.5: (a) Señal OFDM con ruido periódico síncrono, (b) Señal OFDM con ruido periódico asíncrono

Cada señal con ruido obtenida se empleó como señal de entrada del receptor, y se simuló la operación del mismo. Luego, la trama recibida se comparó con la trama ingresada en el transmisor, a fin de hallar la cantidad de bits erróneos en cada caso. La tabla 4.1 muestra los resultados obtenidos, donde la tasa de errores de bit (BER) es la relación entre la cantidad de bits erróneos y el total de bits recibidos (4096 bits). Para valores de SNR mayores a 11 dB, se obtuvo cero bits erróneos en ambos casos.

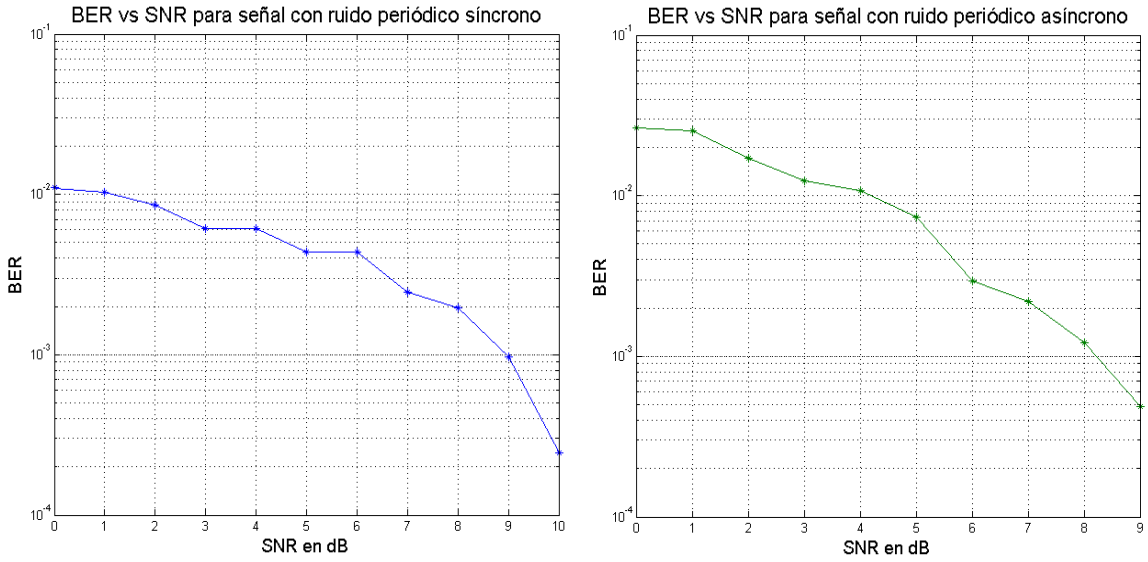
Tabla 4.1: Bits erróneos para distintos valores de SNR por cada modelo de ruido

	Ruido periódico síncrono		Ruido periódico asíncrono	
SNR	Bits erróneos	BER	Bits erróneos	BER
0 dB	45	0.010986	109	0.026611
1 dB	42	0.010254	104	0.025391
2 dB	35	0.008545	70	0.01709
3 dB	25	0.006104	51	0.012451
4 dB	25	0.006104	44	0.010742
5 dB	18	0.004395	30	0.007342
6 dB	18	0.004395	12	0.00293
7 dB	10	0.002441	9	0.002197
8 dB	8	0.001953	5	0.001221
9 dB	4	0.000977	2	0.000488
10 dB	1	0.000244	0	0
11 dB	0	0	0	0

Las figuras 4.6(a) y 4.6(b) muestran la relación BER vs SNR para la señal con ruido periódico síncrono y la señal con ruido periódico asíncrono, respectivamente.

En general, se observa una baja tasa de errores de bit en ambos casos, lo cual significa que el sistema diseñado responde de forma efectiva frente a ambos tipos de ruido. A pesar de que las señales OFDM con ruido presentan múltiples picos o puntos de elevada potencia instantánea, se recupera gran parte de los datos inicialmente transmitidos. Esto se debe principalmente a que los bloques de intercalado de datos empleados permiten que las muestras que presentan picos de potencia sean esparcidas, de modo que se facilita la corrección de errores en el Decodificador de Viterbi. Asimismo, la codificación QPSK utilizada reduce la probabilidad de error, puesto que genera símbolos muy distantes entre sí (en el diagrama de constelación),

de modo que evita en gran medida de que los símbolos se desplacen considerablemente a causa del ruido, y sean decodificados de forma errónea.



(a) (b)

Figura 4.6: (a) BER vs SNR para ruido periódico síncrono, (b) BER vs SNR para ruido periódico asíncrono

4.2. Resultados

La descripción del sistema diseñado mediante el lenguaje de descripción de hardware VHDL fue sintetizada, utilizando las herramientas del software ISE 14.4 de Xilinx®, para el dispositivo FPGA Spartan-6 XC6SLX45. Los principales resultados del proceso de síntesis se muestran en la tabla 4.2.

