UNIVERSIDAD CENTRAL "MARTA ABREU" DE LAS VILLAS

FACULTAD DE INGENIERÍA ELÉCTRICA

DEPARTAMENTO DE ELECTRÓNICA Y TELECOMUNICACIONES

Codificador de Televisión Digital Terrestre de la norma DTMB con FPGA

Tesis presentada en opción al Título Académico de Máster en Telemática

Maestría en Telemática

Autor: Ing. Rafael Orestes Arias Zamora

Tutor: Dr. C. Juan Pablo Barrios Rodríguez

Santa Clara, Cuba, 2017

RESUMEN

El presente trabajo contribuye al desarrollo de la televisión digital terrestre en el país y tiene como objetivo desarrollar un codificador para la norma de televisión digital DTMB ***Digital Terrestrial Multimedia Broadcast*** que pueda configurarse en un FPGA (del inglés, ***Field Programmable Gate Arrays***), lo cual permita expandir el servicio de televisión digital en el país. En el trabajo se describen las características generales de la norma DTMB y el proceso de codificación de la misma. Se hace referencia a diferentes estrategias de diseño digital basadas en FPGA para desarrollar sistemas de procesamiento digital de señales y se aborda el diseño digital de las etapas aleatorizador y codificación de canal a través de herramientas ***Xilinx System Generator*** y ***Matlab-Simulink***. Se realiza la simulación del diseño mediante Simulink y se comprueba que los resultados obtenidos cumplen con el funcionamiento y requerimientos establecidos por la norma DTMB.

Palabras Clave: TDT, FPGA, DTMB, Xilinx System Generator

ÍNDICE

INTRODUCCIÓN

La Televisión Digital Terrestre TDT, es el resultado de la revolución tecnológica que se ha producido en los últimos años, debido a grandes innovaciones en los campos de procesamiento digital de señales e imágenes y la microelectrónica. Estos avances sumados a las tecnologías de transmisión de telecomunicaciones hacen posible la transmisión de datos digitales con un ancho de banda soportado por las diversas redes de transmisión con la suficiente robustez ante errores y pérdidas de información, además de un procesamiento de datos en tiempo real.

La TDT es la transmisión de video y su sonido asociado mediante una señal digital a través de una red de repetidores terrestres. La TDT aporta diversas ventajas. Entre ellas cabe destacar en primer lugar, la posibilidad de comprimir la señal, lo que implica que esta requiere un ancho de banda menor para su transmisión. Como resultado, se puede efectuar un uso más eficiente del espectro radioeléctrico. Tras proceder a su multiplexación, se pueden emitir más canales, que en el sistema digital pasan a denominarse programas digitales en el espacio antes empleado por uno, denominado ahora canal múltiple digital o multiplex. Puesto que en el ancho de banda empleado por un canal analógico ahora se pueden transmitir varios programas digitales, la emisión digital aporta un importante ahorro energético por canal. Ello implica una reducción de costos para los radiodifusores. Tiene la ventaja aportada por la codificación digital de una mejora de la calidad de la imagen y el sonido en el momento de la recepción, puesto que ambos están codificados de manera digital, cuando se produce alguna distorsión en la señal, aquella puede ser corregida por el receptor[1].

La transmisión de TDT se realiza siguiendo los parámetros técnicos establecidos por diferentes estándares tecnológicos. Existen varios y su uso por parte de los estados responde a su capacidad para crear estándares, a su ubicación geográfica y a su pertenencia a la esfera de influencia de los estados creadores de estándares. Actualmente existen básicamente cuatro estándares para la radiodifusión de televisión digital terrestre. ATSC utilizado en Norteamérica, DVB-T utilizado en Europa, ISDB-T desarrollado por Japón, la variación del estándar japonés SBTVD-T desarrollado por Brasil y el estándar DTMB desarrollado por China.

La incursión de la televisión digital terrestre en los países no comprende solamente consideraciones de carácter técnico, sino también legales, sociales y políticas, en donde se debe definir la tecnología a usar, cómo se realizará el proceso de transición, el tiempo definido para la migración hacia lo que se conoce como el apagón analógico, las políticas de importaciones de los nuevos receptores, e incluso políticas sociales para facilitar el acceso a esta tecnología a los grupos de menores recursos económicos.

Luego de estudios y la realización de pruebas técnicas, en septiembre del 2013 fue publicado en gaceta oficial, el acuerdo 7455 del Consejo de Ministro mediante el que se

aprueba el estándar internacional de trasmisión de televisión digital terrestre DTMB y se autoriza su introducción y despliegue en Cuba según sus especificaciones técnicas, con las adecuaciones y mejoras tecnológicas necesarias para el país y las posteriores evoluciones tecnológicas que de esta se deriven[2].

