TELEMÁTICA
REVISTA DIGITAL de las Tecnologías de la Información y las Telecomunicaciones
Revista Telemática. Vol. 18. No. 2, mayo-agosto, 2019, p.1- 17 ISSN 1729-3804

# DISEÑO EN FPGA DE MÓDULO DE PROPIEDAD INTELECTUAL PARA RECEPCIÓN DE SEÑALES DE ESPECTRO ESPARCIDO

**Alejandro Arteaga Pérez[1], Miguel Ángel Bring Fernández[2], Sanjony Yasmany O´Reilly Lebrijio[3], Jorge Torres Gómez[4]**

[1, 2, 3]CIDP "Grito de Baire", Santa Ana # 711, e/ 47 y Reforma, Nuevo Vedado, Plaza de la Revolución, La Habana, Cuba; [4]CUJAE-Facultad de Telecomunicaciones y Electrónica, Cuba
[1]email: alejandroa@gb.reduim.cu
[4]email: jtorres151184@gmail.com

**RESUMEN**

En la actualidad los sistemas de comunicación inalámbricos demandan mayor robustez de comunicación para evitar los efectos no deseados del canal de comunicaciones. Entre estos efectos no deseados se encuentran la interferencia inter-símbolo (ISI) y la producida por otras comunicaciones. A modo de contrarrestar este efecto los sistemas de espectro esparcido brindan una mayor robustez y seguridad en las comunicaciones, tanto en aplicaciones de índole comercial y en interés de la defensa. El presente trabajo aborda la modulación de salto de frecuencia (FHSS) como técnica de espectro esparcido. Se describe la etapa de transmisión y recepción para su diseño en FPGA. En la recepción se implementa el recuperador de sincronismo y la demodulación teniendo en cuenta las características principales de aleatoriedad y condiciones del canal de comunicaciones. Se presenta el diseño en FPGA para XILINX en relación a las etapas de adquisición y seguimiento para la recuperación de sincronismo integrado con el microprocesador Microblaze. Adicionalmente se muestran los resultados de las simulaciones de dichas etapas para el correcto desempeño del sistema.

**PALABRAS CLAVES:** Espectro Esparcido, FHSS, FPGA.

# INTELLECTUAL PROPERTY MODULE DESIGN TO RECEIVING SPREAD SPECTRUM SIGNALS

**ABSTRACT**

Currently, wireless communication systems demand to further improve robustness to avoid non-desirable channel effects. Commonly, these channel effects are related to inter-symbol interference (ISI) and the interference produced by other communicating signals. In order to reduce the impact of channel on the received signal Spread Spectrum systems are currently implemented to bring increased robustness and communication security. This is commonly used on commercial as well as military applications. Current work addresses frequency hopping spread spectrum (FHSS) waveform to be used on waveform transmissions. Transmitter and receiver are designed for FPGA devices. Synchronization block to be implemented on the receiver is described based on main characteristics of the transmitted signal regarding randomness and channel state. FPGA design is also described on XILINX

platforms regarding acquisition and tracking stages. Furthermore, simulation results shown proper performance of the proposed system.
**KEY WORDS:** Spread Spectrum, FHSS, FPGA.

## 1. INTRODUCCIÓN

En la actualidad las comunicaciones inalámbricas son altamente afectadas por los efectos adversos del canal de comunicaciones en la medida que aumentan las capacidades de comunicación. Efectos no deseados como la interferencia inter-símbolo y la causada por la masividad de dispositivos de comunicación degradan la calidad de la señal recibida. En este sentido, se establecen y conforman distintas formas de onda para la transmisión de información para contrarrestar estos efectos no deseados.

Entre los esquemas de comunicación empleados los sistemas de espectro esparcido por salto de frecuencia ofrecen una solución flexible al empleo de bandas de comunicación más apropiadas [1]. Diversas soluciones de sistemas por salto de frecuencia (frequency hopping) se reportan por nuevos diseños de lazo [2], esquemas de seguridad para la palabra pseudo-aleatoria [3] así como nuevos receptores [4]. Aplicaciones de estos esquemas se reportan también en sistemas de radar [5], comunicaciones satelitales [6] y comunicaciones cooperativas [7].

Los sistemas de espectro esparcido se constituyen a partir de las siguientes características [8]:

La señal esparcida ocupa un ancho de banda muy superior al mínimo ancho de banda requerido para transmitir la información.

El esparcimiento se obtiene por la aplicación de una señal, comúnmente llamada señal de código, la cual es independiente del dato transmitido.

En el receptor se recupera la señal de información por la correlación de la señal recibida con una réplica sincronizada de la señal de código.

Los beneficios que aporta la modulación por salto de frecuencia se logran a partir de aumentar la dimensionalidad de la señal dado por 2WT, donde W representa el ancho de banda de la señal y T la duración. El aumento de esta dimensionalidad se logra con la aplicación de una señal adicional. Esta señal adicional se representa por una secuencia pseudo-aleatoria aplicada en el transmisor. La secuencia en la etapa del receptor debe reproducirse en sincronía con la generada en el transmisor para una recuperación correcta de la señal.

El presente artículo describe de forma general la creación de un módulo de propiedad intelectual (IP) para un sistema de recuperación de sincronía de la secuencia pseudo-aleatoria en receptores de salto de frecuencia. Este diseño se conforma en FPGAs para Xilinx aprovechando la ventaja de estos sistemas de alto rendimiento y capacidad de integración. Este módulo IP se integra luego con el microprocesador Microblaze para una mayor flexibilidad de funciones.

El presente artículo se organiza como sigue: La Sección 2 describe el diseño en FPGA del sistema de sincronía para señales de salto de frecuencia. La Sección 3 describe el diseño del módulo IP. La Sección 4 muestra los resultados obtenidos en la recuperación de sincronía. Por último, las conclusiones establecen consideraciones y proyecciones futuras del presente trabajo.

## DISEÑO DEL SISTEMA DE SINCRONISMO EN FPGA

Para la recuperación correcta de la información en la señal recibida se debe hacer sincronía entre las secuencias de pseudocódigo del transmisor y el receptor. Esta sincronía se conforma en dos etapas: por Adquisición (Adquisiton) y Seguimiento (Tracking) [9]. La etapa de Adquisición establece sincronía entre ambas secuencias hasta la mitad del tiempo de chip. La etapa de Tracking establece una sincronía más ajustada entre ambas secuencias. Ambas etapas funcionan de forma secuencial, primero se establece la adquisición y luego el seguimiento.

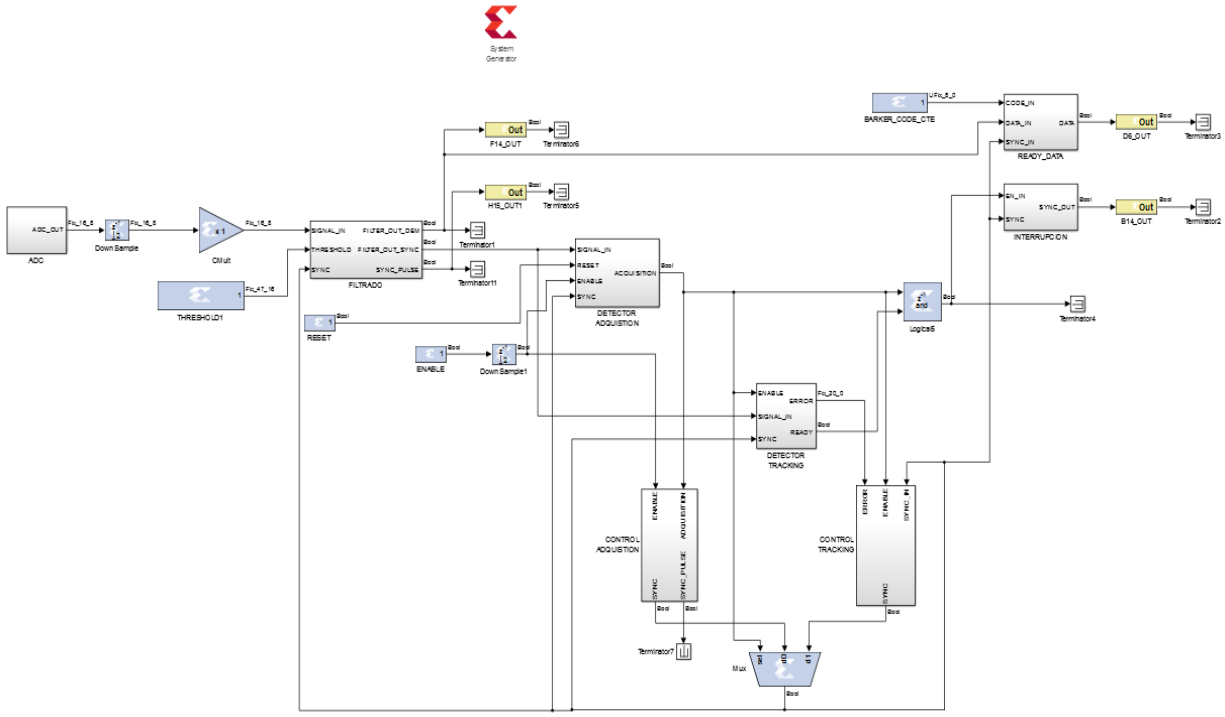



Figura 1: Esquema del receptor para el sistema FHSS en System Generator.

