DESDE 2013
https://repository.uaeh.edu.mx/revistas/index.php/icbi/issue/archive
Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI

ISSN: 2007-6363

Publicación Semestral Pädi Vol. 10 No. Especial 4 (2022) 181-189

Emulación en FPGA de un sistema transceptor de RF basado en radio definida por software
FPGA emulation of a RF system transceiver based on software define radio

M. S. Aviña-Zuñiga[a], M. A. Estudillo-Valdez[b], G. E. Vazquez-Alcaraz[a], A. Calvillo-Tellez[b], J. C. Nuñez-Perez[b]

[a] *Tecnológico Nacional de México, Instituto Tecnológico de Tijuana, 22430, Tijuana, Baja California, México.*
[b] *Instituto Politécnico Nacional, IPN-CITEDI, 22435, Tijuana, Baja California, México.*

Resumen

La Radio Definida por Software SDR utiliza técnicas digitales para reemplazar el hardware de radio tradicional como mezcladores, moduladores, demoduladores y circuitos analógicos relacionados, el resultado es una radio flexible que puede ser rápidamente reconfigurada. En este artículo se desarrolla y emula un sistema de radiocomunicaciones completo, es decir el transmisor-receptor, usando SDR. Para lograr lo anterior se usa la técnica de DSSS y el System Generator for DSP de Xilinx en Matlab-Simulink. Se compila el modelo y se obtiene el código VHDL para emularlo en una tarjeta de FPGA Artix-7 AC701 de Xilinx. Posteriormente se realiza la encriptación y desencriptación de datos digitales utilizando LFSR y BPSK. Además, se realiza la comparación entre los resultados obtenidos, entre un transceptor DSSS usando un LFSR estándar contra otro DSSS usando un LFSR extendido. Finalmente, se presenta el diseño de una interface gráfica en Matlab, capaz de modificar los parámetros del modelo de transceptor DSSS para adaptarlo a diferentes frecuencias de transmisión.

Palabras Clave: FPGA, RF, SDR, Transmisión-recepción, VHDL.

Abstract

SDR Software Defined Radio uses digital techniques to replace traditional radio hardware such as mixers, modulators, demodulators, and related analog circuits, the result is a flexible radio that can be quickly reconfigured. In this article, a complete radio communication system, that is, the transmitter-receiver is developed and emulated using SDR. To achieve the above, the DSSS technique and the System Generator for DSP in Matlab-Simulink are used. The model is compiled and the VHDL code is obtained to emulate it on a Xilinx Artix-7 AC701 FPGA card. Finally, the encryption and decryption of digital data is performed using LFSR and BPSK. In addition, the comparison between the results obtained is made, between a DSSS transceiver using a standard LFSR against another DSSS using an extended LFSR. Finally, the design of a graphical interface in Matlab is presented, capable of modifying the parameters of the DSSS transceiver model to adapt it to different transmission frequencies.

Keywords: FPGA, RF, SDR, Transmission-reception, VHDL.

*Autor para la correspondencia: nunez@citedi.mx
Correo electrónico: mario.avina19@tectijuana.edu.mx (Mario Salvador Aviña-Zuñiga), mestudillo@citedi.mx (Miguel Angel Estudillo-Valdez), gilberto.vazquez@tectijuana.edu.mx (Gilberto Enrico Vazquez-Alcaraz), calvillo@citedi.mx (Andrés Calvillo-Tellez), nunez@citedi.mx (Jose Cruz Nuñez-Perez).

Historial del manuscrito: recibido el 02/05/2022, última versión-revisada recibida el 22/08/2022, aceptado el 01/09/2022, publicado el 05/10/2022. **DOI: https://doi.org/10.29057/icbi.v10iEspecial4.9141**

1. Introducción

Desde las primeras transmisiones inalámbricas alrededor de 1890, las técnicas de transmisión por radio han evolucionado continuamente, brindando a los usuarios la posibilidad de permanecer conectados con velocidades de transmisión crecientes (Raychaudhuri *et al.*, 2012). La era triunfal de la radio llegó primero, a mediados de 1930, en un momento en que se usaban anchos de banda limitados para las comunicaciones de voz analógicas. Luego, llegó la era dorada de la transmisión de radiodifusión en los años 50 con transmisiones de televisión analógicas que consumían más ancho de banda, pero brindaban una experiencia más simple y sencilla al cliente (Kleinrock, 2010). A medida que las computadoras se fueron haciendo más pequeñas y potentes, llegando a los años 60, comenzaron a ser útiles los medios de comunicación a largas distancias, utilizando tanto la conectividad a través de cables (Abramson, 1985). Por esta época también surgieron los teléfonos móviles (Frenkiel *et al.*, 2010), que permitían a los usuarios establecer comunicaciones inalámbricas de voz desde cualquier lugar público o vehículo.

En el mundo actual, las radios existen en una multitud de artículos, como teléfonos celulares, computadoras, vehículos y televisores, llaves de automóviles, etc. La evolución hacia los sistemas de Radio Definida por Software (por sus siglas en inglés, SDR) ha sido impulsada en parte por el desarrollo de las tecnologías habilitadoras. En primer lugar, los convertidores DAC y ADC y los Procesadores de Señales Digitales (por sus siglas en inglés, DSP), pero también los Procesadores de Propósito General (por sus siglas en inglés, GPP) y los arreglos de compuertas lógicas programables en campo (por sus siglas en inglés, FPGA) (Ulversoy, 2010). Una fuerza impulsora también ha sido la demanda de soluciones de comunicación radio más flexibles y reconfigurables, en particular del sector militar. Esta demanda ha dado lugar a importantes programas gubernamentales de desarrollo (Stephens *et al.*, 2006; North *et al.*, 2006).

GNU Radio (Tucker *et al.*, 2009), es una herramienta de desarrollo libre y abierto que provee bloques de procesamiento de señal para implementar sistemas de SDR. Puede utilizarse con hardware de RF de bajo costo para crear radios definidas por software, o sin hardware en un ambiente de simulación. Las aplicaciones de GNU Radio se construyen mediante un entorno gráfico o mediante el lenguaje de programación Python directamente, mientras que la sección que requiera alto rendimiento se implementa en C++, como es el caso de sus librerías. El Periférico de Radio de Software Universal (USRP) es una serie de hardware de RF compatible las plataformas Linux, MacOS y Windows (Olivieri, 2011). Existen varios modelos de hardware compatibles con el software matemático Matlab / Simulink y con GNU Radio, los cuales son los soportes más utilizados en las investigaciones de SDR (Abdulsatar *et al.*, 2013; Feng, 2013). El éxito de GNU Radio y Matlab reside principalmente en que proporcionan herramientas fáciles de manejar para la caracterizar señales.

La tecnología Radio definida por software (SDR) tiene como objetivo aprovechar los módulos de hardware programables para construir distintos sistemas usando software de radio basado en arquitecturas abiertas (Abdullah *et al.*, 2010). Un enfoque conveniente es utilizar tarjetas de FPGA para implementar el procesamiento de señales de la capa física para radios definidas por software (Tanvika, 2010). SDR surgió cuando se requirieron nuevos desarrollos de arquitecturas reprogramables con el rendimiento suficiente para realizar las operaciones más comunes de procesamiento de señales de radiocomunicaciones. Muchos de los desarrollos de SDR se han incrementado debido a la necesidad de receptores y transmisores de radio reconfigurables. Por ejemplo, es posible que sea necesario reconfigurar el sistema de radio en un sitio celular para implementar nuevos estándares o protocolos, evitar fallos en el hardware o reconfigurar la red debido a un cambio en el patrón de tráfico.