El programa para la implementación de la TDT en Cuba cuenta con tres fases; etapa de simultaneidad, despliegue del primer servicio de televisión digital y por ultimo despliegue del segundo servicio. La ejecución total de este programa se previó en 15 años. Durante la necesaria etapa de simultaneidad estimada en 10 años aumenta el consumo energético, ya que es necesario mantener los dos flujos analógico y digital. Posteriormente, en la etapa de completamiento se agregarán equipos en nuevas localizaciones para lograr la mayor cobertura posible. Con esta información resulta obvio que mientras más rápido se pueda realizar el apagón analógico, mayor eficiencia se alcanzará en el programa. También se debe considerar que el apagón analógico implicaría una disminución considerable de los costos de mantenimiento y operación, que en paralelo y debido a la obsolescencia tecnológica de la red analógica continuarían en ascenso[2].

Como resultado de los estudios realizados se han identificado debilidades y vulnerabilidades que pueden afectar la eficacia y eficiencia del proceso de implementación de la TDT. Principalmente se han presentado dificultades con el sistema de recepción de la señal, dígase antena, bajante o cable, conector, insuficiente oferta e incentivos que estimulen la migración, o sea servicios de valor agregado, precios asequibles, facilidades para la adquisición de los nuevos receptores, nuevos contenidos, etc. Además los recursos financieros para acometer las inversiones que demanda esta transición son limitados[2].

Este proceso de avance de la TDT está determinado por la capacidad del país para invertir en el equipamiento necesario para esta tecnología, parte de esta inversión son los codificadores para la norma. Una ventaja en el proceso de despliegue de la televisión digital sería la posibilidad de desarrollar moduladores para la norma en el país lo cual incluye el proceso de codificación. Esto posibilitaría aumentar la cantidad de transmisores de manera que se pueda llevar la señal digital a todo el país, lograr una independencia tecnológica; así como una apropiación de esta tecnología. Además se ha podido comprobar que diferentes centros de investigación en diferentes países están realizando investigaciones que pretenden desarrollar moduladores para la norma de televisión digital que tienen definida utilizar.

En Cuba el Instituto de Investigación y Desarrollo de Telecomunicaciones LACETEL se encuentra enfrascado desde el 2013 en el desarrollo de un modulador propio de transmisión digital terrestre multimedia (DTMB), a partir de la norma china[3].

Debido al número relativamente pequeño de moduladores en las cadenas de transmisión de la televisión, aun en los países más extensos territorialmente, no es factible económicamente la producción de ASIC para realizar la modulación. Por otra parte, la velocidad requerida para el procesamiento de la información exige, en muchos casos, el empleo de estructuras de hardware dedicadas. Estas dos razones determinan que los moduladores para TDT y dentro de los mismos la etapa de codificación, sean desarrollados actualmente sobre tecnología FPGA [4].

Por su versatilidad y flexibilidad los arreglos de compuertas programables por campo FPGA constituyen una solución viable para implementar sistemas digitales de procesamiento de señales como es el caso de un codificador de televisión digital. Hoy en día existen varias familias de FPGA de diferentes fabricantes las cuales tienen características diferentes, dependiendo de las mismas están son en mayor o menor medida atractivas para los diferentes diseños de procesamiento digital de señales.

Existen diferentes programas como Simulink, de Matlab que tiene ejemplos de diseños digitales de codificación basados en FPGA para otras normas de TDT como DVB, además tiene disponibles bloques de codificación que se pueden configurar según la norma DTMB pero los diseños utilizando dichos bloques no pueden ser implementados en hardware. Por otra parte existen varias publicaciones y trabajos realizados por diferentes institutos como el Instituto de Investigación y Desarrollo de Telecomunicaciones LACETEL los cuales son referenciados [5], [6], [7]; que abordan cómo funciona la norma DTMB y la etapa de codificación de la misma pero no describen específicamente como realizar el diseño digital de un sistema que permita lograr el funcionamiento correcto de la codificación.

De aquí que el problema de investigación de este trabajo sea, la necesidad de desarrollar codificadores de canal de la norma DTMB que pueda implementarse en hardware para expandir el servicio de televisión digital en el país.

El objeto de investigación es la norma de televisión digital DTMB y el campo de acción es la codificación BCH (siglas en inglés de ***Bose-Chadhui Hocquenghen***) y LDPC (siglas en inglés ***Low Density Parity Check***) de la norma de DTMB.

El objetivo general de esta investigación es: Desarrollar un codificador para la norma de televisión digital terrestre DTMB sobre un hardware configurable FPGA.

Los objetivos específicos que se plantean son.

- Describir el funcionamiento de las etapas de codificación de la norma DTMB.
- Seleccionar una estrategia de diseño de sistemas digitales complejos con FPGA.
- Realizar el diseño del aleatorizador y codificación de canal de la norma DTMB para ser implementado en hardware configurable.
- Comprobar, mediante simulación, que los resultados del diseño realizado cumple con los requerimientos que establece la norma.