El esquema diseñado para sincronismo en FPGA se muestra en la Fig. 1. El receptor está compuesto por esquemas para la adquisición, tracking, demodulación de datos y generación de interrupciones. Estos esquemas se constituyen por los siguientes bloques [10]:

1. ADC: Implementa el conversor análogo-digital de 8 bits empleado para la captura de la señal FHSS de entrada al FPGA.
2. Filtrado: Implementa el producto, filtrado y elevación al cuadrado de las ramas en fase y cuadratura. Este bloque es compartido por las etapas de Adquisition y Tracking.
3. Detector Adquisition: Implementa el bloque de detección de la etapa de Adquisición. Su señal binaria a la salida se establece en nivel alto cuando las palabras de pseudocódigo recibidas y generadas se encuentran alineadas.

4. Control Adquisition: Implementa la habilitación de los pulsos de reloj del sistema. Este bloque adelanta o mantiene la secuencia de pulsos de reloj dados por la etapa de Adquisición en medio tiempo de chip.
5. Detector Tracking: Bloque encargado de determinar la alineación de las palabras de pseudocódigo transmitidas y recibidas mediante el algoritmo de Seguimiento.
6. Control Tracking: Genera la señal de reloj para la generación de la palabra pseudo-aleatoria. Este bloque adelanta o mantiene la secuencia de pulsos de reloj dados por la etapa de Seguimiento según la diferencia de fase entre la señal recibida y la generada por el receptor.
7. Bloque Mux: Multiplexor que conmuta entre la señal de reloj generada por la etapa de Adquisición y la generada por la etapa de Tracking. Este bloque activa de forma consecutiva ambas etapas, una vez que finaliza la adquisición se activa el funcionamiento del algoritmo de seguimiento.
8. Ready_data: Implementa la detección del código Barker insertado al inicio de cada paquete de datos para una correcta detección de la información transmitida.
9. Interrupción: Genera las interrupciones a mitad de tiempo de bit para recuperar los datos transmitidos de forma correcta.

Los bloques de Adquisition y Tracking se integran de forma secuencial en el tiempo mediante señales de salidas llamadas READY. Estas señales indican que el procesamiento asociado al bloque ha terminado. Cada una de las señales READY se conecta a la entrada ENABLE del bloque que le sigue para habilitar su procesamiento. A continuación se describen los bloques que fueron actualizados para un mejor desempeño del sistema en el presente diseño en comparación con el mostrado en [10].

**Bloque de Filtrado**

El bloque de filtrado determina entre las ramas en fase y cuadratura el grado de parecido entre el tono que arriba y el generado por el receptor. Su esquema está compuesto por dos ramas idénticas conformadas por dos DDS, filtros y detectores de nivel. La Fig. 2 muestra la interconexión de estos bloques. Los bloques DDS junto con la señal de entrada alimentan a los bloques de detección FILTER_UPPER_BRANCH y FILTER_LOWER_BRANCH. Cada uno de estos bloques de detección se encarga de establecer la presencia del tono rápido o lento en la señal recibida. La salida de los bloques de detección es luego comparada con el bloque Relational para conformar la señal binaria como lo muestra la Fig. 3, estas obtenidas desde un osciloscopio externo al FPGA. En la Fig. 3 a) se muestra la salida de ambos bloques de detección, en la Fig. 3 b) se muestra el resultado de comparar ambas ramas con el bloque Relational.

Cada uno de los bloques FILTER_UPPER_BRANCH y FILTER_LOWER_BRANCH está conformado de forma idéntica por un detector de envolvente como muestra la Fig. 4. La conexión de elevadores al cuadrado, filtrado y valor absoluto establece la presencia de energía cuando coinciden las frecuencias de la señal recibida SIGNAL_IN y las generadas en el receptor PHASE_IN y QUADRATURE_IN.

La Fig. 5 muestra una sección de filtrado de segundo orden. Los filtros utilizados se componen de tres secciones en serie idénticas a la mostrada, cambiando solo los parámetros de los multiplicadores. De esta forma se establecen cuatro filtros IIR de orden 6.

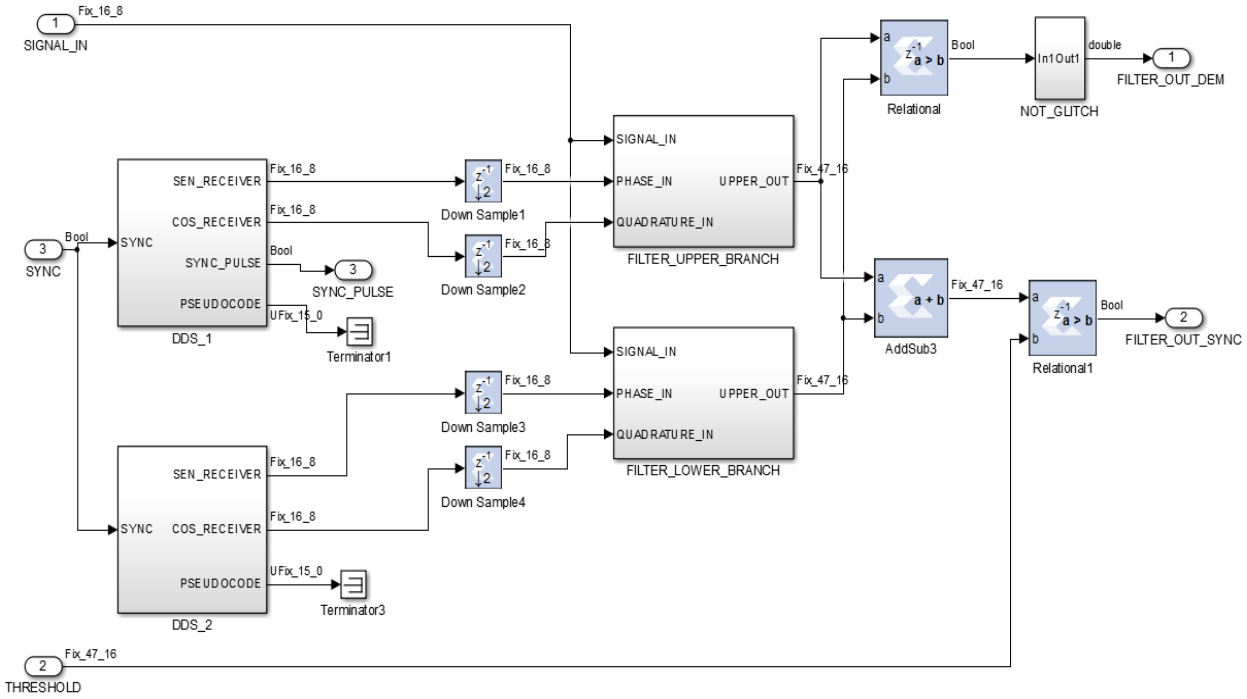
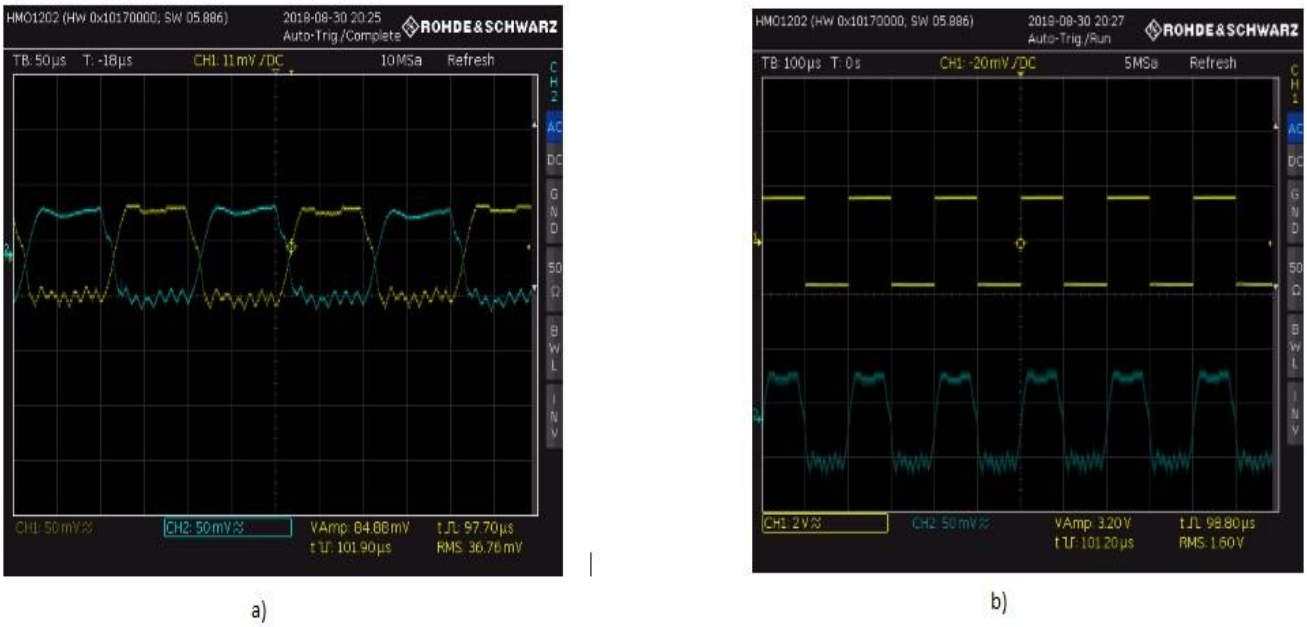