Existen algunos trabajos y proyectos donde se han realizado simulaciones y pruebas de transceptores de SDR. Inicialmente en junio del 2000 en la Universidad de Twente, R. Schiphorst se presentó una descripción global del concepto de software de radio (Schiphorst, 2000). Además, construyó un transmisor y receptor de radio por software con fines de demostración. De distinta manera, en el año 2013 (Sruthi *et al.*, 2013), se utilizó RTL-SDR (Realtek Software Defined Radio) en la parte del receptor como una alternativa de bajo costo a la USRP. En la parte de transmisión se utilizó un circuito mezclador conectado a una computadora personal (PC) para mapear la señal de banda base a la banda que puede ser recibida por RTL-SDR y fue conectado a otra PC. En la Universidad de Al-Mustansiryah (Abdullah, 2010) se presentó el procedimiento de diseño y los resultados de implementación de una SDR propuesta utilizando una tarjeta de FPGA de la familia Altera Cyclone II. La implementación utiliza el software por bloques de Matlab/Simulink y la tarjeta de desarrollo Cyclone II. Igualmente, en el año 2013, Kumar y Mahammad (Kumar *et al.*, 2013) presentaron el diseño de un sistema de comunicación de múltiples datos basado en SDR, en este trabajo se utilizó el FPGA Spartan-3E. La novedad fue el diseño de un sistema de anuncio de eventos para un campus y el sistema de comunicación que es reconfigurable, este consistió en un módulo transmisor único que se implementa en base al principio SDR y con múltiples receptores, que se encontraban en varias ubicaciones del campus. Uno de los trabajos más recientes dentro del área SDR fue el mostrado en (Nuñez Perez *et al.*, 2020), donde el objetivo fue proporcionar una adecuada integridad de la señal a un amplificador de potencia (PA), este propuso un sistema digital para evitar la degradación en la ruta del transmisor, y se implementa en una tarjeta de FPGA. El sistema ofrece las siguientes características: Generación de señales digitales de modulación de amplitud en cuadratura (QAM) y corrección del desequilibrio en fase/cuadratura (IQ), y por defecto, realiza una extracción del modelo de predistorsión a partir de los datos medidos del PA. Las simulaciones y pruebas se realizaron para verificar efectivamente la linealidad del PA y del sistema de transmisión, fue utilizando señales 256-QAM.

Este artículo está organizado de la siguiente manera: en la sección II se describen las definiciones y conceptos de SDR, Registros de Desplazamiento con Realimentación Lineal (LFSR) y Espectro Extendido por Secuencia Directa (DSSS) que se utilizaron en esta investigación. En la sección III se muestran los resultados de las simulaciones realizadas en Matlab con el sistema DSSS-LFSR, así como la descripción del comportamiento de dos casos usando un LFSR estándar y un LFSR extendido. La sección IV presenta los resultados de implementación en una tarjeta FPGA usando VHDL, se describe además el diseño de una interfaz gráfica que permite desplegar gráficamente el comportamiento de tres casos con

diferente frecuencia, con la flexibilidad de cambiar los parámetros del sistema DSSS-LFSR. Finalmente, las conclusiones se presentan en la sección V.

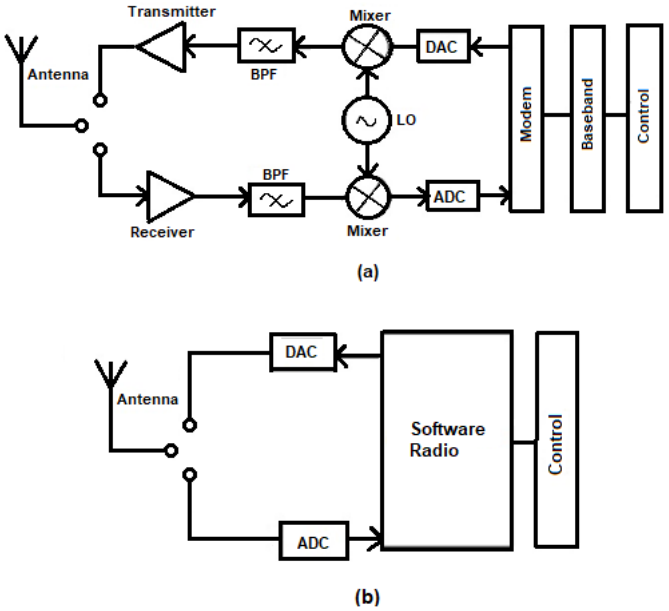


Figura 1: Diagrama de bloques básico de: a) un sistema transceptor tradicional, b) arquitectura de un transceptor SDR.

2. Marco Teórico

2.1. Radio Definida por Software

La Figura 1(a) muestra el diagrama de bloques básico de una radio convencional. En la ruta de recepción de una radio convencional, la señal de RF de radio es recibida por una antena y pasa a través de un filtro de pasa banda que atenúa todas las señales excepto una parte específica del espectro. El ancho de banda de esta etapa limita las frecuencias permitidas de la señal que está siendo recibida. Por ejemplo, las frecuencias centrales comunes para la etapa de IF son 455 kHz y 10,7 MHz para AM y FM comerciales, respectivamente. Del mismo modo, para FM comercial, el ancho de banda es de aproximadamente 100 kHz y para AM está por encima de 5 kHz, en consonancia con el espacio entre canales que es de 200 kHz para AM y 10 kHz para FM. La conversión de frecuencia de la RF transmitida a una frecuencia intermedia (IF) más baja se logra multiplicando la señal de RF por un oscilador local sinusoidal (LO) en el mezclador. Luego un ADC muestrea la salida de la etapa IF final y los datos digitales son procesados por un circuito de procesamiento de señales digitales (DSP). Los componentes desde la antena hasta el ADC son todos circuitos analógicos. Si existen más etapas de conversión descendente, entonces se necesitan más componentes analógicos. Sin embargo, los componentes analógicos tienen limitaciones en la capacidad y flexibilidad de procesamiento de señales, además son costosos. Si se reduce el número de componentes analógicos, resultará en la simplificación del sistema de radio, lo que a su vez dará como resultado una mayor confiabilidad y un costo reducido. Sería mejor si todas las etapas analógicas intermedias pudieran ser reemplazadas por componentes digitales, de modo que la antena esté conectada directamente a un ADC o DAC.

El esquema de funcionamiento de la SDR se presenta en la Figura 1(b), donde los componentes de hardware del receptor van desde la antena hasta los ADC y DAC, en los bloques de Software Radio y de Control, entra la tarjeta de FPGA y la PC que son parte del procesamiento digital en los que se programan y representan los algoritmos que utilizan SDR. La idea detrás de la radio definida por software es trasladar la mayor parte de la funcionalidad de la radio al software. Para la SDR de baja frecuencia, todas las tareas de RF como la conversión ascendente y descendente digital (DUC, DDC, respectivamente por sus siglas en inglés). Estos conversores se utilizan ampliamente en los sistemas de comunicación para escalar la frecuencia de muestreo de las señales. La DUC es necesaria cuando una señal se traduce de banda base a banda de IF. La DDC ocurre cuando una señal se convierte de una banda de IF a una banda base.

