AVANCES
EN CIENCIAS E INGENIERÍAS
ARTÍCULO/ARTICLE SECCIÓN/SECTION C

Simulación e implementación en FPGA de un esquema de codificación del canal sujeto al estandar de Wimax

José Andrés Marzo Icaza* **Rebeca Leonor Estrada Pico**

Facultad de Ingeniería en Electricidad y Computación (FIEC)
Escuela Superior Politécnica del Litoral(ESPOL)
Campus Gustavo Galindo, vía Perimetral Km. 30.5, Apartado 09-01-5863, Guayaquil, Ecuador
**Autor principal/Corresponding author, e-mail: jmarzo@espol.edu.ec*

Editado por/Edited by: R. Játiva, M.Sc.
Recibido/Received: 08/19/2010. Aceptado/Accepted: 03/01/2011.
Publicado en línea/Published on Web: 06/30/2011. Impreso/Printed: 06/30/2011.

Abstract

This paper describes the design, simulation and implementation on a FPGA of a channel encoder for Wimax, focusing on the IEEE 802.16-2004 standard which represents the fixed implementation and it is part of research in the field of fixed wireless access networks Broadband without line of sight. The paper presents the main features used in Wimax for transmitting and receiving also the operation of each of the blocks used for error correction.

This project presents an implementation using FPGA-based design model, using the System Generator software with Matlab and Simulink, to obtain data that allow us to verify the operation of the proposed design according to the standard specifications and also analyze the ability for error correcting using BER vs. SNR curves and constellations to the channel output to justify the use of the system design.

Keywords. Wimax, FEC, channel encoder, FPGA, System generator.

Resumen

El presente trabajo describe el diseño, simulación e implementación en un FPGA de un Codificador de Canal para su uso en un sistema Wimax, enfocándose en el estándar IEEE 802.16-2004 el cual representa la implementación fija y forma parte de la investigación en el campo de redes de acceso fijo inalámbrico de banda ancha sin línea de vista. El trabajo presenta las principales características usadas en Wimax para la transmisión y recepción además del funcionamiento de cada uno de los bloques usados para la corrección de errores.

Este proyecto presenta una implementación en FPGA que utiliza diseño basado en modelo, y usa el software System Generator junto a Matlab y Simulink, para obtener los datos que nos permitan comprobar el funcionamiento del diseño propuesto de acuerdo a las especificaciones del estándar, y además analizar la capacidad de corrección de errores mediante el uso de curvas BER vs SNR y de las constelaciones a la salida del canal para justificar el uso del sistema diseñado.

Palabras Clave. Wimax, FEC, codificador de canal, FPGA, Generador de sistema.

Introducción

Desde que comenzaron a usarse redes WLAN, éstas han evolucionado para soportar cada vez un mayor número de aplicaciones más exigentes y tasas de datos superiores. Como consecuencia de esta evolución nació en el año 2001 Wimax (Worldwide Interoperability for Microwave Access), con el estándar IEEE-802.16 [1]. Wimax soporta comunicaciones sin línea de vista y resulta una solución para sistemas de última milla en lugares donde los costos de implementación y mantenimiento de tecnologías como DSL no justifiquen la inversión.

El uso de herramientas programables reconfigurables, como el FPGA, son una gran ayuda para el diseño de sistemas como Wimax. Las ventajas que éstas ofrecen son las siguientes [2]:

- **Velocidad de procesamiento:** Wimax posee requerimientos superiores en lo que se refiere al manejo de las tasas de datos y la velocidad de procesamiento del sistema.
- **Flexibilidad:** Wimax es una tecnología que está en

ISSN 1390-5384

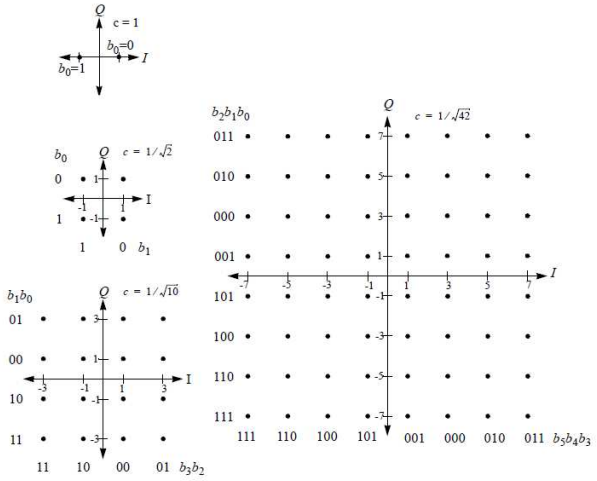

Figura 1: Constelaciones usadas en el estándar IEEE 802.16-2004

constante evolución, por lo que es necesario que el FPGA sea flexible para poder reprogramar el diseño hasta obtener el producto final.

- **Mercado:** Al ser Wimax una tecnología emergente, el tiempo que los productos lleguen al mercado es un factor clave. Las herramientas de desarrollo para los FPGAs, y la existencia de Cores ya diseñados y optimizados, logran que el tiempo de desarrollo sea menor.

Fundamentos Teóricos

El desarrollo de este proyecto se realizó en base al estándar 802.16-2004. Las características consideradas en el diseño se muestran a continuación.

Etapas Generales del estándar

Para realizar este trabajo se considera Wireless MAN-OFDM que utiliza una transformada rápida de Fourier de 256 puntos y opera en la banda de frecuencia de 2 a 11 GHz. El estándar fijo de Wimax provee servicio para un área de 5 Km, permite una tasa de datos máxima de 70 Mbps con un ancho de banda del canal de 20MHz, y ofrece a los usuarios conectividad de banda ancha sin la necesidad de tener línea de vista con la estación base.

La corrección de errores (Forward Error Correction, FEC), se realiza en 2 fases; primero pasando por un codificador exterior Reed-Solomon y luego por un codificador interior Convolucional. El codificador Reed-Solomon se encarga de corregir errores de ráfaga a nivel de byte, lo cual es muy útil en presencia de propagación multicamino. El codificador convolucional corrige errores de bits independientes. Funcionalidades de "puncturing"se aplican en el codificador convolucional para variar la capacidad de corrección de errores, eliminando cierto número de bits al momento de la transmisión.