Figura 2: Bloque de filtrado.




a) b)

Figura 3: a) Señal alternada de ambas ramas después de filtradas. b) Datos binarios recibidos en color amarillo.

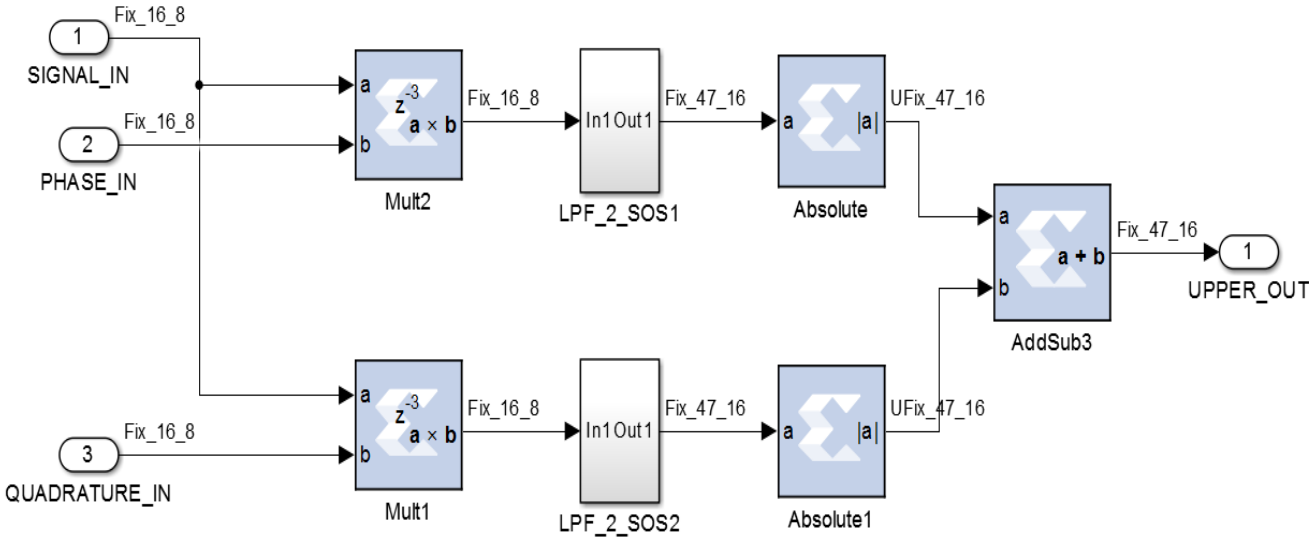



Figura 4: Conformación de los bloques FILTER_UPPER_BRANCH y FILTER_LOWER_BRANCH mediante un detector de energía.

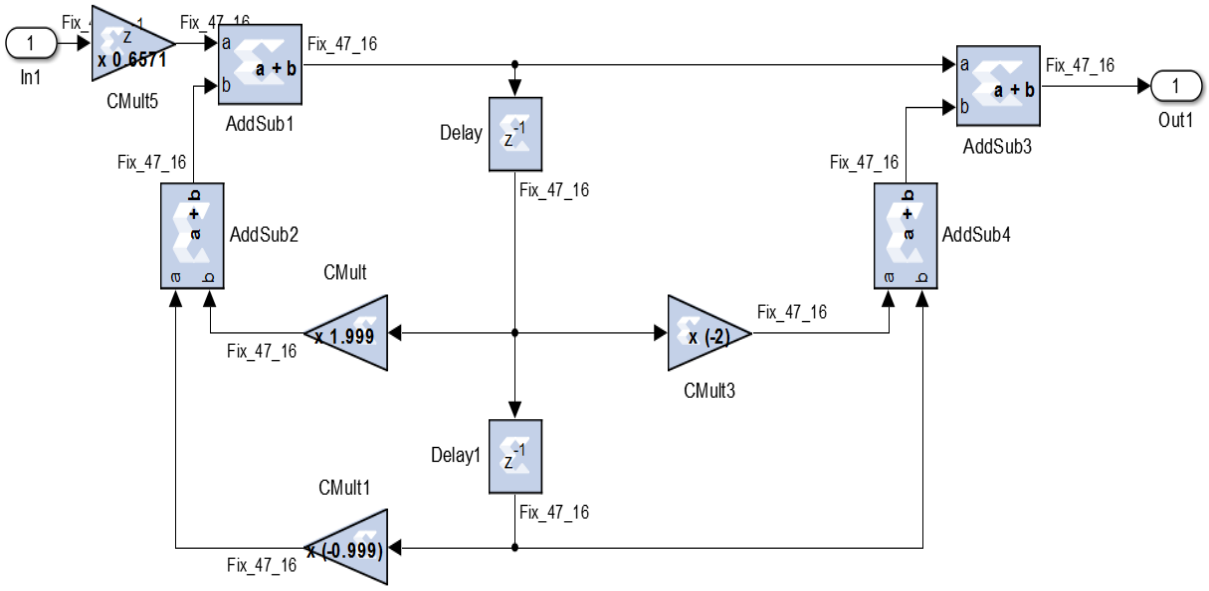



Figura 5. Sección de segundo orden del filtro IIR.

**Bloque de detección del código Barker.**

Los datos provenientes del transmisor están conformados por paquetes de 48 bits, existiendo intervalos de tiempo entre cada uno donde no hay información. A los mismos se le añade un código Barker de 8 bits como cabecera para facilitar la detección de las tramas de información.

El bloque de detección del código Barker se adiciona al diseño expuesto en [10], el mismo se muestra en la Fig. 6 y consta de diferentes bloques. Los retardadores (Delay) se encargan de convertir los bits demodulados que se reciben por DATA_IN de serie a paralelo. Posteriormente el bloque concat concatena los datos conformando una secuencia de datos de 8 bits y por último se compara con una referencia del código Barker guardado en una constante y que se introduce al bloque a través del puerto de entrada CODE_IN. Una vez detectado el código correcto se conforma un pulso a la salida para informar que a partir de ese momento se pueden comenzar a recibir datos validos hasta completar un paquete de 48 bits.

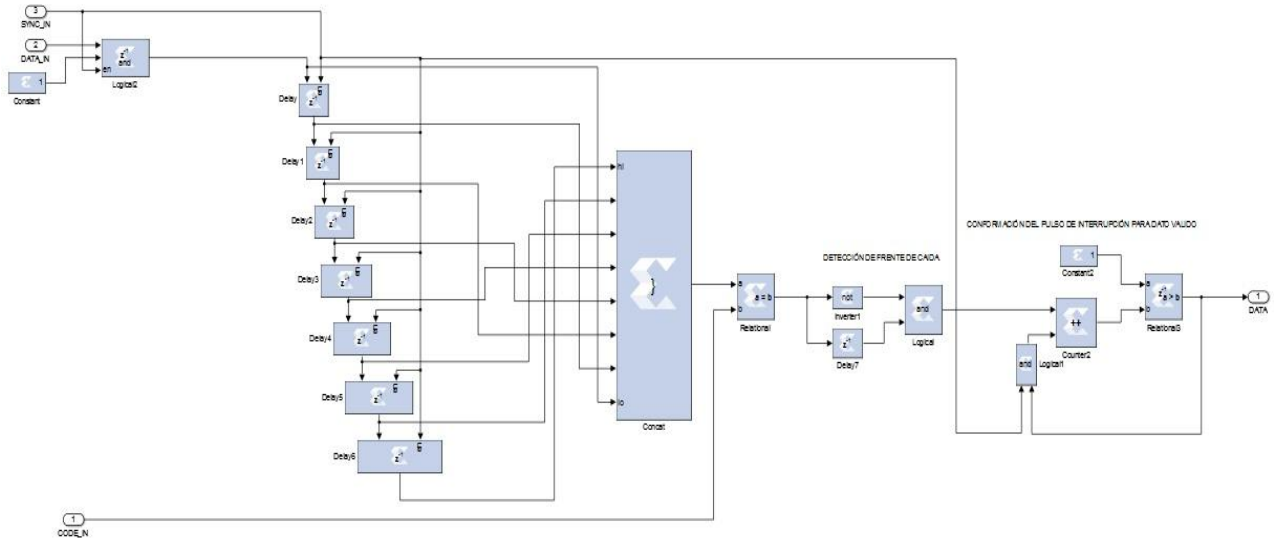



Figura 6: Bloque detector de código Barker.