2.2. Registros de Desplazamiento con Realimentación Lineal

En el diseño de los sistemas de encriptación de datos, los Registros de Desplazamiento con Realimentación Lineal (o por sus siglas en inglés, (LFSR) *Lineal Feedback Shift Register*), juegan un papel preponderante por ser un mecanismo de generación de datos encriptados en forma aleatoria. No obstante, los LFSR pueden ser modificados, en función de optimizar el aprovechamiento de las capacidades actuales de los procesadores, mediante estructuras de diseño algebraicas, llamadas cuerpos compuestos de Galois. Este nuevo diseño se conoce como Registro de Desplazamiento con Realimentación Lineal extendido (o por sus siglas en inglés, (LFSRe, *Lineal Feedback Shift Register extended*). En los sistemas criptográficos, los datos son generados de forma aleatoria, con secuencias que son determinísticas, o sea, que son finitas. Esto se realiza principalmente utilizando un generador de secuencias pseudo-aleatorias, que utiliza una estructura combinacional denominada LFSR. El mismo tiene una amplia utilización, gracias a las propiedades estadísticas de las secuencias, en ramas como la criptografía, la TV digital, GPS y telefonía móvil. El valor inicial se denomina semilla y, como la forma de operar el registro es determinista, la secuencia de valores producidos está completamente determinada por el estado actual o el estado anterior. La secuencia tiene un periodo de repetición, es decir que la secuencia vuelve a generarse y se repite indefinidamente. Cuando el periodo de repetición es máximo, ese LFSR tiene interés criptográfico.

Un LFSR de longitud L, tendrá L elementos o elementos de retardo. Esto se describe como $S_0, S_1, \cdots, S_{L-1}$, cada uno con la capacidad de portar un bit de información. Un reloj controla el LFSR, y con cada unidad de tiempo transcurrida ocurre lo siguiente: 1) el contenido del elemento S_0 es un bit de salida de la secuencia del LFSR, 2) el contenido de los elementos de Si se mueven para S_{i-1}, para todo $1 \leq i \leq L-1$, c) el contenido de los elementos S_{L-1} se calculan mediante adición módulo-2 (X-OR), con respecto al contenido del resto de los elementos.

De esta manera, los LFSR son dispositivos que permiten encriptar la información mediante algoritmos que utilizan aritmética lógica, a partir de los estados que tienen los registros para un determinado tiempo y los nuevos valores que tienen estos, luego de transcurrido un tiempo o periodo de reloj. En función de la cantidad bits por los que estén compuestos, será

más difícil de encriptar la información que porten los LFSR, pues la capacidad de desencriptar los algoritmos de encriptado depende del período de la secuencia, que es proporcional, en forma exponencial, a la cantidad de bits.

Tradicionalmente los LFSR son componentes básicos de muchos encriptadores o cifradores en flujo, y se han definido sobre los campos de Galois (GF, de *Galois Field*) binarios, es decir, *GF(2)*. Sin embargo, nada impide que estos registros puedan trabajar sobre cuerpos de mayor tamaño, El hecho de realizar registros de desplazamiento extendidos consiste en definir los LFSR sobre cuerpos compuestos de Galois, $GF(2^n)^m$, en lugar de los tradicionales $GF(2^n)$. Los registros de desplazamiento extendidos se definen de igual manera que los tradicionales. Un registro de desplazamiento lineal extendido (LFSRe) de longitud L es una máquina de estado finita que opera sobre un cuerpo finito *Fq*, de característica *p*, donde $q = p^e$, para algún $e \geq 1$. En este caso, $q = 2^n$ y, por tanto, $Fq = GF(2^n)$. En cada unidad de tiempo, o pulso de reloj en las implementaciones hardware, el registro actualiza su estado. Así, el estado S_{t+1} se deriva del estado S_t de la siguiente forma: 1) el contenido de la celda r_0 se convierte en la salida de la unidad de tiempo *t* y entra a formar parte de la secuencia cifrante, 2) el contenido de la celda r_i se mueve a r_{i-1} para cada $i,\ 1 \leq i \leq L-1$, c) el nuevo contenido de la celda r_{L-1} es denominado elemento de realimentación, y se calcula de acuerdo a la siguiente función:

$$S_{t+L} = c_L \sum_{i=0}^{L-1} c_i S_t + i,\ \ c_i \in GF(2^n), \qquad (1)$$

donde $C_0, C_1, \ldots, C_{L-1}$ son los coeficientes de realimentación y la conexión en c_L es la posición de realimentación.

2.3. Transmisión de datos utilizando la técnica de Espectro Extendido por Secuencia Directa

La técnica de Espectro Extendido o SS (por sus siglas en inglés, *Spread Spectrum*) es una herramienta muy empleada en sistemas de radiocomunicaciones para transmisión de datos. Se llama de esa forma pues en el proceso de transmisión el ancho de banda empleado es mucho mayor que el mínimo requerido para la transmisión de la información, para con ello dificultar las interferencias y su posible intercepción. En un sistema donde se emplea un transceptor, la señal antes de ser modulada de forma "extendida", es mezclada por otra proveniente de un generador de secuencias pseudoaleatorias, que a su vez, como se conoce, basa su funcionamiento en un LFSR.

La técnica de espectro extendido puede ocultar una señal para transmitirla en baja potencia. Extendiendo la energía de la señal sobre el ancho de banda disponible y usando la mínima potencia necesaria, la señal puede ocultarse en un canal ruidoso. Esto significa que cualquier interceptor no autorizado tendrá una baja probabilidad de detectar la señal asociada al receptor intencional. El primer requerimiento de este tipo de sistemas es que el ancho de banda de transmisión de la señal debe ser más grande que el ancho de banda mínimo asociado con los datos de información. El segundo requisito es que el ancho de banda de la señal debe extenderse usando una señal extendida o código que es independiente de los datos. Este código tiene propiedades pseudo-aleatorias que le permiten al receptor saber lo que para el código es a priori. La demodulación es entonces realizada situando en correlación el código recibido con una réplica sincronizada en el receptor y por eso se produce la de-extensión (recuperación de los datos originales) de la señal.

Dentro de la técnica de espectro extendido existen dos variantes importantes: Espectro Extendido por Salto de Frecuencia (FHSS, *Frequency Hopping Spread Spectrum*) y Espectro Extendido por Secuencia Directa (DSSS, *Direct Sequence Spread Spectrum*). Ésta última variante se apoya en una técnica de extensión directa, en la cual el espectro del mensaje es extendido usando la multiplicación de la señal por una secuencia pseudo-ruidosa. El DSSS es la más común de las versiones del espectro extendido usadas en la actualidad. En DSSS, la portadora ó señal de datos, es modulada por la secuencia de código pseudoaleatorias, la cual es de una frecuencia mayor que la razón de datos deseada. En la Figura 2 se ilustra un modelo de DSSS.