Tabla 4.2: Recursos utilizados en el FPGA Spartan-6 XC6SLX45

Recursos	Transmisor	Receptor	Transmisor-receptor
Número de *slice registers*	6832 (12%)	6495 (11%)	13357 (24%)
Número de *slice LUTs*	4790 (17%)	4790 (17%)	9576 (35%)
Número total de *slices*	1778 (26%)	1834 (26%)	3586 (52%)
Frecuencia máxima de operación	122,46 MHz	124,83 MHz	107,68 MHz

Se observa en la tabla que el transmisor-receptor diseñado emplea un total de 3586 elementos lógicos o *slices* (52% del total disponible). A partir de un análisis adicional, se obtuvo que únicamente los bloques IFFT y FFT utilizan 3205 *slices*. Esto se debe a que la arquitectura seleccionada para los *IP Cores* empleados en dichos bloques utiliza una gran cantidad de recursos a fin de realizar el cálculo de la FFT en una menor cantidad de ciclos de reloj en comparación con otras arquitecturas.

Asimismo, se aprecia que la frecuencia máxima de operación del sistema de transmisión-recepción es de 107,68 MHz. Esto hace posible que el sistema pueda operar a una tasa de bits máxima de 107,68 Mbps. Sin embargo, cabe resaltar que la tasa de bits real del sistema de transmisión-recepción depende adicionalmente de la capa de enlace de datos utilizada y del circuito empleado para el acondicionamiento analógico de la señal.

Se calculó, además, la latencia de procesamiento tanto en el transmisor como en el receptor, esto es, el tiempo que tardan los datos en ser procesados en ambos módulos. La latencia en el transmisor es de 137 110 ciclos de reloj, mientras que en el receptor es de 124 950 ciclos de reloj. Considerando una frecuencia de reloj de 100 MHz, esta latencia es de 1,37 ms en el transmisor, y 1,25 ms en el receptor.

Finalmente, por medio de la herramienta *XPower Analyzer* de Xilinx®, se obtuvo que la potencia promedio consumida por el sistema diseñado es de 202 mW, a una temperatura de 25°C, frecuencia de operación de 100 MHz y para un voltaje de alimentación del FPGA de 1,2V. Este valor está muy por debajo de la potencia máxima permitida por el dispositivo FPGA (960 mW) [20].

CONCLUSIONES

1. A partir de las simulaciones realizadas sobre el FPGA Spartan-6 XC6SLX45, a través de las cuales se comprobó el correcto funcionamiento del sistema descrito, se concluye que el diseño de la arquitectura propuesta para el sistema de transmisión-recepción PLC se ha realizado con éxito, y por tanto el objetivo general de la tesis ha sido cumplido.

2. El sistema de transmisión-recepción PLC diseñado está basado, en gran parte, en las especificaciones que señala el estándar IEEE 1901 para el diseño de la capa física de un dispositivo PLC de banda ancha. Por tanto, es una primera aproximación al desarrollo de un sistema de transmisión-recepción PLC totalmente compatible con dicho estándar.

3. En base a los resultados satisfactorios obtenidos en las simulaciones de la operación del sistema en presencia de diferentes modelos de ruido, se puede afirmar que los mecanismos empleados para la codificación y corrección de errores reducen efectivamente la propagación de los mismos en los paquetes de datos transmitidos.

4. El empleo de un dispositivo FPGA para la implementación del sistema de transmisión-recepción permitió alcanzar una elevada velocidad de procesamiento, principalmente para los cálculos de la IFFT y la FFT utilizados para la modulación y demodulación de los datos, respectivamente. Esto se debe fundamentalmente a la alta capacidad de paralelismo que este dispositivo ofrece.

5. La arquitectura propuesta se presenta como un esquema modular, por lo que sus bloques internos pueden ser modificados sin afectar el funcionamiento de los otros bloques. Asimismo, pueden ser reutilizados para posteriores diseños, incluso para aplicaciones distintas a la presentada.

6. A excepción de los bloques utilizados para el cálculo de la IFFT y la FFT, los módulos empleados en el diseño son portables, por lo que pueden ser implementados en dispositivos FPGA de distintas familias y fabricantes.

RECOMENDACIONES

El presente trabajo tuvo como objetivo el diseño de una arquitectura para un sistema de transmisión-recepción PLC. Para la implementación física del mismo y para su operación sobre un canal eléctrico, se requiere adicionalmente del diseño de un circuito de acondicionamiento de la señal a la línea eléctrica. Éste consta básicamente de circuitos ADC y DAC para la conversión analógica/digital y viceversa, circuitos de amplificación de voltaje y corriente, circuitos de filtro, y finalmente circuitos de acoplamiento a la red eléctrica. Asimismo, debe implementarse un circuito de sincronización de tiempo y frecuencia en el receptor, de modo que se pueda detectar efectivamente el inicio y fin de una trama de datos recibida.