## CONFECCIÓN DEL MÓDULO DE PROPIEDAD INTELECTUAL (IP)

Una vez concluida la etapa de diseño del sistema para sincronizar y recibir las señales FHSS, se conforma un módulo de propiedad intelectual (IP). Este módulo IP se establece con el objetivo de incorporarlo a un microprocesador Microblaze embebido en un Spartan 6 de Xilinx para mayor flexibilidad del diseño. Desde el microprocesador, mediante el lenguaje de programación C/C++, se establecen funcionalidades con mayor flexibilidad, aunque sin la ejecución rápida de operaciones de hardware del módulo IP.

En el desarrollo del módulo IP intervienen tres etapas de diseño fundamentales sobre diferentes entornos de programación:

1. System Generator (SYSGEN 14.7): En esta aplicación se realiza el diseño RTL del receptor, se comprueban los resultados desde la simulación y se generan los ficheros VHDL para conformar el módulo.

2. XPS 12.1: Software para conformar el proyecto de integración entre el microprocesador Microblaze y el módulo IP del receptor. En este software también se conforma el módulo IP de usuario y se generan los drivers para un correcto manejo del módulo generado por parte del microcontrolador Microblaze.

3. ISE 14.7: Software en el que se editan y compilan los ficheros de funcionamiento del módulo IP y se establece el orden jerárquico de los mismos.

**Flujo de diseño del módulo IP:**

1. Primeramente se compila el diseño del receptor en SYSGEN. De esta forma se generan los ficheros VHDL a conformar como módulo IP. La Figura 7 muestra las opciones seleccionadas en el Token de Xilinx para la compilación del diseño.

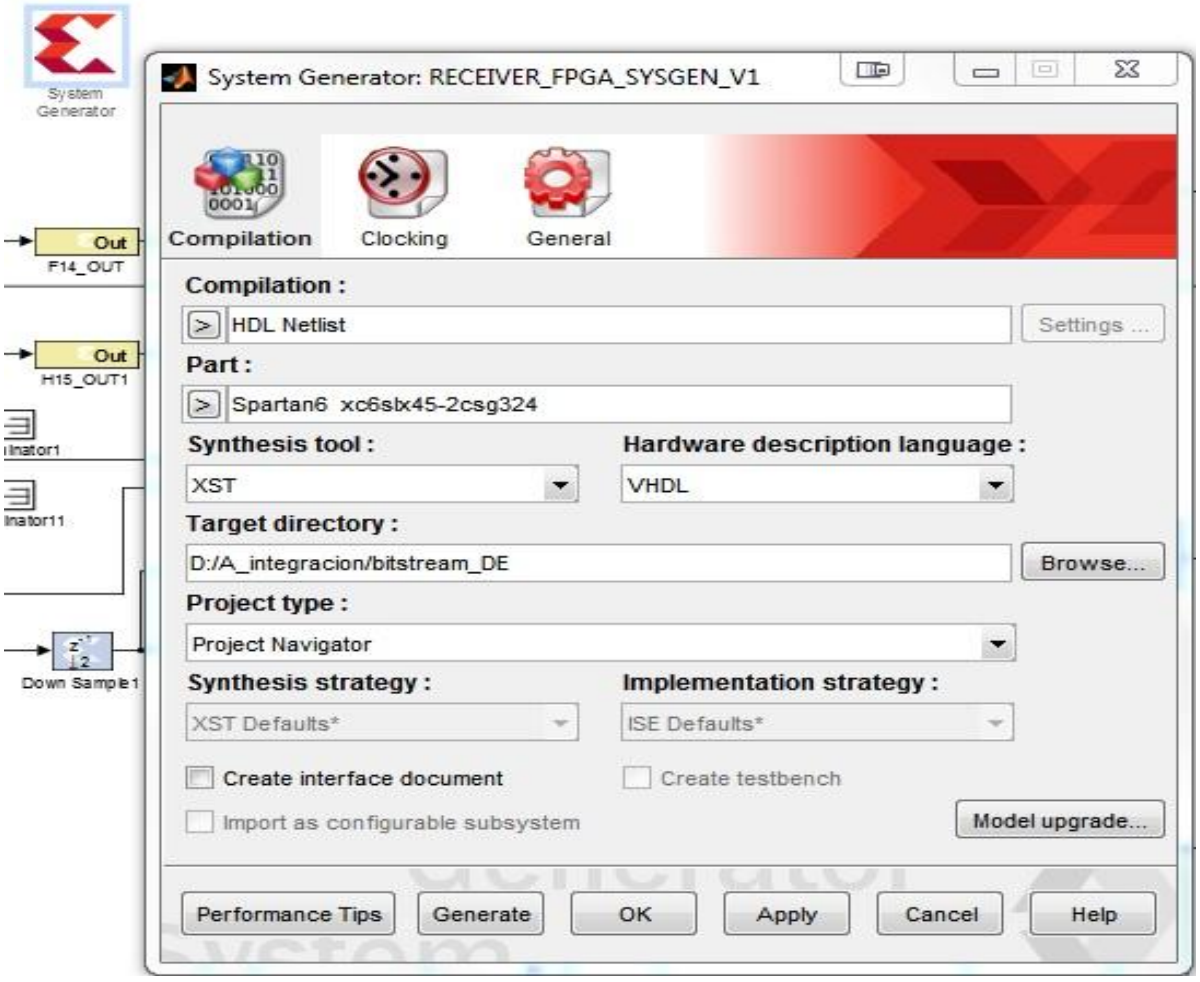



Figura 7.Configuracion de opciones para generar ficheros VHDL en sysgen.

2. Una vez generados los ficheros VHDL, se procede a conformar el proyecto en XPS (Xilinx Platform Studio). Para crear el nuevo módulo IP se selecciona la opción Hardware/Create or Import Peripheral…, donde aparece una ventana de configuración. A través de este Wizard se escogen las opciones de configuración del módulo IP: se escoge el bus local del procesador (PLB v46), se selecciona el empleo del control de interrupciones y se definen la cantidad de registros de escritura/lectura que serán empleados. En este diseño se establece un solo registro de escritura para la comunicación entre Microblaze y el módulo IP. Este registro está conectado a la entrada de habilitación de los bloques de sincronía del receptor. Después de generado el módulo IP aparecerá en la ventana IP Catalog del XPS el

nuevo periférico como se muestra en la Fig. 8 a). En este paso se generan de forma automática por la herramienta los ficheros para el trabajo con los drivers y los ficheros de configuración VHDL.

3. Una vez conformados los dos ficheros de configuración VHDL, estos generados en el paso anterior, se procede a describir la estructura jerárquica del módulo IP empleando el software ISE, la Fig. 8 b) muestra cómo quedan estructurados en el proyecto los ficheros VHDL. En estos ficheros se establece la funcionalidad del demodulador conformado en System Generator y se conforma la interfaz con el bus PLB en lenguaje VHDL.