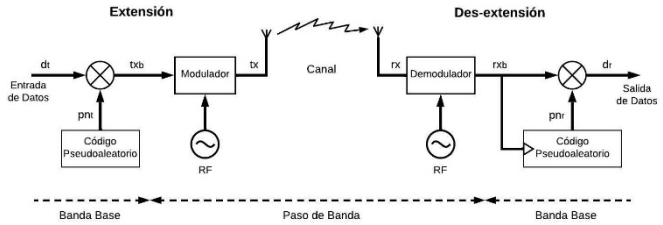


Figura 2: Modelo de Espectro Extendido por Secuencia Directa.

La señal de datos de entrada dt en banda base es multiplicada por una señal de código pseudoaleatoria, cuya forma de onda es binaria, con pulsos de duración *Tc*, llamados tiempo de chip, obteniéndose como resultado una señal txb. Cada pulso de la señal pseudoaleatoria tiene una razón de chip equivalente a $Rc = 1/Tc$. La duración de cada chip es mucho menor que la longitud de los pulsos de los símbolos de información. En el modulador, la señal de información txb emplea una modulación digital de fase, por ejemplo, o cualquier otro tipo de modulación digital. La información se transmite por un canal que puede estar afectado con ruido blanco gaussiano o cualquier otro tipo de interferencia. En el extremo receptor se realiza la operación de demodulación. Posteriormente la señal demodulada rxb se multiplica por la secuencia de datos pseudoaleatorios y de esta forma se obtienen los datos dt a la salida.

Una de las principales aplicaciones del sistema DSSS en las telecomunicaciones, es la posibilidad de utilizar la misma banda de frecuencias para múltiples usuarios que están transmitiendo simultáneamente. Esta es la técnica denominada Acceso Múltiple por División del Código (CDMA, *Code Division Multiple Access*). La idea clave en el sistema CDMA consiste es que a cada usuario se le asigna una secuencia PN diferente y aunque todos los usuarios transmiten simultáneamente por la misma banda de frecuencias, un usuario particular puede extraer de la señal compuesta transmitida, la señal dirigida hacia él utilizando la secuencia PN apropiada. En la práctica, el sistema CDMA demanda un control estricto de la potencia de salida de cada transmisor, esto con el fin de asegurar que la señal de cada transmisor llegue a la antena receptora aproximadamente con la misma potencia. Lo anterior debido a que, si una o más estaciones transmiten con altas potencias, la calidad de la señal en todas las estaciones se degrada rápidamente. Anteriormente se había abordado la relación que guardan los LFSR con el desarrollo

de aplicaciones de la SDR, y que tuvieran como razón fundamental la seguridad en la transmisión de información. Un caso empleado son los modelos de transmisión de datos que utilizan la técnica DSSS.

3. Simulación en Matlab/Simulink DSSS con LFSR

La Figura 3 muestra la implementación de DSSS con LFSR. Se puede notar que el LFSR introducido genera secuencias codificadas que se combinan con los datos aleatorios y son recibidos bajo una mayor seguridad en la información.

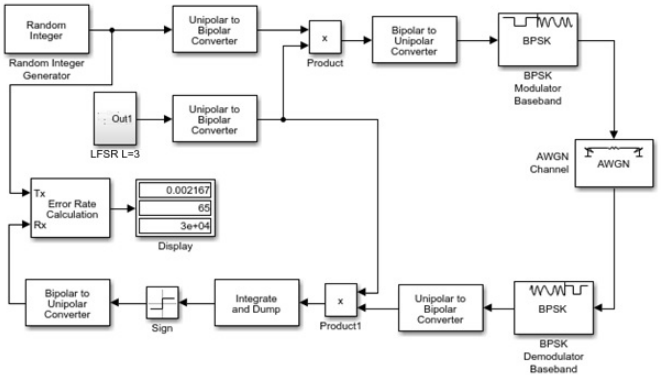


Figura 3: Modelo de transmisión de datos utilizando DSSS con LFSR L = 3.

En este caso, el LSFR es un modelo convencional con L = 3. Los datos son generados por el bloque *Random Integer Generator*, de la biblioteca *Communications System Toolbox* de Matlab/Simulink. Su función es generar de forma aleatoria distribuciones de números enteros y como la transmisión es binaria, las secuencias que se transmiten se ubican entre "0" ó "1".

Dentro de los parámetros de configuración del bloque se encuentran el tiempo de muestreo (*sample time*), es el retardo en generar un bit y el consecutivo; y el número M-ario (*M-ary number*) es el entero positivo que determina el rango de los valores de salida. Para el primero de los parámetros se adoptó 1/1000 y en el segundo $M = 2$.

Dentro de la etapa de transmisión, las secuencias comandadas por el *Random Integer Generator* son convertidas a señales bipolares, al igual que las generadas por el LFSR mediante el convertidor de señales unipolares a bipolares de la sub-biblioteca *Utility Blocks*, luego ambas señales son multiplicadas y convertidas a señales unipolares. En esta operación los bits de información son mezclados con secuencias codificadas, lo que implica que la secuencia resultante ya se encuentra debidamente cifrada. La secuencia por transmitir es modulada utilizando modulación digital de fase, en este caso BPSK (*Binary Phase Shift Key*). Los datos pasan a través de un canal con ruido blanco gaussiano, para ello se usa el bloque *AWGN Channel*.

En la etapa de recepción, se realiza la operación inversa, pues los datos son demodulados utilizando BPSK, después se convierten a señales bipolares y se multiplican por la secuencia codificada del LFSR. El proceso de filtrado se realiza empleando el bloque *Integrate and Dump* de la sub-biblioteca *Comm Filters*. El bloque crea una suma acumulativa de la señal de entrada en tiempo discreto. Cuando comienza la simulación, el bloque desecha el número de muestras que se especifica en el parámetro de Desplazamiento (*Offset*). Finalmente, los datos se vuelven a convertir a una señal unipolar obteniendo así la señal original que contiene los datos transmitidos.

En la Figura 4 se muestra el resultado de la simulación, donde la primera señal corresponde a los datos enviados, la segunda a la salida del bloque *BPSK Modulator Baseband*, la tercera señal a la recibida por el receptor en la salida del bloque *AWGN Channel* y finalmente la cuarta señal corresponde a los datos recibidos luego de demodular y extraer los datos.

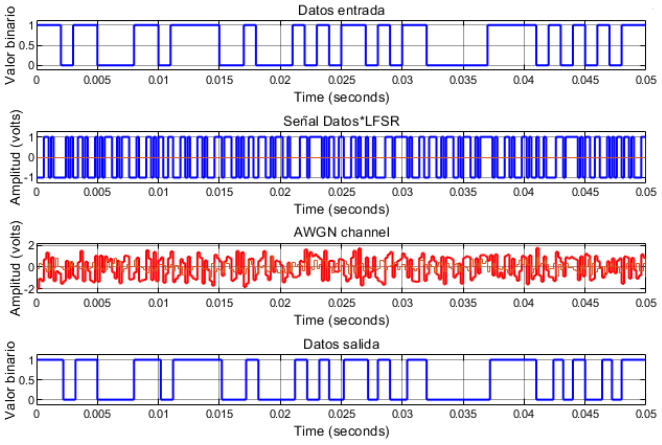


Figura 4: Señales obtenidas del modelo de transmisión de datos utilizando DSSS con LFSR L=3.