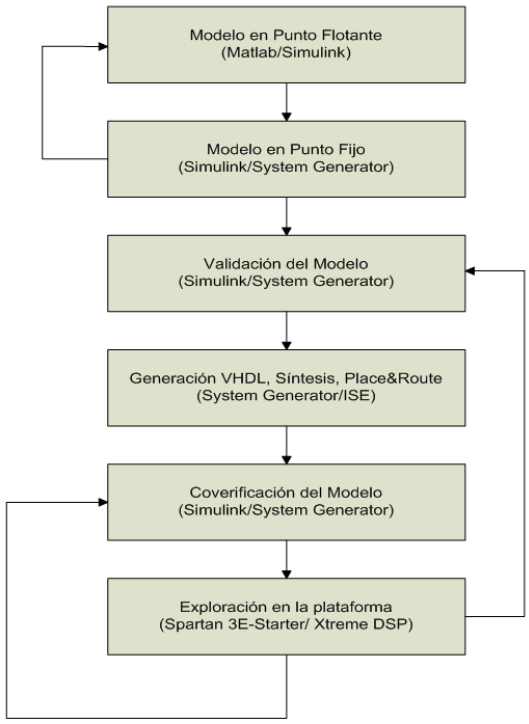

Figura 2: Metodología usada en el diseño basado en modelo.

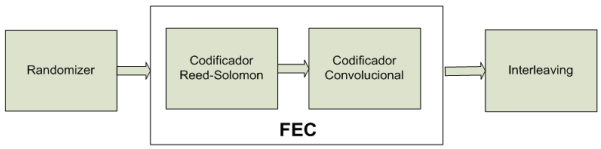

Figura 3: Diagrama de Bloques del Codificador del Canal

Las modulaciones especificadas tanto para el enlace de bajada “downlink” como para el de subida (“uplink”), son BPSK, QPSK, 16-QAM y 64-QAM y sus constelaciones se muestran en la Figura 1. Las opciones de FEC son emparejadas con cada esquema de modulación. El estándar especifica siete combinaciones de modulación y tasas de codificación, las cuales se seleccionan para dar comunicación a cada usuario, lo que es conocida como modulación adaptativa. La tasa de codificación se refiere a la relación de los datos sin codificar para los datos codificados, por lo cual se tiene que decidir si lo que se quiere es tener una tasa de datos alta o una mayor robustez del sistema. La tabla 1 muestra las combinaciones descritas.

Metodología

Para implementar el sistema se sigue el diseño basado en modelo que se ilustra en la Fig. 2 [3]. En particular, el entorno escogido para realizar la implementación de este proyecto, es el software System Generator de Xilinx en conjunto con Matlab y Simulink. Además, para realizar la experimentación en hardware, se utiliza la tarjeta de prototipado de Digilent Spartan 3E-Starter, la cual

ID	Modulación	Bloque de entrada (bytes)	Bloque de salida (bytes)	Tasa de codificación total
0	**BPSK**	12	24	1/2
1	**QPSK**	24	48	1/2
2	**QPSK**	36	48	3/4
3	**16-QAM**	48	96	1/2
4	**16-QAM**	72	96	3/4
5	**64-QAM**	96	144	2/3
6	**64-QAM**	108	144	3/4

Tabla 1: Perfiles de modulación y codificación para el estándar IEEE-802.16-2004

ID	Bloque de entrada k'	Bloque de salida n'	Símbolos que pueden ser corregidos t'
1	32	24	4
2	40	36	2
3	64	48	8
4	80	72	4
5	108	96	6
6	120	108	6

Tabla 2: Variaciones al Codificador de Reed-Solomon

dispone del FPGA donde se implementó el diseño y en el cual se han realizado todas las pruebas de verificación de funcionamiento.

Codificación y decodificación del canal.

El proceso de codificación de canal se encarga de asegurar una transmisión fiable en la comunicación de los datos, protegiendo a estos de degradaciones causadas por el canal. Para lograrlo se añade información redundante a los datos transmitidos de manera que el decodificador pueda interpretarlos para corregir los errores causados por el medio en que son transmitidos.
Esto se realiza por medio de un FEC (Forward Error Correction) el cual detecta y corrige la información recibida.

Diseño del Codificador

El estándar especifica el diagrama de bloques de la Figura 3 para el codificador de canal.

El primer bloque está constituido por un Randomizer que se utiliza para evitar largas secuencias de unos y ceros. Se implementa utilizando un Registro de Desplazamiento con Retroalimentación Lineal (LFSR en la literatura ingles), utilizando el polinomio $1 + x^{14} + x^{15}$, y que se ha implementado conforme se muestra en la Figura 4.

La salida del randomizer debe ser concatenada con un byte 0x00 ("zero tailing") el cual se encarga de inicializar el codificador convolucional en cada ráfaga de datos.

Para la etapa del FEC, primero se tiene un codificador Reed-Solomon el cual trabaja a nivel de bytes. Se utiliza un codificador nativo RS(255,239,8) y se utiliza esquemas de shortening y puncturing para variar la capacidad de corrección del código en base al perfil de codificación usado, como se muestra en la Tabla 2. Los polinomios de generación de código y de generación de campo se expresan a continuación:

$$g(x) = (x - a^{\circ})(x - a^{1})(x - a^{2})...(x - a^{2t-1}) \quad (1)$$

$$p(x) = x^8 + x^4 + x^3 + x^2 + 1 \quad (2)$$

La implementación se esquematiza en la Fig. 5, donde el primer bloque se encarga del "shortening" y de la codificación de los datos (Fig. 6), el segundo bloque realiza el proceso de "puncturing" (Fig. 7), y finalmente se reordenan los datos para transmitir los símbolos de paridad antes de los bytes de información, tal como lo especifica el estándar. Se recalca que se utiliza el diseño usado en el interleaver para el reordenamiento de los datos, y que en las Figuras 6 y 7 se detallan los bloques de codificación y shortening y de puncturing utilizados.

Para la etapa de puncturing se utiliza una memoria de tipo RAM de doble puerto haciendo que la velocidad de lectura sea diferente al de la escritura modificándola con un bloque de *"Down Sample"* seguido de un *"Up Sample"*.