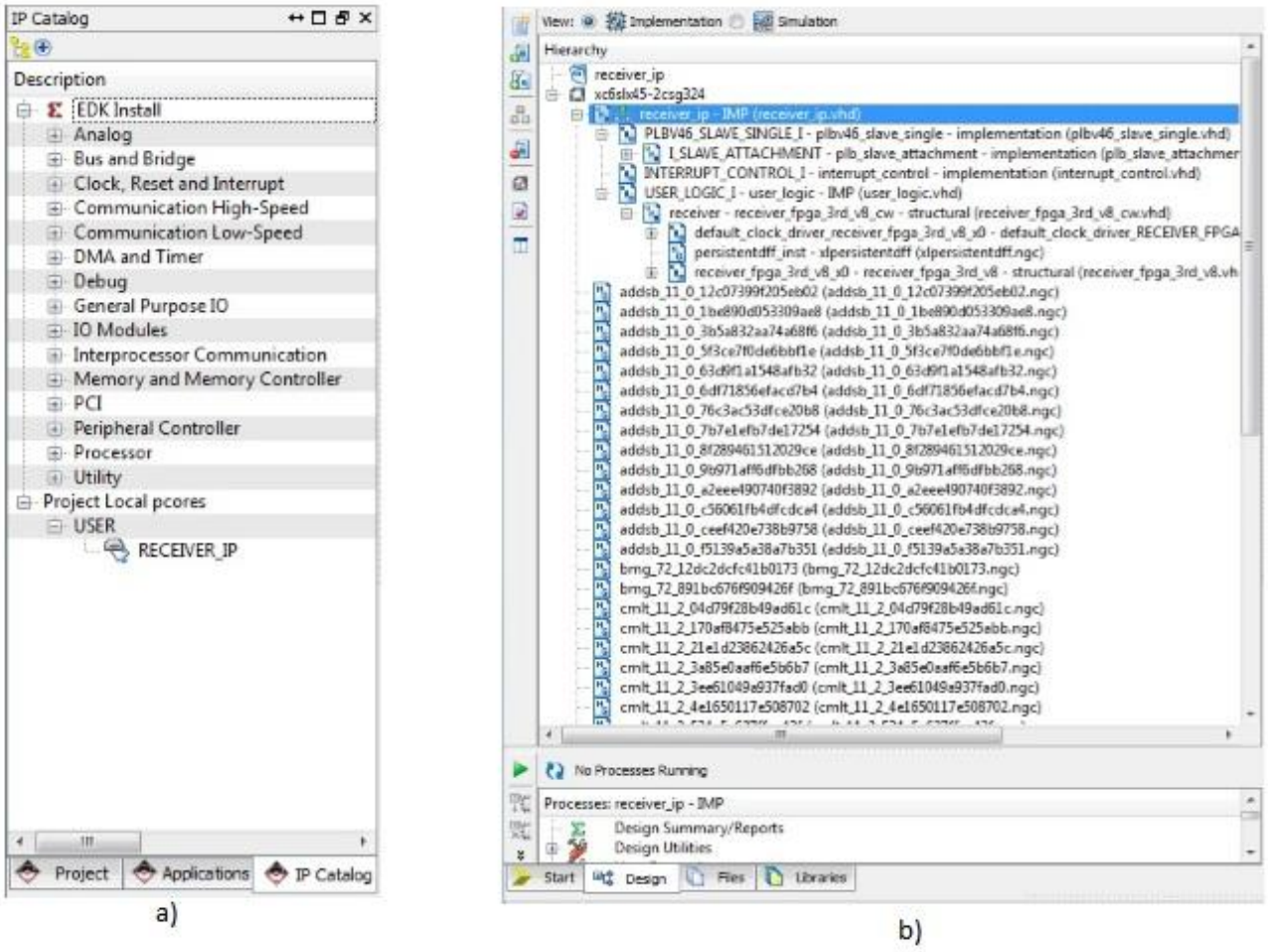



Figura 8: a) Módulo IP generado en XPS. b) Jerarquía de ficheros VHDL.

4. Una vez interconectada la lógica de usuario en el módulo IP, entonces se procede a importar el módulo hacia el diseño. En el entorno de XPS se accede nuevamente a la opción de importar el módulo IP. Esto se realiza empleando el Wizard para la importación de periféricos integrando los nuevos ficheros y descripciones de funcionalidades mostradas en la Fig. 8 b). La importación del módulo hacia XPS permite su posterior conexión a Microblaze y configuración software.

5. Una vez importado el módulo IP, queda la adición del módulo al proyecto y su interconexión al bus PLB. Este bus es el que establece la comunicación entre el

microprocesador Microblaze y el módulo IP. El módulo IP se encuentra disponible en la pestaña IP Catalog/ USER, este se añade al proyecto dando doble click sobre el mismo. En la Figura 9 se puede observar la conexión establecida del módulo IP "receiver_ip_0" al bus PLB.

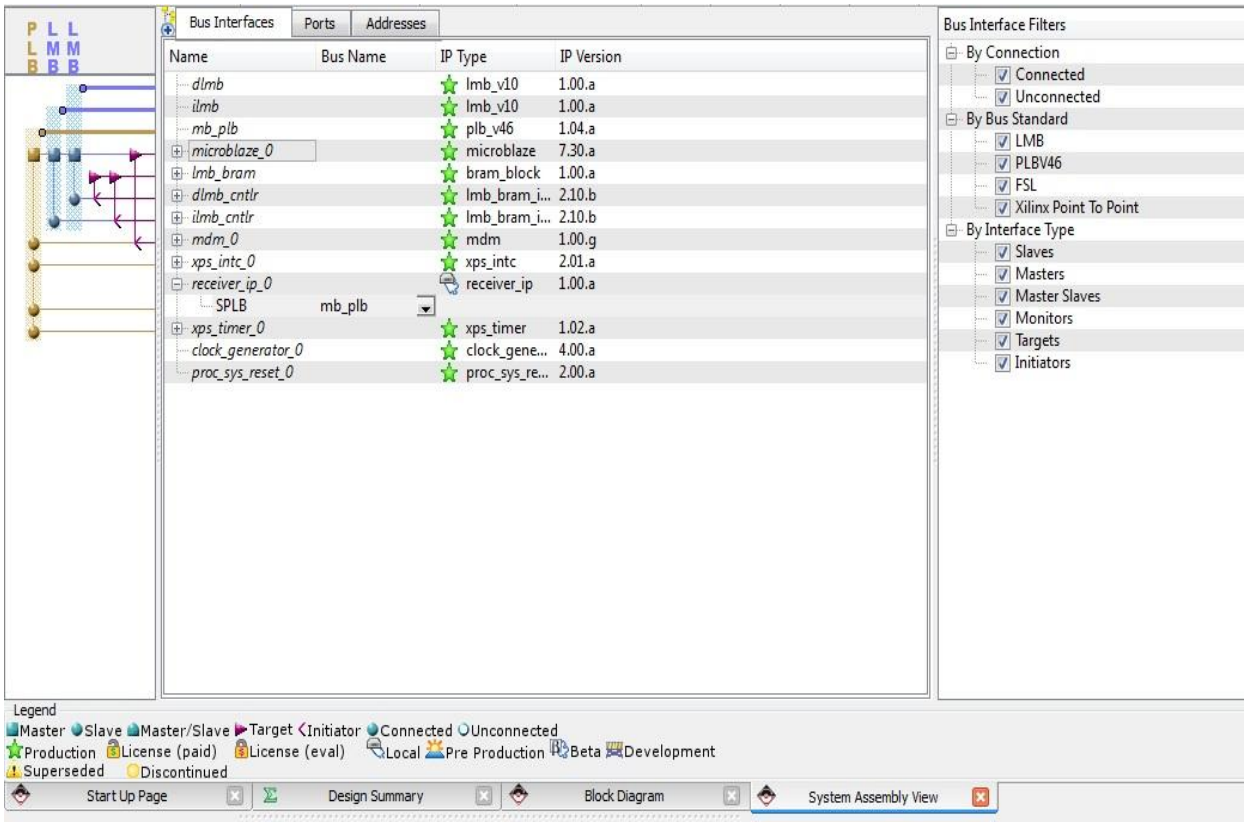



Figura 9: Conexión del módulo IP al bus PLB.

Luego de conectar el módulo IP al bus PLB se necesitan configurar sus puertos y direcciones de acceso a sus registros internos. En la misma pestaña de System Assembly View se selecciona la opción Ports. En esta ventana se interconectan el resto de los puertos que no pertenecen al bus PLB como son los externos y las interrupciones. Entre los puertos externos están los que se conectan al conversor A/D, así como otros para el análisis del sistema con el osciloscopio, como son la salida de sincronismo, interrupciones y datos demodulados.

Para interconectar la interrupción debe añadirse un módulo IP controlador de interrupciones si antes no fue incorporado al inicio del proyecto. Esto se hace de la misma forma a como se incluyó el módulo IP creado anteriormente. Luego se accede a la opción Addresses en la misma pestaña del System Assembly View. En esta se generan las direcciones por la cuales Microblaze accederá al módulo IP. En esta ventana se da click al botón Generate Addresses y el sistema las genera automáticamente. De esta forma queda establecido el mapa de direcciones para todos los módulos IPs del proyecto.

Por último, se modifica el fichero UCF que se encuentra bajo la pestaña Project, donde se establece la localización física de los puertos de entrada, salida, reloj y reset del sistema. La Fig. 10 muestra cómo queda dicha configuración.