A fin de desarrollar un sistema de transmisión-recepción compatible con el modelo de capa física (PHY) propuesto por el estándar IEEE 1901, debe hacerse una serie de modificaciones al diseño propuesto. En primer lugar, debe reemplazarse el par Codificador Convolucional-Decodificador de Viterbi utilizado para la detección y corrección de errores por un Codificador-Decodificador Turbo Convolucional. Esto obliga, además, a modificar el bloque Intercalador, pues trabajaría con una mayor cantidad de bits de paridad por cada dato codificado. Asimismo, debe modificarse los módulos IFFT y FFT según las especificaciones del estándar, sobre todo en relación a los ángulos de fase establecidos para las subportadoras. Adicionalmente, debe incluirse un bloque de inserción de preámbulo en el transmisor, y uno de sincronización de tiempo en el receptor, a fin de asegurar la detección de los símbolos OFDM. Por último, deben añadirse los bloques que señala el estándar para los datos de control de trama.

Finalmente, con el objetivo de facilitar la transferencia de grandes volúmenes de información, debe implementarse una capa de enlace de datos (*data link layer*) que controle la operación del sistema diseñado. Para ello, puede emplearse como referencia el modelo de sub-capa de control de acceso al medio (MAC) que propone el estándar IEEE 1901.

BIBLIOGRAFÍA

[1] K. S. Al Mawali
2011 "Techniques for Broadband Power Line Communications: Impulsive Noise Mitigation and Adaptive Modulation", A Thesis Submitted for the Degree of Doctor of Philosophy, RMIT University

[2] Batres E., Ingrid
2006 "Consideraciones generales para transmisión de datos a través de la red eléctrica", Tesis de licenciatura de Universidad de San Carlos de Guatemala

[3] Carcelle, Xavier
2009 "Power Line Communications in Practice", Artech House, primera edición

[4] HomePlug AV2 whitepaper
2012 Disponible en línea. Consulta: 12 de julio de 2013.
<http://www.homeplug.org/tech/whitepapers/HomePlug_AV2_White_Paper_v1.0.pdf>

[5] IEEE 1901 HD-PLC technical overview
2012 Disponible en línea. Consulta: 12 de julio de 2013.
<http://www.hd-plc.org/modules/newsroom/index.php?page=article&storyid=13>

[6] Shih-Chieh Hsu
2004 "Study and FPGA Implementation of Powerline Communication System using OFDM Signals", Thesis for Master of Science, National Cheng Kung University

[7] Sanchez Pérez, Alfonso
2010 "Implementación en hardware de un transmisor/receptor para comunicaciones mediante PLC según PRIME", Escuela Técnica Superior de Ingeniería, Universidad Pontificia Comillas

[8] García-Baleón, H.A. y V. Alarcón-Aquino
2009 "A Power-Line Communication Modem based on OFDM", International Conference on Electrical, Communications and Computers, CONIELECOMP 2009, pp. 208-213

[9] Shieh, William e Ivan Djordjevic
2010 “Orthogonal Frequency Division Multiplexing for Optical Communications”, Academic Press, primera edición

[10] Vijaya Chandran R.
2007 “Orthogonal Frequency Division Multiplexing”, University of Kansas, Disponible en línea. Consulta: 12 de julio de 2013.
<https://www.cresis.ku.edu/~rvc/documents/862/862_ofdmreport.pdf>

[11] Hanzo, Lajos y Thomas Keller
2006 “OFDM and MC-CDMA: A Primer”, Wiley, primera edición

[12] Oppenheim, Alan V. y Ronald W. Schafer
2009 “Discrete-Time Signal Processing”, Prentice Hall, tercera edición

[13] IEEE Standards Association
2010 “IEEE Standard for Broadband over Power Line Networks: Medium Access Control and Physical Layer Specifications”

[14] Pong P. Chu
2006 “RTL Hardware Design Using VHDL: Coding for Efficiency, Portability and Scalability”, Wiley-IEEE Press, primera edición

[15] Volnei A. Pedroni
2008 “Digital Electronics and Design with VHDL”, Elsevier, primera edición

[16] Sandun Rathnayake
2012 “Viterbi Transceiver”, Disponible en línea. Consulta: 12 de julio de 2013.
<http://www.opencores.org>

[17] Xilinx LogiCORE IP Fast Fourier Transform v8.0 Datasheet
2012 Disponible en línea. Consulta: 12 de julio de 2013.
<http://www.xilinx.com/support/documentation/ip_documentation/ds808_xfft.pdf>

[18] Samuel C. Yang
2010 “OFDMA System Analysis and Design”, Artech House Mobile Communications Series, primera edición

[19] Working Group on Power Line Communications, University of Málaga
Disponible en línea. Consulta: 12 de julio de 2013
<http://www.plc.uma.es/channels.htm>

[20] Xilinx Spartan-6 Family Datasheet
2011 Disponible en línea. Consulta: 12 de julio de 2013.
<http://www.xilinx.com/support/documentation/data_sheets/ds160.pdf>