Después de tener el bloque codificado, se convierten los datos de paralelo a serial para ser procesados por un codificador convolucional nativo con tasa de codificación $\frac{1}{2}$ y una longitud de restricción de 7 utilizando los siguientes polinomios (Ec.4 y 5):

$$G_1 = 171_{oct} \quad \text{para } X \quad (3)$$

$$G_2 = 133_{oct} \quad \text{para } Y \quad (4)$$

Se utiliza puncturing variable para ajustar la capacidad de corrección de errores del codificador, para lo cual se elimina una cierta cantidad de bits tal como está especificado en la tabla 3, donde además se muestra la distancia libre del código d_{free} que representa por medio de la Ec. (5) la capacidad de corrección de errores del codificador convolucional [4].

$$t = \frac{d_{free} - 1}{2} \quad (5)$$

La Figura 8 muestra la implementación del codificador convolucional en "System Generator", donde el primer bloque expandido en la Figura 9 contiene el diseño del

Tasa	**1/2**	**2/3**	**3/4**	**5/6**
$\mathbf{d}_{free}$	10	6	5	4
X	1	10	101	10101
Y	1	11	110	11010
XY	X_1Y_1	$X_1Y_1Y_2$	$X_1Y_1Y_2X_3$	$X_1Y_1Y_2X_3Y_4X_5$

Tabla 3: Vectores de Puncturing para el Codificador Convolucional

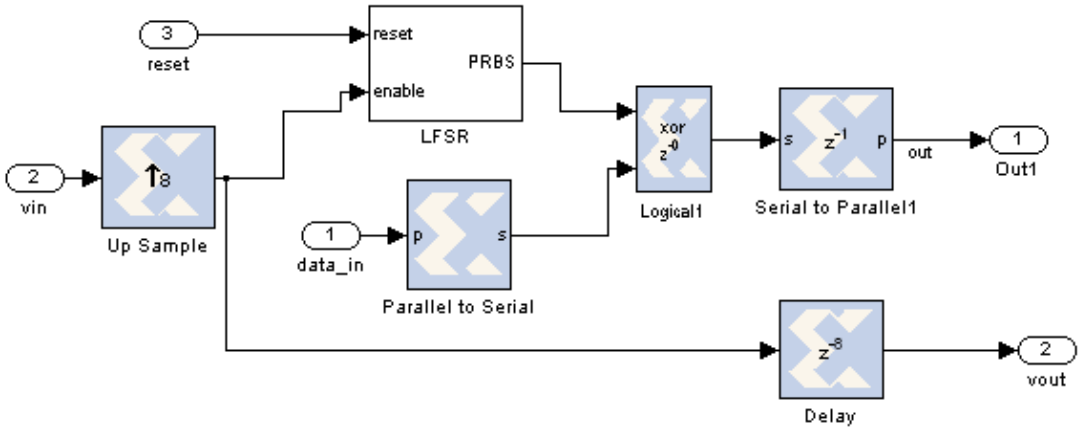

Figura 4: Implementación del Randomizer

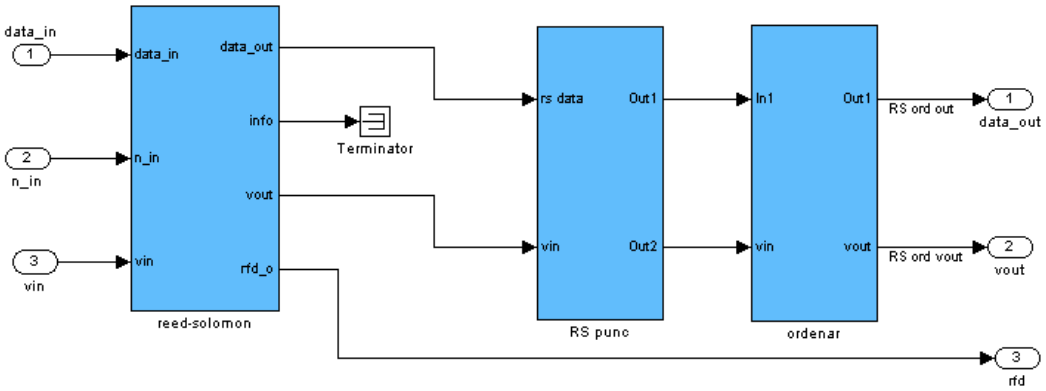

Figura 5: Codificación Reed-Solomon.

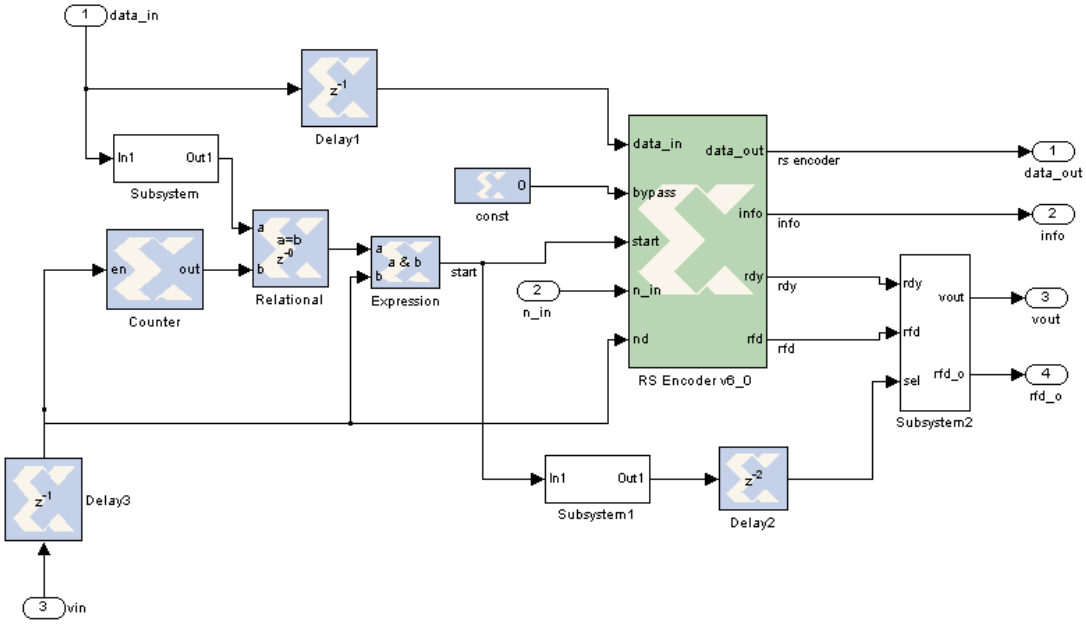

Figura 6: Codificación y Shortening.