6. Una vez configurada la interconexión del módulo IP al bus PLB se procede a generar el proyecto. Para ello se accede al botón Generate Bitstream mediante el menú Hardware/Generate Bitstream. Esta opción compila la síntesis, mapeo y ruteo de las señales y puertos del FPGA respondiendo a las conexiones conformadas en el proyecto. En la Fig. 11 se puede observar cómo quedan conectados todos los módulos IP que intervienen en el diseño al microcontrolador a través del bus PLB. Este paso genera el fichero bitstream empleado para la programación del FPGA.

7. Una vez generado el bitstream es posible la programación de la aplicación sobre Microblaze. Para esto se emplea el software SDK como entorno de desarrollo. Sobre esta interfaz se programará el FPGA y se ejecutará el código sobre Microblaze. En la plataforma SDK se necesitan establecerse tres esquemas de programación fundamentalmente:

a) Plataforma Hardware: Donde queda establecida la programación del FPGA mediante el fichero de extensión ".bit".

b) Plataforma Software: En esta plataforma quedan establecidos los drivers para operar con los módulos IPs.

c) Proyecto C/C++: Proyecto de programación de microblaze en interacción con los módulos IPs.

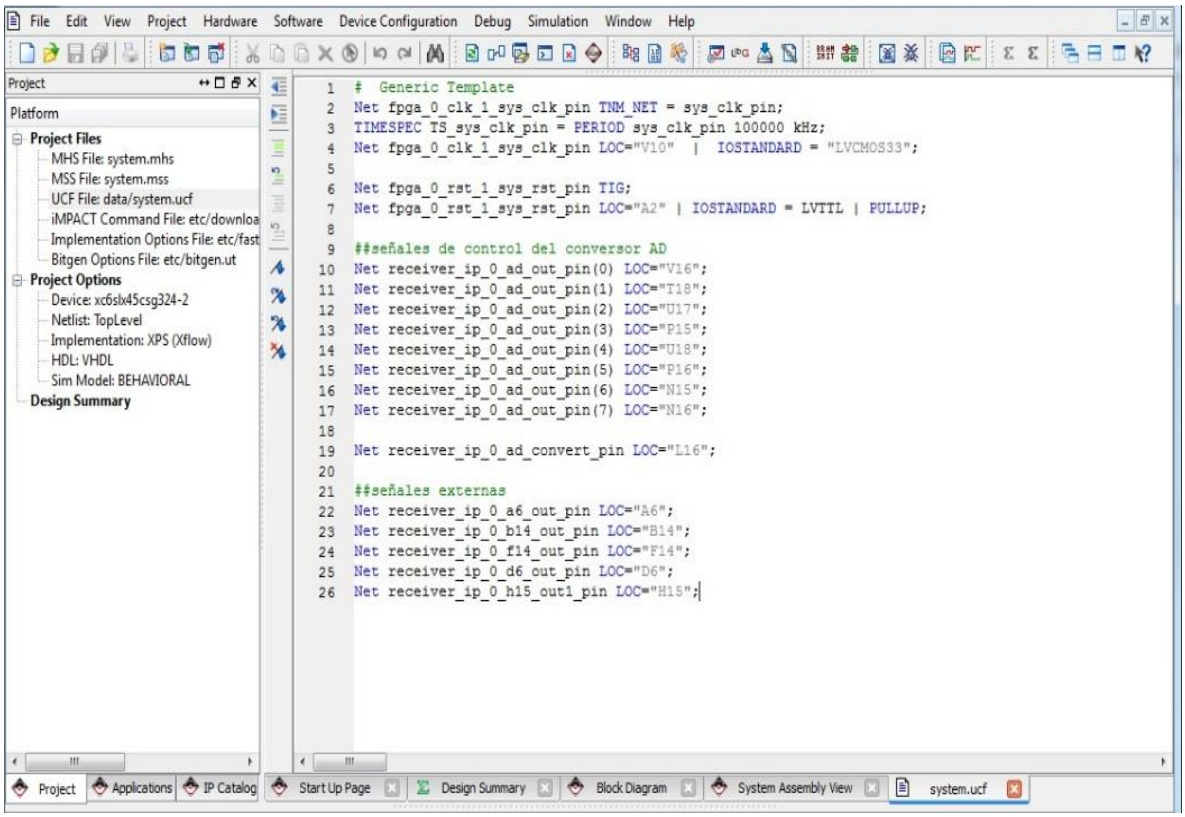


Figura 10: Configuración de los pines físicos del FPGA.

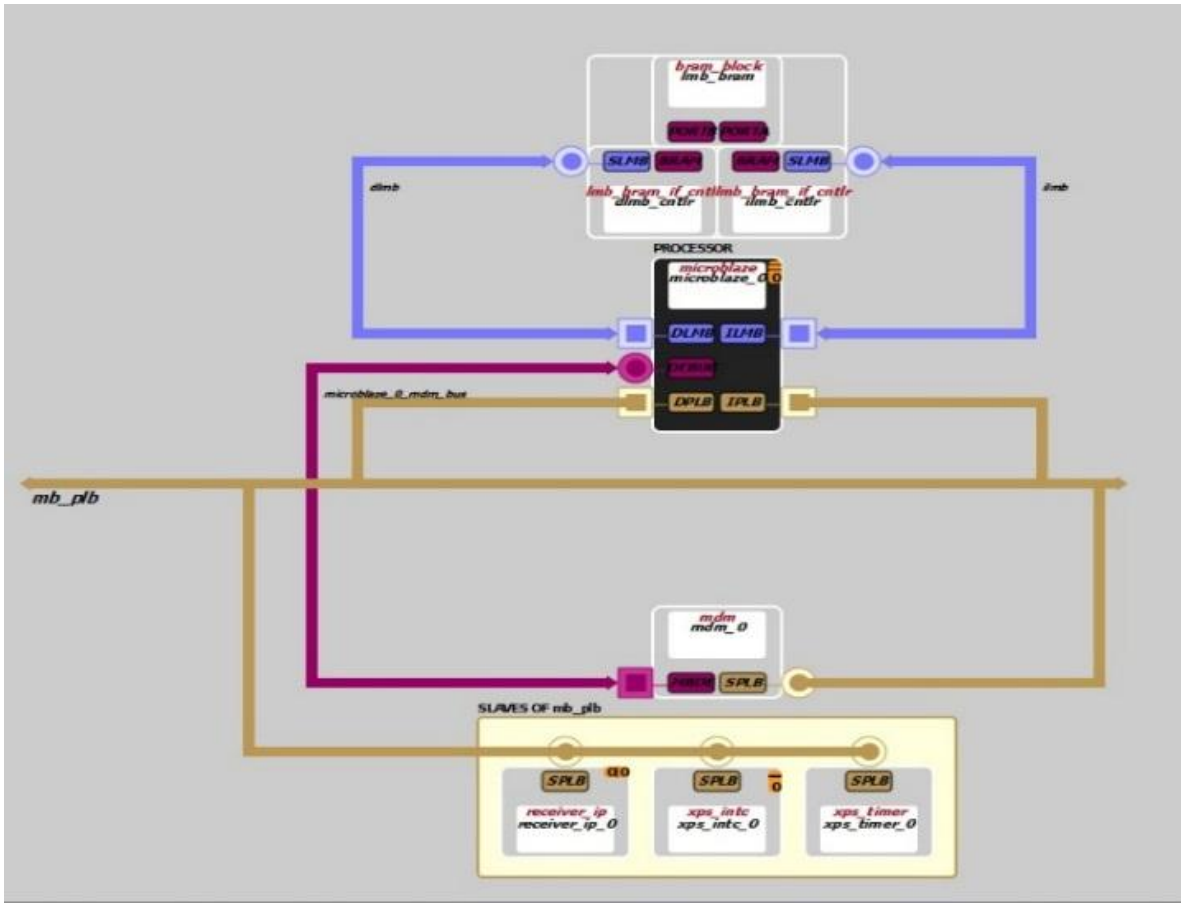



Figura 11: Diagrama en bloques de conexiones al bus PLB.

En la Fig. 12 se puede observar cómo quedan conformadas las diferentes plataformas Hardware y Software, además del proyecto en C. En este proyecto en C se incluyen los diferentes ficheros drivers de los diferentes módulos periféricos que intervienen en el funcionamiento del receptor.