Para ello las tareas científicas que se plantean son

- Descripción del proceso de codificación de la norma DTMB.
- Selección de una estrategia de diseño de sistemas digitales basados en FPGA.
- Diseño de aleatorizador y codificación de canal de la norma DTMB para ser implementado en hardware.
- Comprobación, mediante simulación del desempeño del diseño realizado de acuerdo a la norma DTMB.

En el desarrollo de este trabajo se describe el proceso de codificación de canal de la norma de televisión digital terrestre implantada en Cuba, así como una estrategia de diseño digital

con circuitos VLSI (del inglés: ***Very Large Scale Integrated***) para el procesamiento de señales.

El trabajo está estructurado en tres capítulos. En el primero se caracteriza el estado de transición de la televisión analógica a la digital en Cuba. Se realiza una descripción de las características generales de la norma DTMB y se describe el funcionamiento de las diferentes etapas del proceso de codificación de la misma.

En él segundo capítulo se abordan las diferentes estrategias de diseño de sistemas digitales con FPGA de Xilinx. Se describe la herramienta de software Xilinx System Generator y se realiza el diseño de las diferentes etapas de codificación de canal de la norma DTMB.

En el tercer capítulo se realiza una simulación funcional del diseño realizado mediante bloques propios de Xilinx System Generator que pueden ser implementados en hardware donde se evalúan los resultados obtenidos teniendo en cuenta el funcionamiento de la norma y la comparación con los resultados que se obtienen mediante el diseño en Simulink.

Por último se realizan las conclusiones del trabajo teniendo en cuenta los resultados obtenidos y se plantean algunas recomendaciones.

Como resultado de la presente tesis se espera obtener un diseño de las etapas aleatorizador y codificación de canal de la norma DTMB que pueda ser implementado en hardware.

CAPÍTULO 1. Estándar de televisión digital DTMB empleado en Cuba.

En el presente capitulo se caracteriza el estado actual de transición de la televisión analógica a la digital en Cuba. Se realiza una descripción de las principales características de la norma de televisión digital DTMB y se detalla el funcionamiento de las etapas aleatorizador y codificación de canal del proceso de modulación de la misma.

1.1 Criterios técnicos que se tiene en cuenta para la adopción del estándar de TDT.

Es posible realizar un análisis comparativo de los estándares de Televisión Digital Terrestre: ATSC, DVB-T, ISDB-T y DTMB, en base a la recomendación UIT-R BT.2035-2 [8] de la Unión Internacional de Telecomunicaciones, división de Radiocomunicaciones para la evaluación de sistemas de radiodifusión de televisión digital terrestre.

La recomendación ITU-R BT.2035-2 se centra en las pruebas y ensayos de radiodifusión de televisión digital terrenal que permiten evaluar la calidad de funcionamiento del sistema o sistemas disponibles con diversas configuraciones de transmisión y condiciones de recepción. Es por ello que constituye la base para caracterizar los diferentes estándares para la radiodifusión de televisión digital terrestre y realizar un análisis comparativo entre ellos, en cuanto a su desempeño técnico.

Los planes de pruebas de laboratorio descritos en la recomendación UIT-R BT.2035-2 [8] tienen como objetivo implementar una metodología que permita, independientemente de las características del estándar de transmisión, caracterizar el desempeño técnico de los mismos. Los procedimientos descritos pretenden verificar la calidad de funcionamiento de los moduladores y receptores de TDT. Estas pruebas incluyen mediciones de la calidad de funcionamiento del sistema de transmisión y recepción. Entre los parámetros que se tienen en cuenta están ruido aleatorio, margen dinámico de la señal RF de entrada e interferencia multitrayecto[8].

La medición de la degradación por ruido aleatorio tiene como objetivo determinar la robustez de los receptores de TDT a las degradaciones producidas por ruido aleatorio. Para esto la señal de TDT se ajusta a cuatro niveles de potencia, estos son; muy intenso -24dBm, intenso -28dBm, moderado -53dBm y débil -68dBm, para tres frecuencias diferentes: 473 MHz (canal 14), 587 MHz (canal 33) y 695 MHz (canal 51)[8].

El nivel de ruido se aumenta hasta alcanzar el umbral de visibilidad TOV (del inglés, ***Threshold Of Visibility***). En ese punto se registra entonces el valor de la relación señal a ruido C/N. En el ámbito de las pruebas de laboratorio, se considera que se alcanza el TOV cuando un observador entrenado puede detectar algún tipo de perturbación en la imagen tras un minuto de observación.