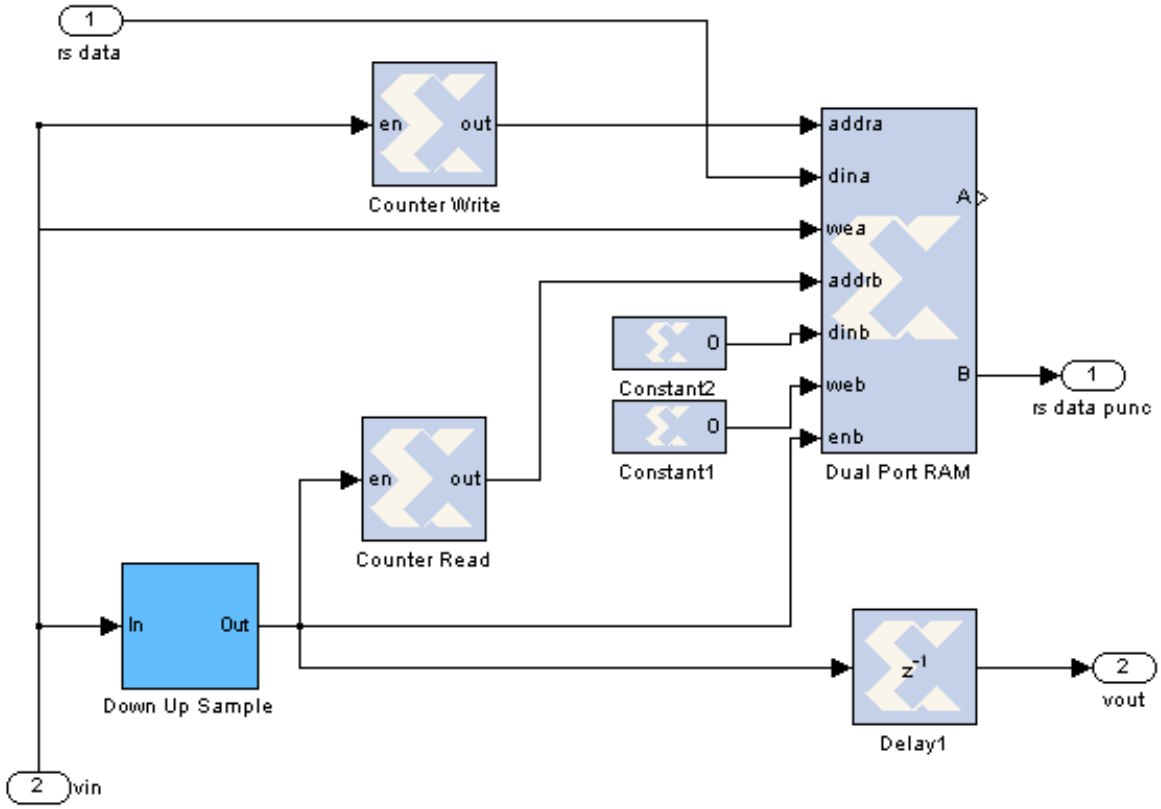

Figura 7: Puncturing del Codificador de Reed-Solomon.

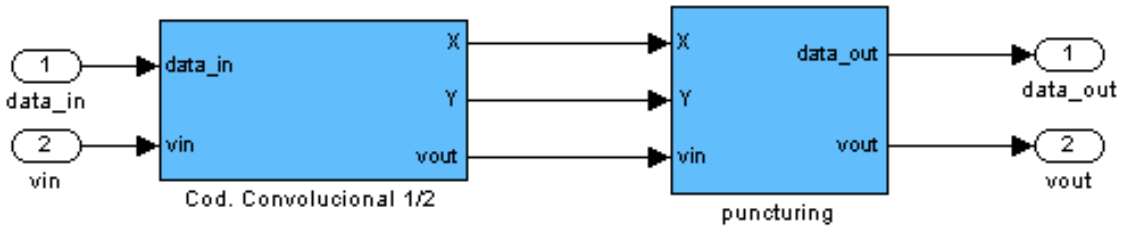

Figura 8: Codificación Convolucional.

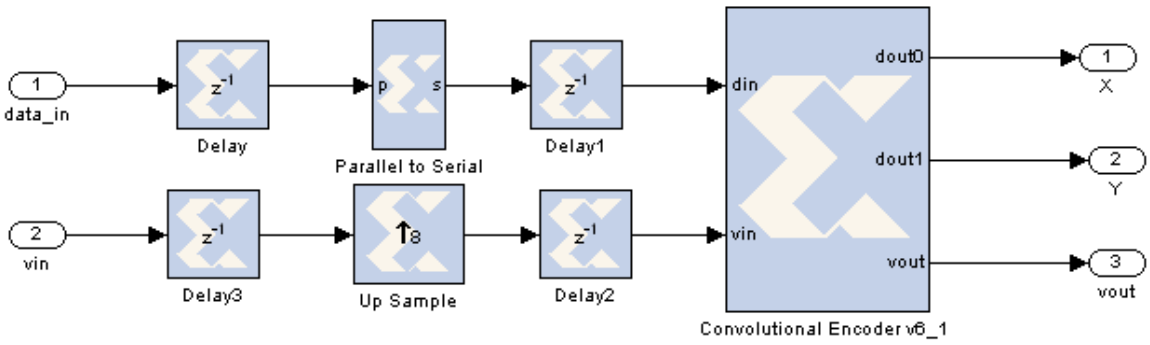

Figura 9: Codificador convolucional nativo.

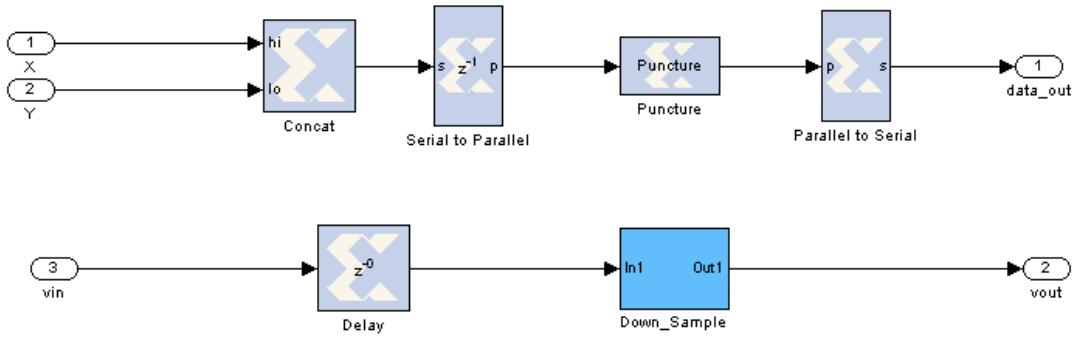

Figura 10: Puncturing usado en el codificador convolucional.