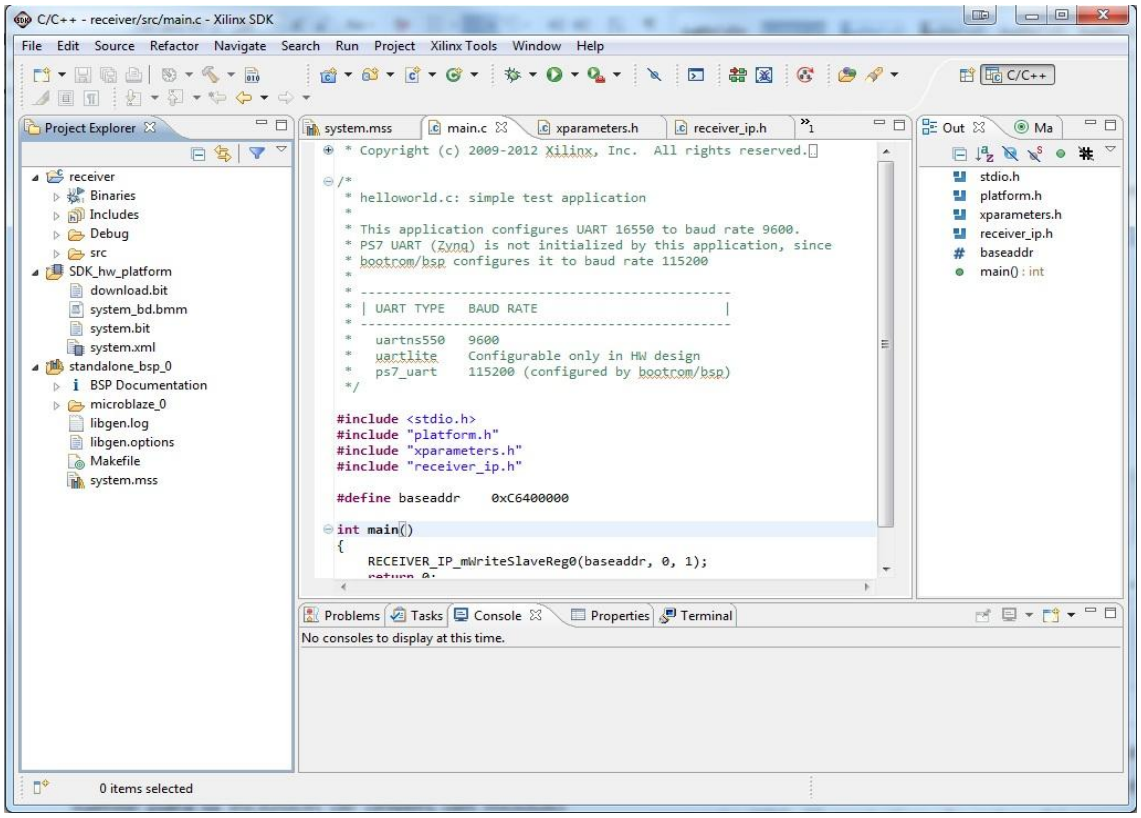



Figura 12: Proyecto C del módulo IP en plataforma SDK.

## RESULTADOS OBTENIDOS EN LA RECUPERACIÓN DE SINCRONÍA

Para la obtención de resultados se emplean los siguientes recursos: para el transmisor se emplea el FPGA Spartan 6 xc6lx16. En este FPGA se conecta el conversor D/A AD9753 de 12 bits 300 MSPS, como muestra la Figura 13. El conversor D/A es el encargado de transmitir la forma de onda en un canal de comunicación continuo.

Para el receptor se emplea el FPGA Spartan 6 xc6lx45. A este FPGA se le conecta el conversor A/D AD9057 de 8 bits 80 MSPS, con la configuración mostrada en la Fig. 13. Este conversor A/D es el encargado de capturar la señal transmitida para procesarla internamente en el FPGA.

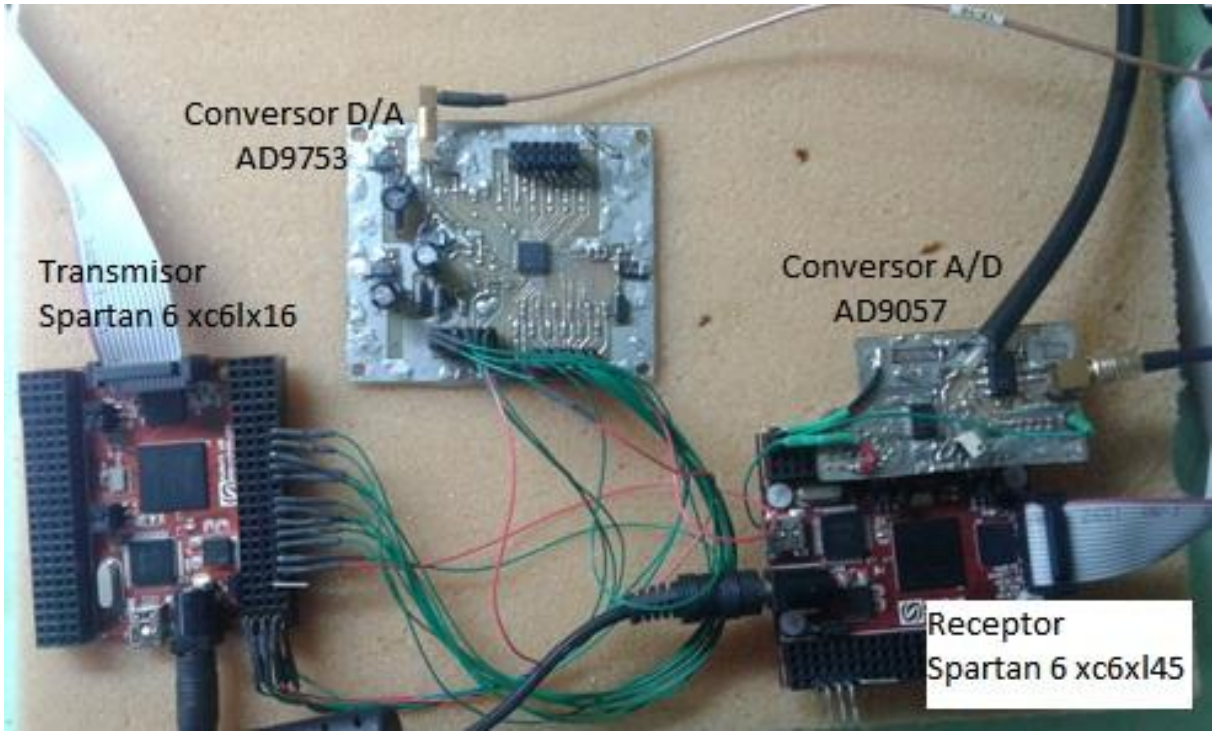



Figura 13: Configuración Transmisor/Receptor empleada.

Para la prueba se configura la transmisión de una secuencia de 56 bits conformada por un código de 8 bits. Este código se establece de valor 219, luego se le incorporan 48bits de datos conformados por 6 registros de 8 bits con un valor de 170 en cada uno. La Fig. 14 muestra un ejemplo de la señal continua transmitida por el conversor DA. En esta captura el osciloscopio muestra tres frecuencias distintas en el gráfico superior (tiempo) y tres picos en la descripción frecuencial del gráfico inferior.

Figura 14: Señal generada en el transmisor

Como parte de los resultados obtenidos, la Fig. 15 muestra la salida de los filtros diseñados para cada rama, teniendo a la entrada una secuencia alternada de ceros y unos. Se puede observar como las ramas alternan sus niveles indicando la recepción de ceros y unos alternos. La duración de las transiciones entre los niveles altos y bajos resulta apropiada en comparación con la duración del símbolo, lo cual permite una correcta recuperación de los datos.