En las Figuras 5 y 6 se observa el diagrama a bloques y los resultados de la simulación para el modelo DSSS con un LFSRe, respectivamente.

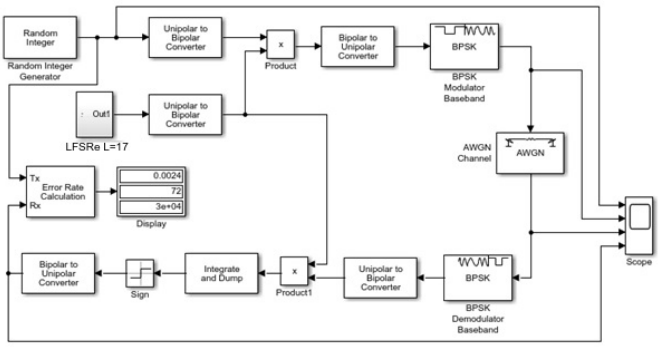


Figura 5: Modelo de transmisión de datos utilizando DSSS con LFSRe L=17.

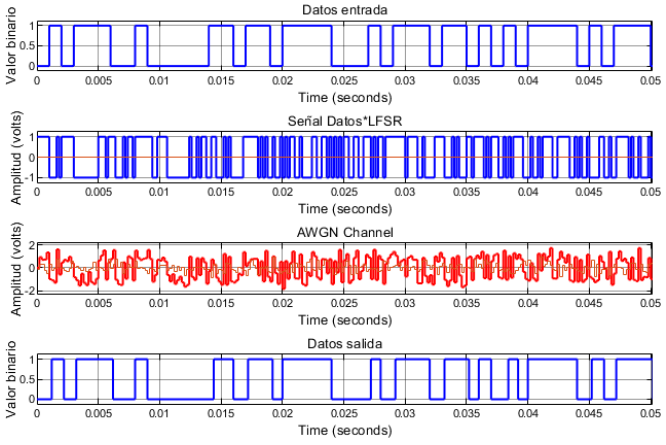


Figura 6: Señales obtenidas del Modelo de transmisión de datos utilizando DSSS con LFSR L=17.

Donde la dimensión del cuerpo base es 8 y la dimensión de la extensión del cuerpo base es $L = 17$, lo que da como resultado una longitud clave efectiva de 136 bits, es decir un $GF(2^8)^{17}$. Es importante destacar que los parámetros del

modelo de transmisión para el LFSRe están bajo las mismas condiciones del modelo de LFSR convencional. Al final se determina la tasa de error de bits BER (*Bit Error Rate*) mediante el bloque *Error Rate Calculation* de Matlab / Simulink. Este bloque posibilita el sincronismo del sistema, a partir de los datos recibidos y enviados, ajustando los tiempos en que son enviados y recibidos los bits.

4. Resultados

En la emulación del sistema de transmisión de datos digitales con técnica DSSS en la tarjeta de FPGA, es necesario construir el modelo del sistema usando el software *System Generator for DSP* de Xilinx. La Figura 7 muestra el modelo general para transmisión de datos digitales con técnica DSSS.

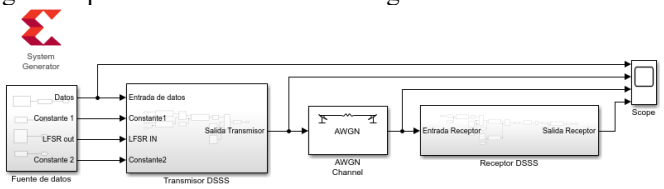


Figura 7: Modelo general del sistema DSSS.

Los cuatro subsistemas implementados son: 1) fuente de datos, 2) transmisión, 3) canal de ruido y 4) recepción. El primer subsistema, posee al generador aleatorio de datos digitales y al LFSR, previamente configurados. En el *blockset* de Xilinx existe un bloque LFSR propiamente diseñado, el cual fue aprovechado para incluirse en la fuente de datos. El LFSR es ajustado introduciendo los valores de los coeficientes del polinomio de realimentación en notación hexadecimal, así como los valores iniciales de los registros en igual notación.

Cabe mencionar que se utilizaron dos modelos de transmisión DSSS iguales, con la diferencia del tipo de LFSR aplicado: para el LFSR ordinario se utilizó uno de tamaño de 3 bits, mientras que para el LFSRe se utilizó un tamaño de 8 bits. La Figura 8 muestra el subsistema de transmisión DSSS donde se realizan los procesos de conversión unipolar a bipolar de la señal de datos digitales a transmitir y de los datos digitales del LFSR por medio de los bloques *MCode* y *MCode1*, respectivamente.

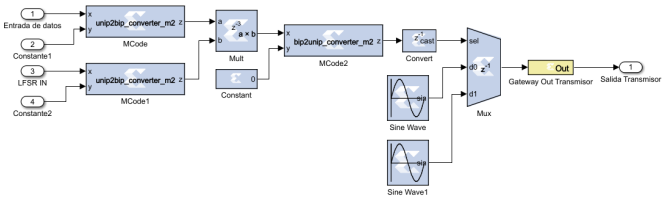


Figura 8: Vista interna del subsistema Transmisor DSSS.

Posteriormente estas señales se multiplican usando el bloque *Mult*, ocurriendo así la encriptación de la información original y luego esta señal bipolar resultante se convierte a señal unipolar con el bloque *MCode2*. Las conversiones de las señales son posibles gracias al bloque *MCode* de la biblioteca *Control Logic* de Xilinx. El bloque ejecuta el M-código para calcular los rendimientos de este durante la simulación y se traduce de una forma transparente equivalente al comportamiento VHDL cuando se genera en el hardware.

Una vez que la señal de datos encriptada se convierte a señal unipolar, el siguiente paso es la modulación BPSK, la cual es generada con el bloque *Convert*, *Mux* y dos bloques *Sine Wave*, de la biblioteca DSP de Xilinx. Con un bloque se genera una portadora sinusoidal y con el otro bloque se genera la misma portadora sinusoidal, pero con un desfase de 180°, donde además ambas sinusoidales tienen una frecuencia de 1 MHz. Para simular el canal de transmisión de la señal, se utiliza el bloque *AWGN Channel* para agregar ruido gaussiano a la señal de su entrada, que suele ser la forma típica de modelar el ruido.

En el subsistema de Receptor DSSS, intervienen los mismos bloques de la parte de transmisión, como lo muestra la Figura 9. La señal es recibida por un bloque *Gateway In* para devolver estos datos al dominio de los bloques de la librería de Xilinx, luego la señal pasa por un convertidor de bipolar a unipolar con el bloque *MCode*. Posteriormente se hace la demodulación BPSK con los bloques *Convert*, *Mux* y *SineWave*, estos datos pasan por un convertidor de unipolar a bipolar, al igual que los datos del LFSR y se multiplican usando el bloque *Mult*. Luego el resultado se presenta usando el bloque *Gateway Out* y se filtra la señal de espectro extendido con los bloques de Simulink, *Integrate and Dump* y *Sign*. Al final los datos ingresan a un bloque *Gateway In* y pasan a un convertidor de bipolar a unipolar y se obtienen los datos originales procesados por el transmisor DSSS.