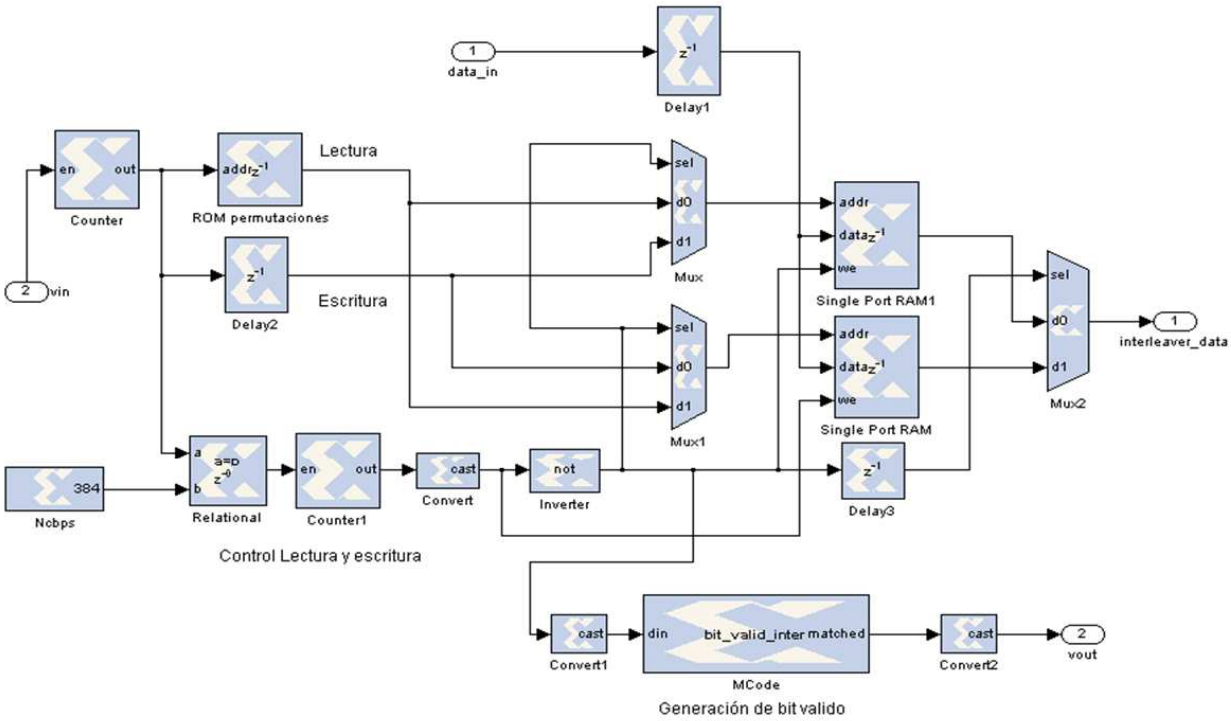

Figura 11: Diseño del Interleaver

codificador nativo, y el segundo bloque la etapa de puncturing que se muestra en la Figura 10 y que depende del perfil de codificación escogido.

Luego de pasar los datos por la etapa de FEC, estos pasan por una etapa de entrelazado (interleaver), el cual se encarga de aleatorizar la posición en que se localicen los potenciales errores. Este proceso se realiza por medio de 2 permutaciones generadas en Matlab y que se muestran a continuación:

```
\%interleaver
\%Ncbps=192(BPSK),384(QPSK),768(16-QAM),1152(64-QAM)
\%Ncpc=1(BPSK),2(QPSK),4(16-QAM),6(64-QAM)
k=0 : Ncbps -1;
mk=(Ncbps/12)*mod(k,12)+floor(k/12);
s=ceil(Ncpc/2);
jk=s*floor(mk/s)+mod(s,mk+Ncbps-floor(12*mk/Ncbps));
[s1,int_idx]=sort(jk);
s=ceil(Ncpc/2);
\%deinterleaver:
[s2,dint_idx]=sort(int_dx);
```

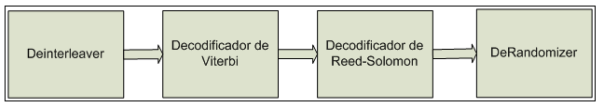

Figura 12: Diagrama de bloques del decodificador del canal.

Para el interleaver se sigue el esquema de la Figura 11, donde se utilizan 2 memorias de tipo RAM, las cuales se alternan para realizar el proceso de escritura y lectura de los datos, donde este último paso se realiza leyendo la posición de los datos de una memoria de tipo ROM que contiene el vector generado en Matlab “int_idx” luego de realizar las 2 permutaciones descritas anteriormente.

Diseño del decodificador

La decodificación se encarga de recuperar la información transmitida dado un bloque de símbolos que contiene información redundante más los errores causados

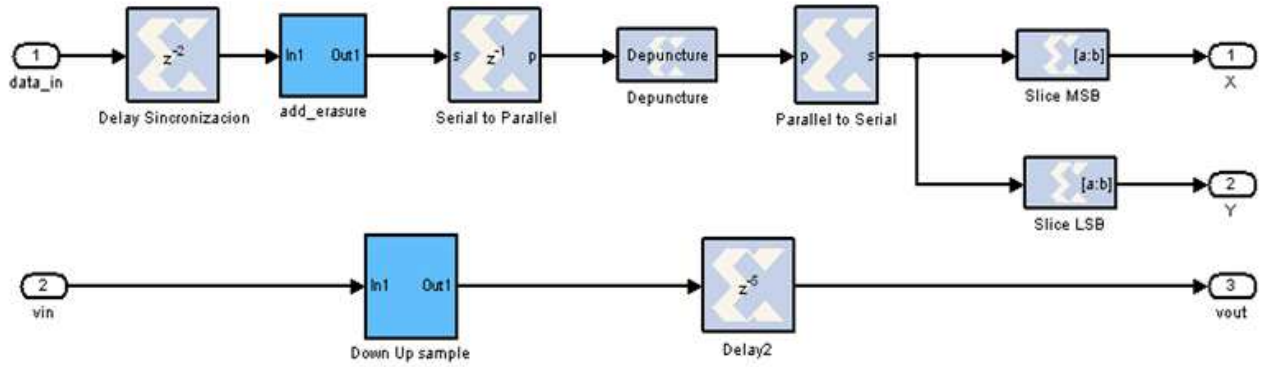

Figura 13: Depuncturing usado en el decodificador de Viterbi.