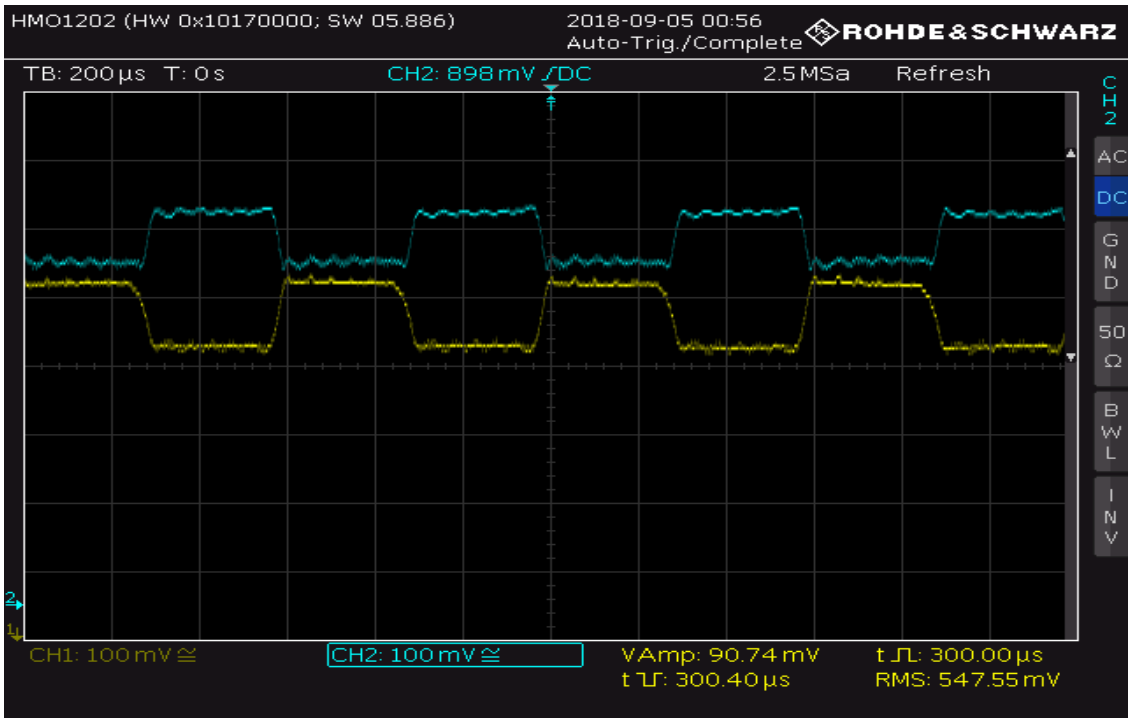
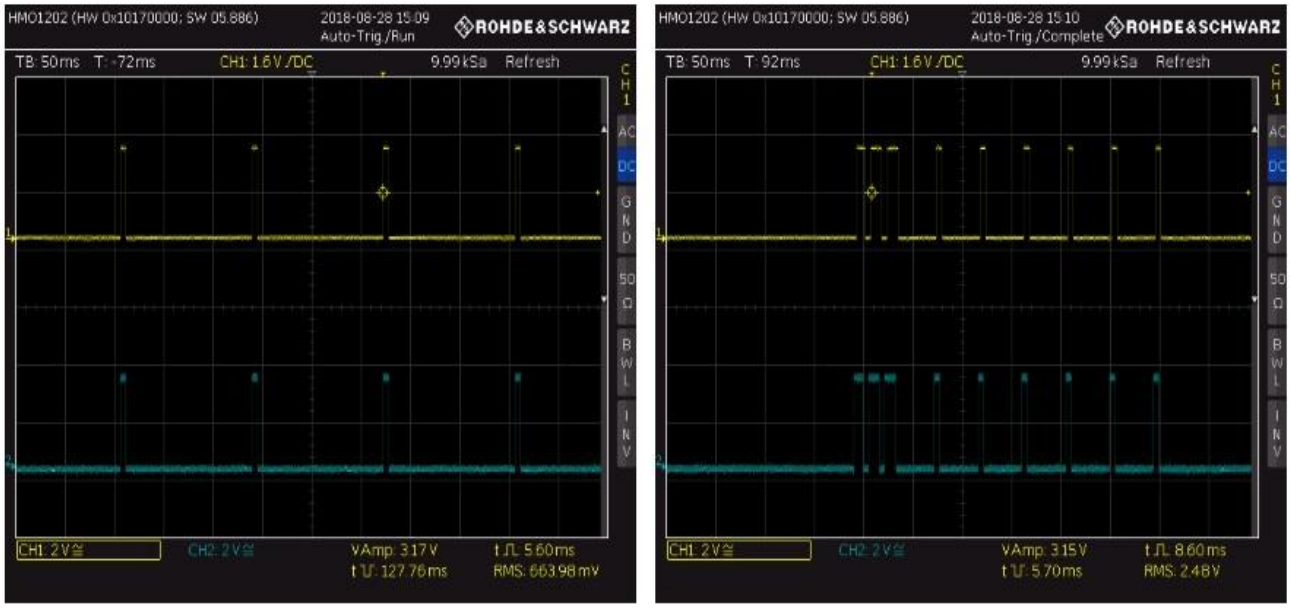



Figura 15: Salida de los filtros de cada rama para una secuencia alternada.

En adición, entre los resultados se evalúa la recuperación correcta de la sincronía en el receptor mediante la alineación de las secuencias transmitida y recibida. Se puede observar en la Fig. 16 como se logra una correcta sincronía entre el transmisor y el receptor. En la Fig. 16 a) se pueden apreciar los pulsos de sincronía del transmisor en amarillo y los del receptor en azul. Estos se encuentran correctamente alineados luego de establecerse las etapas de adquisición y seguimiento. En la Fig. 16 b) se observa el paquete de datos transmitido en color amarillo y su correcta recuperación por parte del receptor en color azul.




a) b)

Figura 16: a) Pulsos de sincronismo, amarillo transmisor y azul receptor. b) Paquete de datos recuperado en el receptor (azul) a partir de los datos transmitidos (amarillo).

## CONCLUSIONES

El sistema presentado en este trabajo integra la concepción de un esquema de transmisión y recepción de señales de salto de frecuencia FHSS. El diseño se concibe en FPGA por las facilidades de y rendimiento del procesamiento en hardware desde su programación en software. Para implementar el sincronismo de señales FHSS en la etapa de recepción se establecen dos etapas fundamentales por las operaciones de adquisición y seguimiento. La integración de estas etapas se produce de manera secuencial en función de su tiempo de procesamiento. Finalmente se conforma un módulo de propiedad intelectual, empleando las principales herramientas ofrecidas por Xilinx y se conecta a un microcontrolador embebido. La conexión de un microcontrolador brinda las potencialidades de interactuar con flexibilidad hacia capas de aplicación como interfaz de usuario.

## RECONOCIMIENTOS

El presente proyecto está soportado por el Centro CIDP-Grito de Baire con la integración de la Facultad de Telecomunicaciones y Electrónica, CUJAE.

**REFERENCIAS**

[1] B. Ning, Z. Li, L. Guan, y F. Zhou, «Probabilistic frequency-hopping sequence with low probability of detection based on spectrum sensing», *IET Commun.*, vol. 11, n.º 14, pp. 2147-2153, 2017.

[2] M. Nashed y A. A. Fayed, «Current-Mode Hysteretic Buck Converter With Spur-Free Control for Variable Switching Noise Mitigation», *IEEE Trans. Power Electron.*, vol. 33, n.º 1, pp. 650-664, ene. 2018.

[3] A. Chorti y E. V. Belmega, «Secret key generation in Rayleigh block fading AWGN channels under jamming attacks», presentado en 2017 IEEE International Conference on Communications (ICC), 2017, pp. 1-6.

[4] F. Yang, W. XIONG, y J. SONG, «Equal Gain Combining Based Maximum Likelihood Detector for FFH/MFSK Systems», *IEEE Commun. Lett.*, vol. PP, n.º 99, pp. 1-1, 2017.

[5] P. Zhou, F. Zhang, Q. Guo, S. Li, y S. Pan, «Reconfigurable Radar Waveform Generation Based on an Optically Injected Semiconductor Laser», *IEEE J. Sel. Top. Quantum Electron.*, vol. 23, n.º 6, pp. 1-9, nov. 2017.

[6] G. Wang *et al.*, «SATCOM link adaptive configuration design in radio frequency interference environment», presentado en 2017 IEEE Aerospace Conference, 2017, pp. 1-8.

[7] W. M. Jang y S. Sikander, «Cooperative cognitive systems with orthogonal frequency division multiplexing and frequency hopping», presentado en 2017 IEEE International Conference on Electro Information Technology (EIT), 2017, pp. 007-009.

[8] B. Sklar, *Digital Communications, Fundamentals and Applications*, Second. New Jersey: Prentice Hall, 2001.

[9] M. K. Simon, J. K. Omura, R. A. Scholtz, y B. K. Levitt, *Spread Spectrum Communications Handbook, Revised Edition*, Revised, Subsequent edition. New York: McGraw-Hill, 1994.

[10] A. A. Pérez, M. Á. B. Fernández, S. Y. Or. Lebrijio, y T. Gómez, «DISEÑO EN FPGA DE SISTEMA DE SINCRONISMO PARA SEÑALES DE ESPECTRO ESPARCIDO», 2018.

**SOBRE LOS AUTORES**

**Alejandro Arteaga Pérez**, Ingeniero en Telecomunicaciones y Electrónica, se encuentra realizando la maestría en Sistemas de Telecomunicaciones, desarrollador de aplicaciones para las comunicaciones en CIDP-Grito de Baire.

**Miguel Angel Bring Fernández**, Ingeniero en Telecomunicaciones y Electrónica, desarrollador de aplicaciones para las comunicaciones en CIDP-Grito de Baire.

**Sanjony Yasmany O´Reilly Lebrijio**, Ingeniero Informático, desarrollador de aplicaciones para las comunicaciones en CIDP-Grito de Baire.

**Jorge Torres Gómez**, Ingeniero en Telecomunicaciones y Electrónica, Master en Sistemas de Telecomunicaciones, Doctor en Ciencias Técnicas, se encuentra trabajando como profesor Auxiliar en la Facultad de Telecomunicaciones y Electrónica, CUJAE.