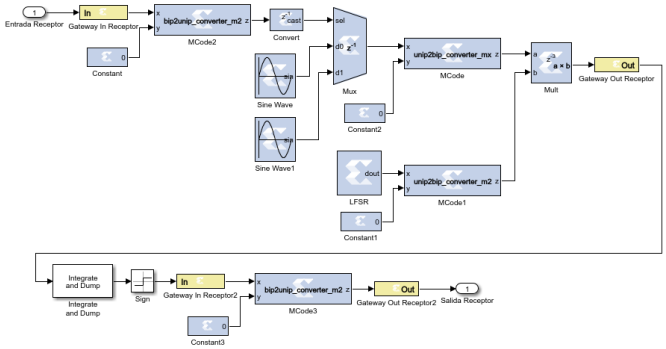


Figura 9: Vista interna del subsistema Receptor DSSS.

La Figura 10 muestra la simulación del sistema DSSS con bloques de Xilinx *System Generator* en Simulink.

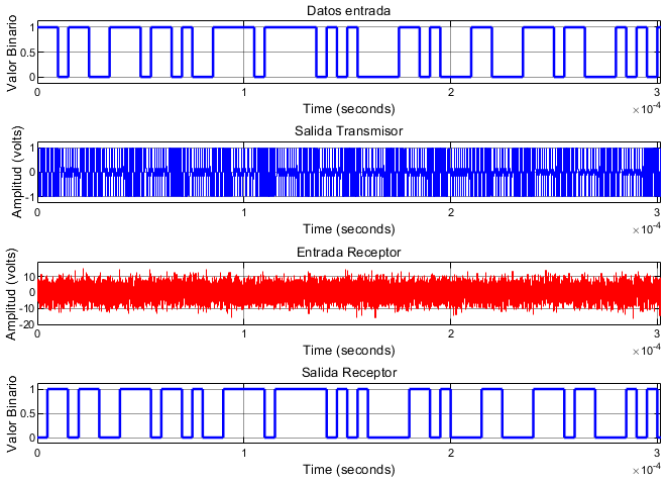


Figura 10: Simulación del transceptor DSSS con el *System Generator*.

La primera señal representa los datos provenientes de la fuente de datos, la segunda a la salida del transmisor DSSS (modulación BPSK), la tercera son los datos en la entrada del

receptor DSSS (después de integrar el ruido gaussiano) y la última señal los datos en la salida del receptor DSSS, tal como se mostró la Figura 7 en el orden de las señales introducidas al bloque *Scope*. Los resultados muestran un buen rendimiento al no existir perdida de datos, con la única observación que en la salida del receptor se obtuvo un desfase de un dato digital, pero aun así la salida del receptor DSSS muestra cero errores comparando con la señal de datos de entrada.

4.1. Análisis del transceptor DSSS con LFSR

En la emulación del sistema transceptor DSSS de forma completa en la tarjeta FPGA Artix-7 de Xilinx, se empleó el diagrama mostrado en la Figura 5. El bloque *System Generator* se puede configurar para compilar el modelo realizado en Simulink en alguna tarjeta de FPGA específica del fabricante Xilinx. En el caso de este trabajo de investigación se utilizó la configuración de una tarjeta de evaluación Artix-7 AC701, que cuenta con el chip FPGA xc7a200t-2. En este caso se configuró una frecuencia máxima de 100 MHz. Es importante recalcar que el periodo del sistema en Simulink es un dato importante, ya que de este dependen los tiempos de muestreo de los bloques HDL, pues impone la máxima frecuencia para todos los bloques de la librería de Xilinx. La configuración de este posibilita introducir los parámetros necesarios tales como: la frecuencia de reloj de la FPGA de 25 ns y la asignación del pin de reloj, que será M21, para que posteriormente el sistema pueda compilar en el software *System Generator* de Xilinx. Se analizaron primero los resultados del LFSR convencional y luego el LFSRe. La comparación cuantifica el retardo, recursos lógicos utilizados de la FPGA y los requerimientos de potencia.

El tiempo total empleado para la instalación de la ruta que comprende desde la fuente hacia el destino son 18.557 ns, mientras el retardo de camino de los datos es de 6.078 ns. La tarjeta de FPGA tiene un requerimiento de tiempo de 25 ns como se ajustó en el bloque *System Generator*.

En la Figura 11 se aprecian los detalles del consumo de potencia del dispositivo por sus diferentes partes. Se puede notar que el rango máximo de potencia del sistema es de 154mW, para una temperatura ambiente de 25°C, y la temperatura de unión del chip resulta ser de 25.3ºC en funcionamiento.

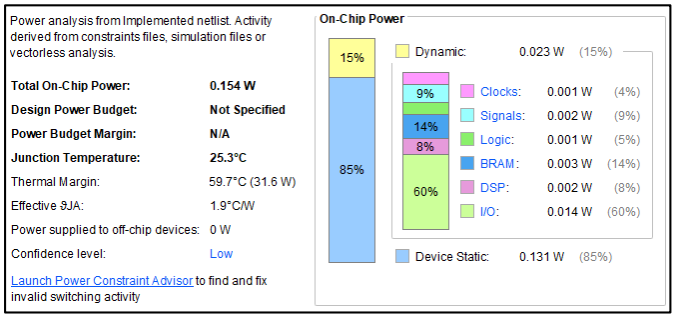


Figura 11: Reporte de potencia del transceptor DSSS con LFSR.

Al analizar el reporte obtenido por el diseño del modelo DSSS con LFSR se obtiene que el requerimiento de recursos de la FPGA para el diseño de este sistema es mínimo, puesto que no se requiere utilizar el 1% de *slices* como LUT o incluso *Flip Flops*, esto demuestra la eficiencia del diseño.

4.2. Análisis del transceptor DSSS con LFSRe

El procedimiento para LFSRe es el mismo que el explicado en la sección 4.1, por tanto, los pasos explicados con anterioridad son válidos para este caso. En este caso el tiempo máximo de configuración de camino de los datos es de 18.911 ns, con retardo de 5.648 ns. Si bien es cierto que la capacidad de los microprocesadores actuales soporta los elementos del LFSRe, este genera secuencias que se ven incrementadas a una tasa de 8 bits por cada pulso de reloj. Tal como se puede observar en la comparación, es un resultado mayor esto debido a los recursos a emplear, los cuales exigen de este modelo mayor requerimiento de tiempo. Aunque para efectos prácticos, la diferencia de tiempo no es tan significativa, sólo de 0.354 ns, por lo que este modelo es viable.

Al realizar la comparación de ambos modelos se obtuvo que el modelo DSSS con LFSR utiliza dos redes físicas (*Physical Nets*), 777 redes (*Nets*), 97 I/O ports y 709 Celdas (Cells). Mientras que el modelo con LFSRe utiliza dos redes físicas (*Physical Nets*), 788 redes (*Nets*), 97 terminales I/O y 725 Celdas, observando así que este modelo utiliza ligeramente más recursos lógicos.