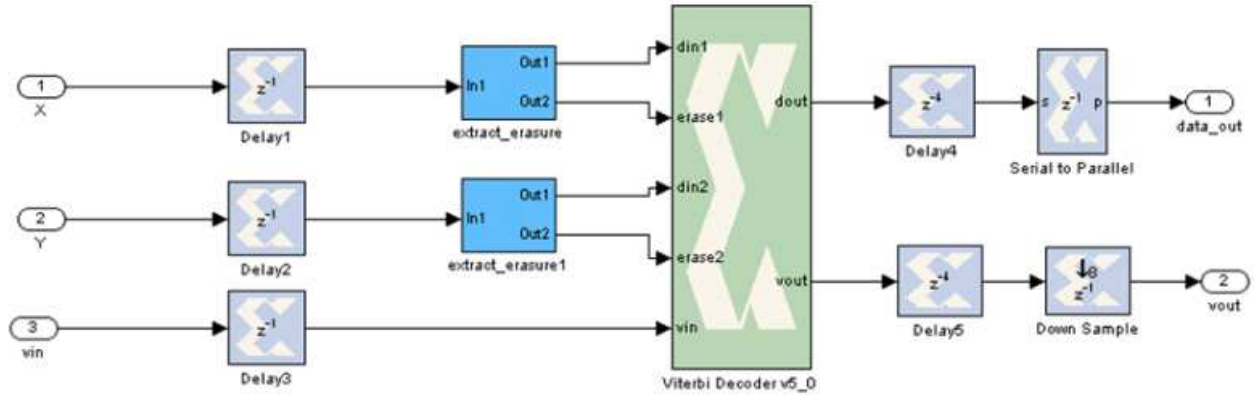

Figura 14: Decodificación de Viterbi.

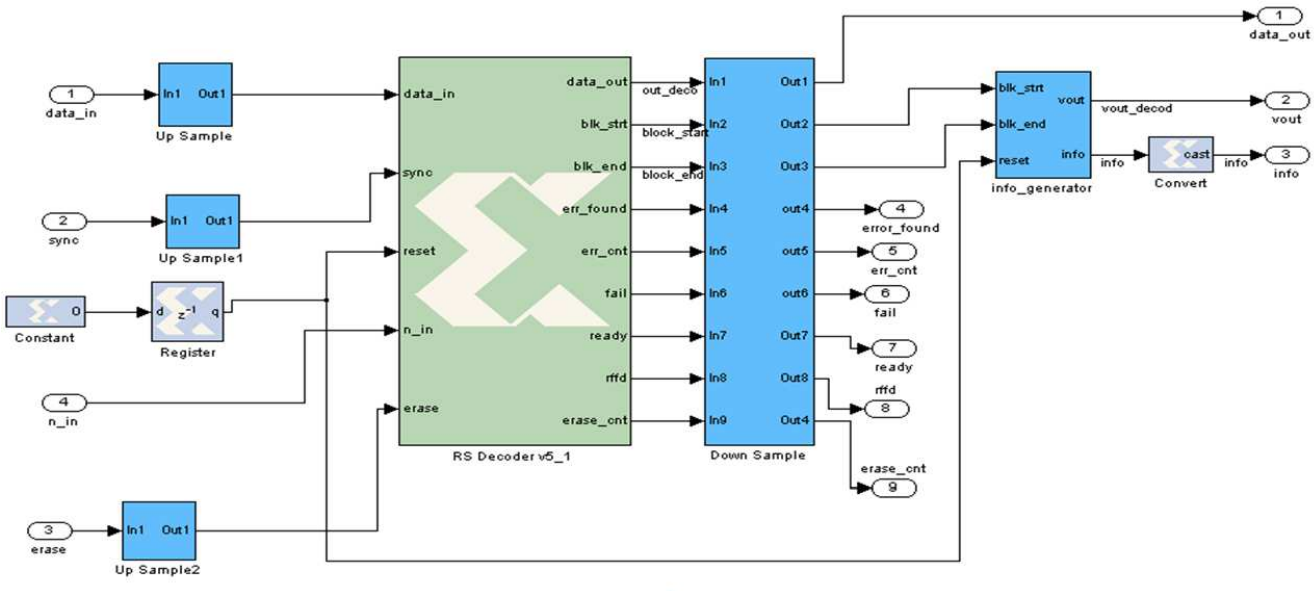

Figura 15: Depuncturing del decodificador de Reed-Solomon.

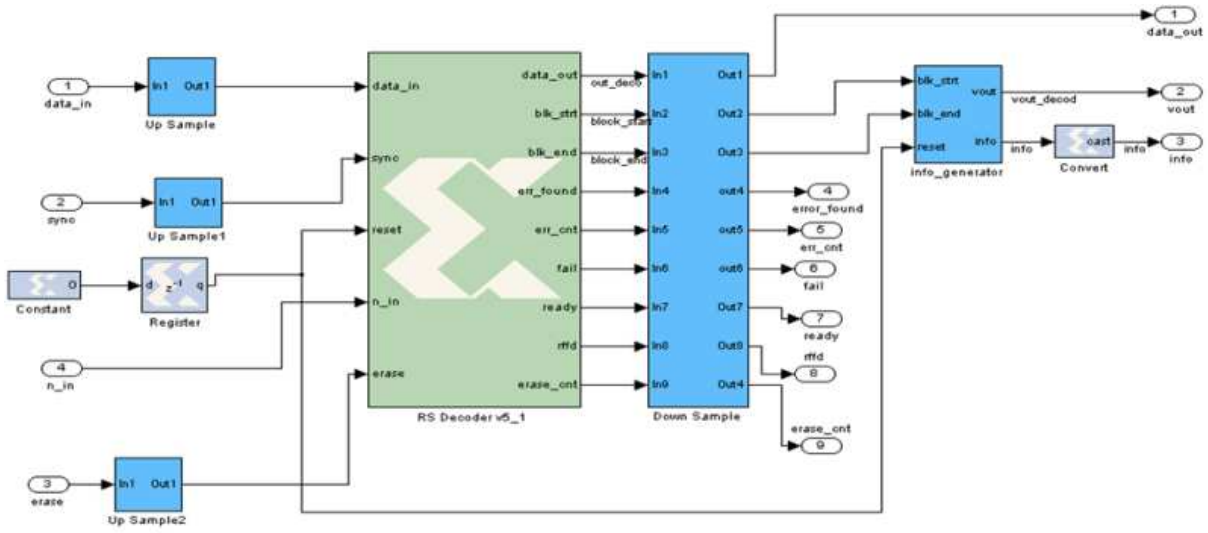

Figura 16: Decodificación de Reed-Solomon.