El análisis de potencia se realiza nuevamente empleando el *Report Power*. Se pudo conocer que el rango máximo de potencia del sistema es de 154 mW, para una temperatura ambiente de 25°C la temperatura de unión del chip resulta ser de 25.3ºC en funcionamiento. En la Figura 12 se pueden apreciar más detalles acerca de la disipación de potencia del dispositivo por sus diferentes partes. Al hacer la comparación se obtuvo que el análisis de potencia de ambos modelos arroja el mismo resultado, puede ser debido a que como se mostró en la parte anterior, el modelo con LFSRe no necesita de una cantidad elevada de recursos adicionales.

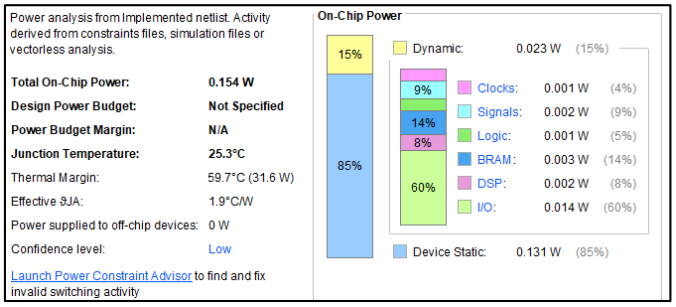


Figura 12: Reporte de potencia del transceptor DSSS con LFSRe.

El reporte del diseño del modelo del transceptor DSSS con LFSRe muestra que el requerimiento de recursos de la FPGA para el diseño de este sistema es mínimo, al igual que en LFSR en la sección 4.1. Estos resultados demuestran la viabilidad de implementar los modelos de transmisión de datos que utilizan LFSRe, al igual que los LFSR convencionales poseen los mismos tiempos de camino para los datos dentro del hardware. Además de tener iguales requerimientos y consumo de potencia, con el inconveniente de utilizar más recursos lógicos de la FPGA. Así por medio de la emulación en la FPGA, se demuestra la ventaja de utilizar LFSRe para aplicaciones en encriptación de información en un transceptor DSSS.

En la Tabla 1 se presenta una comparación a detalle de los recursos utilizados del hardware por ambos modelos y el total de recursos disponibles en la FPGA.

Tabla 1: Comparación de recursos utilizados entre ambos modelos

Nombre de recursos lógicos	Recursos totales FPGA Artix-7	Modelo con LFSR	Modelo con LFSRe
Slice LUT	133,800	215 (0.16%)	221 (0.17%)
Slice Flip Flop	267,600	247 (0.09%)	257 (0.10%)
Slices usadas	33,450	95 (0.25%)	100 (0.30%)
Block RAM	365	4 (1.10%)	4 (1.10%)
BUFG	32	1 (3.13%)	1 (3.13%)
IO	400	97 (24.25%)	97 (24.25%)
DSP	740	10 (1.35%)	10 (1.35%)

4.3. Interface gráfica de usuario

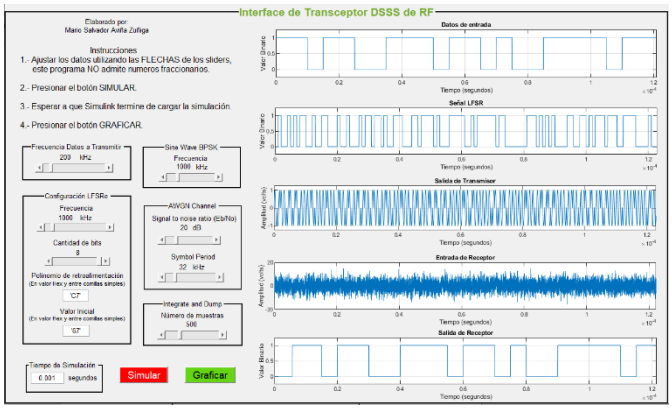
Seguidamente de la comparación de modelos de transceptor DSSS y de la comprobación de la emulación en una FPGA, se procedió a crear una interfaz gráfica de usuario (GUI) utilizando la herramienta GUIDE de Matlab para poder modificar las características del transceptor DSSS. La interfaz permite ajustar los diferentes componentes del transmisor y receptor, como lo son la frecuencia de los datos a enviar, la frecuencia del LFSRe, así como las características del polinomio de valor inicial y el valor del polinomio de retroalimentación. También se ajusta la frecuencia de las señales sinusoidales de la modulación BPSK, las características del bloque *AWGN Channel* y el bloque de integración *Integrate and Dump*, además cuenta con visualización de las diferentes señales del sistema transceptor de RF para verificar el óptimo funcionamiento del sistema, a las cuales se les puede ver con más detalle si se utiliza la opción de zoom en la parte superior derecha de la imagen de la señal.

La Figura 13 muestra la interfaz para una transmisión de datos usando tres casos: a) 200 kHz, b) 400 kHz y c) 1 MHz, respectivamente.

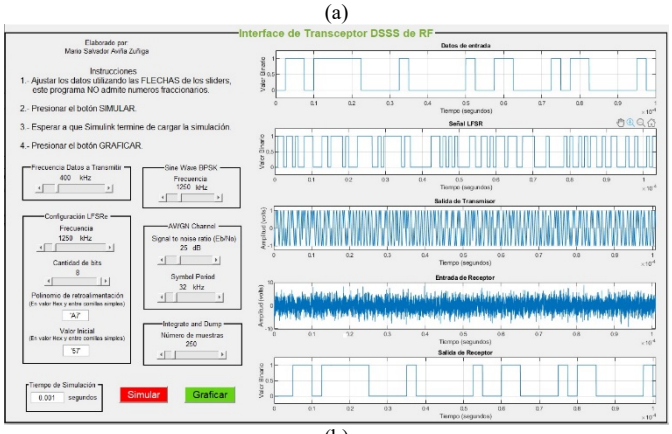
Cada uno de estos sistemas se diseñó a partir del mismo modelo de transceptor DSSS con LFSRe explicado con anterioridad. Se puede observar que en los tres modelos la señal desencriptada corresponde a la señal de datos digital inicial, aunque con un retraso de un dato digital, esto debido al retardo de muestreo del bloque *Integrate and Dump*, puesto que el valor del número de muestras de este bloque se ajusta de acuerdo con la frecuencia de los datos a transmitir. La causa de esto coincide con dicho retraso temporal, aunque para aplicaciones prácticas dicho retraso pasará desapercibido a final de cuentas.

En la Tabla 2 se muestra las comparaciones de los resultados obtenidos al compilar los modelos al lenguaje VHDL, y luego hacer la emulación en la FPGA Artix-7. Los resultados demuestran la versatilidad del programa y la funcionalidad de la modificación por medio del software aplicado en un sistema transceptor DSSS basado en una tarjeta de FPGA.