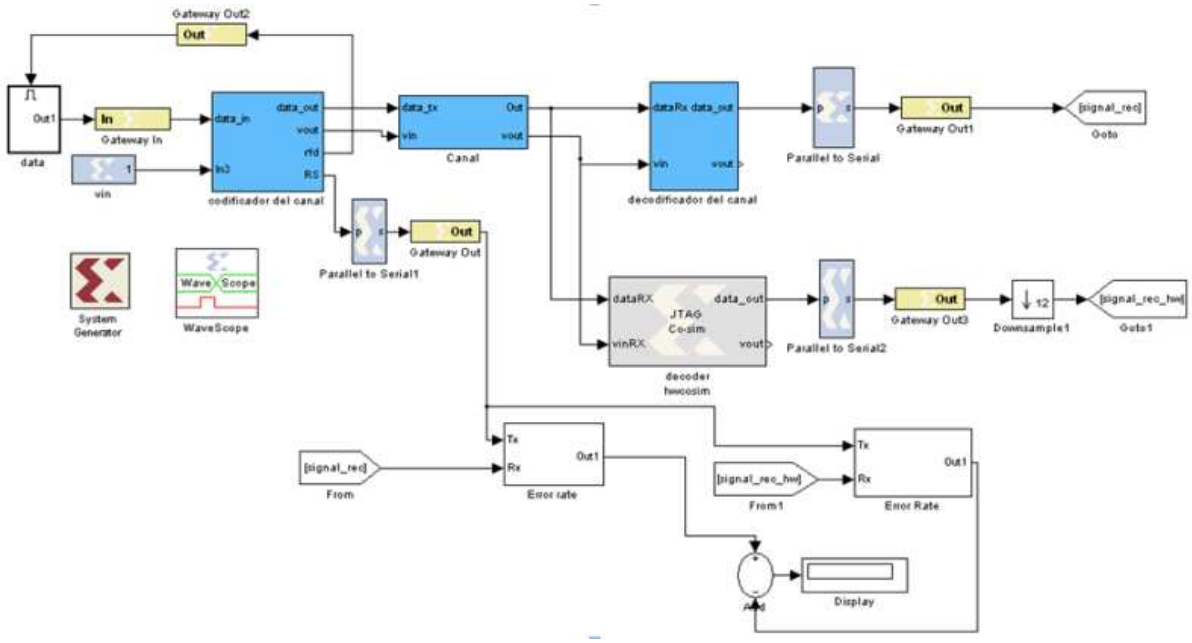

Figura 17: Verificación en FPGA del sistema diseñado.

por el canal. Se asume una demodulación de los datos usando decisión soft con 8 niveles de cuantización (3 bits). Se sigue el diagrama de bloques de la Figura 12.

Para el desentrelazado (deinterleaver) se sigue el mismo diseño descrito en la Figura 11, donde el único cambio consiste en cambiar el vector de inicialización de la memoria ROM por "dint_idx". Para el diseño del decodificador de Viterbi se sigue el proceso inverso al usado en el codificador convolucional. En las Figuras 13 y 14 se ilustra este proceso. Se utiliza el mismo vector de puncturing usado en la codificación, añadiendo un símbolo que indique una mayor probabilidad de haber recibido un cero, y etiquetando ese símbolo nulo con un bit adicional para que pueda ser interpretado por el bloque del decodificador de Viterbi.

Los datos obtenidos, en forma serial (un bit) se convierten a un formato en paralelo (8 bits) para que puedan ser procesados por el decodificador de Reed-Solomon. Para la decodificación de Reed-Solomon se sigue el proceso inverso, donde para reordenar los símbolos sólo se cambiél vector de inicialización de la ROM. Para la etapa de depuncturing se tiene que agregar bytes ceros para reemplazar los símbolos eliminados en el codificador y agregar además un bit de bandera en el instante de tiempo en que son agregados ("erase" en las Figuras 15 y 16).

Implementación en FPGA.

Tanto el codificador como el decodificador fueron implementados en una spartan 3E xc3s500e-4fg320 y se analizó su funcionamiento usando "*hardware cosimulation*". En la Figura 17 se presenta la verificación del

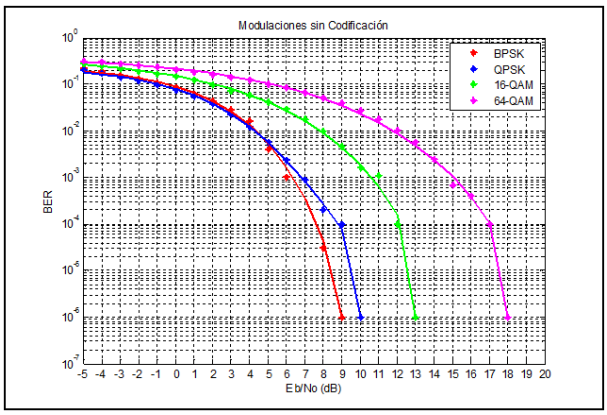

Figura 18: Comparación de las modulaciones usadas en Wimax sin usar codificación.

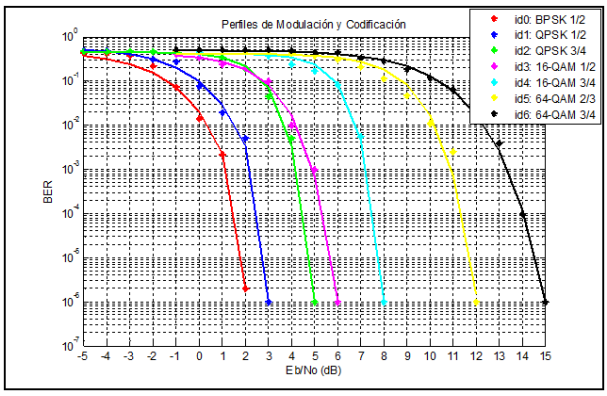

Figura 19: Resultados de la Simulación de los Perfiles de Codificación y Modulación usando el sistema diseñado.

diseño usando Simulink y el FPGA que esta expresado en el bloque "JTAG Co-Sim", el cual se comunica con el computador por medio de una interfaz USB. Dado que el diseño está hecho en punto fijo, los resultados en hardware deben ser los mismos obtenidos al realizar la simulación. Se realizó la implementación en forma separada para cada perfil de codificación.

Se presenta en la tabla 4 los recursos utilizados en hardware considerando el perfil de codificación 2 (QPSK $\frac{3}{4}$).