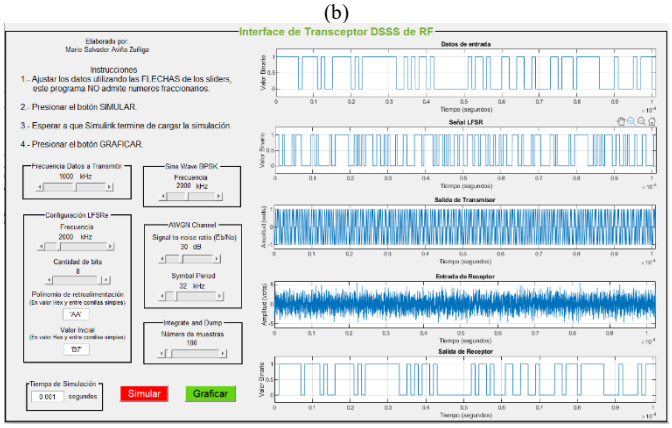
Con esto se demuestra que el transceptor DSSS, diseñado con el *blockset* de *System Generator*, funciona de manera efectiva usando lenguaje VHDL. Además, se demuestra que un transceptor de este tipo puede ser modificado para distintas aplicaciones con tan solo software.



(a)



(b)



(c)

Figura 13: Interfaz de transceptor DSSS para transmisión de datos a la frecuencia: a) 200 kHz, b) 400 kHz, c)1 MHz.

Tabla 2: Comparación de recursos usados entre los 3 ejemplos de transmisión

Nombre de recursos lógicos	Recursos totales FPGA Artix-7	Modelo con LFSRe	Modelo con LFSR	Modelo con LFSR
Slice LUT	133,800	221 (0.17%)	223 (0.17%)	226 (0.17%)
Flip Flop	267,600	257 (0.10%)	260 (0.10%)	266 (0.10%)
Slices usadas	33,450	100 (0.30%)	101 (0.30%)	103 (0.31%)
Block RAM	365	4 (1.10%)	4 (1.10%)	4 (1.10%)
BUFG	32	1 (3.13%)	1 (3.13%)	1 (3.13%)
IO	400	97 (24.25%)	97 (24.25%)	97 (24.25%)
DSP	740	10 (1.35%)	10 (1.35%)	10 (1.35%)

5. Conclusiones

En este trabajo se llevó a cabo la emulación de un sistema transceptor de datos digitales utilizando la técnica de Espectro Extendido por Secuencia Directa en una FPGA Artix-7 de Xilinx utilizando los LFSR estándar y extendido. Se encontró

que el sistema implementado con LFSR extendido presenta mejores resultados tanto en simulación usando Matlab-Simulink como en la emulación en FPGA empleando lenguaje VHDL y Vivado de Xilinx. Este es un resultado esperado ya que el LFSRe posee dos o más retroalimentaciones con compuertas lógicas AND y X-OR a lo largo del registro, mientras que el LFSR estándar solo posee una retroalimentación X-OR al final del registro, evidenciándose la superioridad del primero a pesar de ser una arquitectura más compleja. Además, ambos modelos utilizan por lo menos el 1% de los recursos lógicos (*Slices*) del FPGA, 2% de memoria RAM al usar tan solo 4 de 365 bloques disponibles, el 25% de las terminales de I/O, y el 2% de los DSP al ocupar 10 de 740 disponibles. De los análisis, se obtuvo que el transceptor a temperatura ambiente de 25°C, con la potencia consumida de 0.154 W, la temperatura en el chip termina siendo de 25.3°C. Además, con el análisis temporal se obtuvo un retraso en la ruta de datos es de 5.5 ns en promedio. Por último, se diseñó una interfaz gráfica, en ésta se ejecuta el modelo DSSS-LFSR para la modificación de los parámetros, para transmitir y modular los datos digitales, recibir la señal, filtrarla, demodularla y hacer un re-muestreo de esta. La interfaz de usuario permite modificar el transceptor para sintonizar diferentes frecuencias en un rango específico, y reconfigurar el sistema.

Agradecimientos

Los autores agradecen el Instituto Politécnico Nacional por el apoyo recibido mediante el proyecto SIP 20220014.

Referencias

Abdullah, H. N., Hadi, H. A., (2010). “Design and Implementation of FPGA based Software Defined Radio Using Simulink HDL Coder”, CUAS Engineering and Technology Journal, Vol. 28, No. 23, pp. 6750-6768.

Abdulsatar, H. S., Tijil, K. J., Jryian, A. H., (2013). “Low power Transceiver Structure for Wireless and Mobile Systems Based SDR Technology Using MATLAB and System Generator”, International Journal of Engineering and Advanced Technology (IJEAT), vol. 3, no 2, pp 162-167.

Abramson, N., (1985). “Development of the ALOHANET”, IEEE Transactions on Information Theory, vol. 31, no. 2, pp. 119-123.

Feng, Z., (2013). “A Software Defined Radio Implementation Using MATLAB”, Tesis de grado de maestría en tecnologías de la información, Vaasa University of Applied Sciences.

Frenkiel, R., Schwartz, M., (2010). “Creating Cellular: A history of the AMPS project (1971-1983)”, IEEE Communications Magazine, vol. 48, no. 9, pp. 14-24.

Kleinrock, L., (2010). “An Early History of the Internet [History of Communications]”, IEEE Communications Magazine, vol. 48, no. 8, pp. 26-36.

Kumar, P., Mahammad, N., (2013). “SDR based Multi Data Communication System Design”. International Conference on Design and Manufacturing, IConDM 2013, Chennai, India.

North, R., Browne N., and Schiavone L., (2006). “Joint tactical radio system – connecting the GIG to the tactical edge,” in Military Commun. Conf., 2006.MILCOM 2006, pp. 1-6, Washington, DC, USA.

Nuñez Perez, J. C., Juarez Cazares, S. A., Galaviz Aguilar, J. A., Sandoval Ibarra, Y., Perez Pinal, F. J., Tlelo Cuautle E., (2020). “FPGA-based system for effective IQ imbalance mitigation of RF power amplifiers”. Int J Circ Theor Appl., pp. 1-13.

Olivieri, S. J., (2011). “Modular FPGA-Based Software Defined Radio for CubeStats”, Master of Science thesis, Worcester Polytechnic Institute.

Raychaudhuri, D., Mandayam, N., (2012). "Frontiers of Wireless and Mobile Communications," Proceedings of the IEEE, vol. 100, no. 4, pp. 824-840.

Schiphorst, R., (2000). “Demonstration of the Software – Radio Concept”, Master thesis, University of Twente, Department Of Electrical Engineering, Signals & System Netherlands.

Sruthi, M. B., Abirami, M., Manikkoth, A., Gandhiraj, R., and Soman, K. P., (2013). “Low cost digital transceiver design for Software Defined Radio using RTL-SDR”. International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing, Kottayam, India, pp. 852-855.

Stephens, R., Salisbury, B., and Richardson, K., (2006). “JTRS infrastructure architecture and standards,” MILCOM 2006 - 2006 IEEE Military Communications conference, Washington, DC, USA, pp. 1-5.

Tanvika, M. P., (2010). “Implementation of a Software Defined Radio on FPGAs using System Generator”, Tesis de Maestría en Ciencias, University of Tennessee at Chattanooga.

Tucker D. C., and Tagliarini, G. A., (2009). “Prototyping with GNU Radio and the USRP - where to begin”, IEEE Southeastcon, Atlanta, GA, USA, pp. 50-54.

Ulversoy, T., (2010). “Software Defined Radio: Challenges and Opportunities”. IEEE Communications Surveys & Tutorials, vol.12, no.4, pp. 531–550.