Etapa	Slice	Slice FF	LUT	BRAM
Randomizer	34	61	34	0
Reed-Solomon	214	285	285	4
Convolucional	66	115	74	0
Interleaver	26	36	51	3
Deinterleaver	22	34	43	3
Viterbi	1667	1515	2379	4
Reed-Solomon	1627	1978	2907	9

Tabla 4: Recursos de la FPGA usando perfil de codificación QPSK $\frac{3}{4}$

Análisis de Resultados

Variación del BER según el SNR.

La mejor forma de verificar la fiabilidad del sistema diseñado es midiendo la probabilidad de error de bit

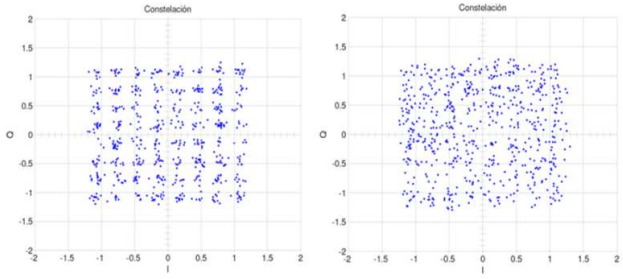

Figura 20: Comparación de constelaciones a la salida del canal usando modulación 64-QAM. A la izquierda grafica sin codificación y a la derecha grafica usando codificación 2/3

(BER) para diferentes valores de relación de señal a ruido (SNR). Utilizando codificación de canal se puede detectar y corregir errores causados por el canal de comunicaciones, por lo que el BER tiene que mejorar comparado a un sistema que no utilice codificación.

Antes de analizar el sistema con codificación, se muestra en la figura 18 las diferentes modulaciones usadas en el estándar de Wimax y las simulaciones se realizaron con detección *hard*. Luego en la figura 19 se muestran los resultados del sistema diseñado donde se utilizó la misma trama de datos que el caso anterior. Las curvas realizadas son de BER ("Bit Error Rate") vs EbNo ("energy per bit to noise").

Se puede observar comparando las dos gráficas que con el sistema diseñado la curva del BER para cada modulación tiende a desplazarse a la izquierda. Para el caso de la modulación BPSK hay una mejora de 5 dB cuando se tiene un BER de 10-3. Esto se conoce como ganancia de codificación, y conforme se acerca el BER a valores 10-6 esta ganancia de codificación crece.

Comparación de Constelaciones a la salida del canal.

Para analizar los beneficios del codificador del canal, se muestra en la Figura 20 un gráfico de la constelación de una señal 64-QAM a la salida de un canal AWGN con un SNR dado para tener un BER de 10-3, donde el gráfico de la izquierda representa el caso sin codificación y el de la derecha usa una codificación con tasa 2/3.

Se observa en la gráfica que con el uso de la codificación de canal se necesita un SNR menor para tener el mismo desempeño al que se tiene en el caso de no usar codificación.

Verificación del estándar.

Para verificar que el diseño propuesto cumple las especificaciones del estándar IEEE 802.16-2004, se utilizaron los vectores de prueba [1] que exige el mismo para analizar los datos a la salida de cada bloque. Estos vectores se usan como entrada de datos al sistema por medio de los bloques de Simulink y se espera que a la salida de cada uno se obtengan los mismos valores que especifica el estándar, tal como se muestra en el diagrama de tiempo de la Fig. 21.

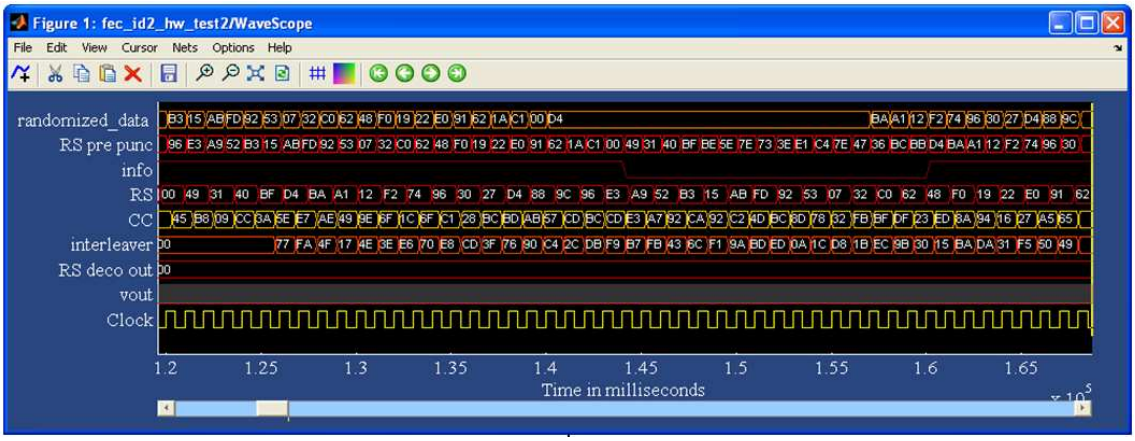

Figura 21: Diagrama de Tiempo del sistema diseñado

Conclusiones

Con el desarrollo de este trabajo se ha podido comprobar las ventajas de utilizar un ambiente de desarrollo de Hardware de alto nivel, con la ayuda del software System Generator de Xilinx. En base a las simulaciones realizadas usando Matlab y Simulink se concluye que el diseño del codificador cumple con las características especificadas en el estándar 802.16, al obtener la secuencia de bits deseados a la salida en base al vector de prueba utilizado como entrada. Con las graficas BER vs SNR obtenidas, se demuestra la capacidad de detección y corrección de errores del sistema. Así mismo al visualizar las constelaciones de las señales a la salida del canal, se puede tener una idea de que tan eficaz es el codificador usado.

Se recomienda para poder usar este diseño en conjunto con otras partes de la transmisión y recepción de Wimax, el uso de una FPGA de mayores prestaciones.

Referencias

[1] Instituto de Ingenieros Eléctricos y Electrónicos. 2004. “Estándar IEEE-802.16-2004”, Enlace: http://standards.ieee.org. 430-445.

[2] Lopez, F. 2005. “Diseño de transmisor y receptor para redes inalámbricas WMAN”. *Tesis de Licenciatura: Málaga*.

[3] Serra, M. 2005. “Prototipado rápido de la capa física de OFDM: Hiperlan2”. *Tesis de Doctorado Universidad Autónoma de Barcelona*.

[4] Sklar, B. 2001. “Digital Communications Fundamentals and Applications”. Prentice Hall. 408-412