Para determinar el margen dinámico de la señal de RF de entrada se debe comprobar la capacidad de los receptores para recibir señales, desde muy intensas a muy débiles. El nivel máximo y mínimo de señal de RF se determinará aumentando y disminuyendo respectivamente el nivel de potencia de la señal de RF a la entrada del receptor hasta que se alcance el nivel de TOV. Se registra en cada caso el valor del nivel de la señal medido con los instrumentos[8].

Para la prueba de interferencia provocada por trayectos múltiples estáticos se mide la calidad de funcionamiento del receptor de TDT para dos combinaciones de trayectos múltiples representativos de varios entornos de recepción. El objetivo de la prueba es medir la robustez de las Cajas Decodificadoras en presencia de reflexiones de la señal al propagarse en el medio. Todas las pruebas de trayectos múltiples se realizan con un nivel de la señal de RF ajustado a un valor moderado (−53 dBm). Nótese que por coherencia con los valores de razón señal a ruido C/N, el nivel de potencia de la señal será el resultado de la combinación de la señal principal y las de eco[8].

La presencia de objetos reflectores y dispersores en el entorno produce múltiples versiones de la señal transmitida. Estas llegan a la antena receptora desplazadas, una respecto a la otra, en el tiempo y orientación espacial. Los tiempos de propagación de las ondas son distintos y dependen de las características del entorno fundamentalmente. El modelado de cada uno de los entornos debe realizarse teniendo en cuenta múltiples consideraciones estadísticas[9].

Una distribución de Rayleigh, ver [9]; caracteriza un entorno en el cual la línea de visibilidad directa entre el transmisor y el receptor se ve interrumpida por obstáculos. Por el contrario, una distribución de Rician, ver [9]; caracteriza una trayectoria con visibilidad directa. El canal principal tiene un nivel de energía superior al resto de las señales multitrayecto captadas por el receptor[9].

En la recomendación de referencia de la UIT se sugiere emplear en los set de medición el esquema mostrado en la figura 1.1 que se muestra a continuación.

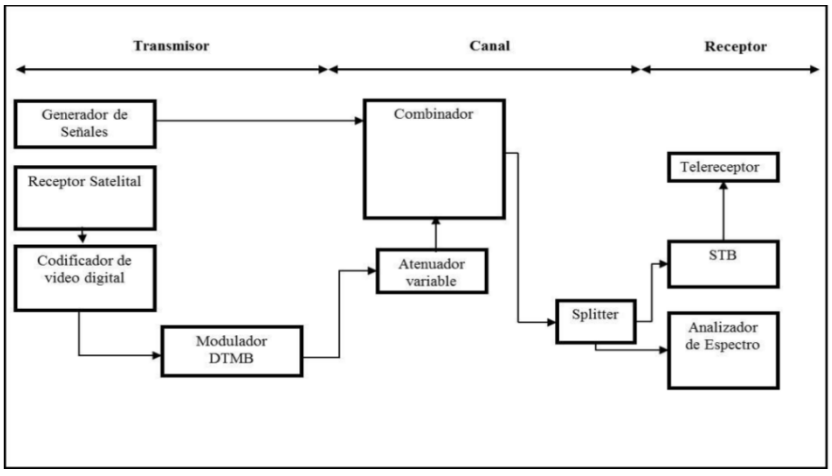

Figura 1.1: Set de mediciones[8].

En la realización práctica de las pruebas realizadas para la norma DTMB en el Laboratorio Nacional de Televisión Digital, en LACETEL se utiliza el esquema mostrado en la figura

anterior y para reducir el número de mediciones se seleccionaron diferentes modos típicos dentro las configuraciones establecidas en la norma GB20600-2006[10].

La tabla 1.1 mostrada a continuación refleja los principales parámetros de los modos de transmisión utilizados.

Tabla 1.1 Principales parámetros de los modos seleccionados[11].

Modo	Número de Portadoras	FEC Razón de Código	Modulación	Longitud de la Cabecera	Entrelazado	Razón de Bits (Mbps)
1	3780	0.4	16QAM	PN945	720	7.220
2	1	0.8	4QAM	PN595	720	7.797
3	3780	0.6	16QAM	PN945	720	10.829
4	1	0.8	16QAM	PN595	720	15.593
5	3780	0.8	16QAM	PN420	720	16.244
6	3780	0.6	64QAM	PN420	720	18.274
7	1	0.8	32QAM	PN595	720	19.492

1.1.1 Análisis comparativo de los estándares según UIT-R BT.2035-2

Los datos de comparación con los estándares ATSC, DVB-T e ISDB-T son los datos oficiales recogidos en el informe anexo a la recomendación UIT-R BT.2035-2, mientras los datos para DTMB con canalización de 6 MHz fueron obtenidos a partir de mediciones realizadas en LACETEL.

Como parámetro común para garantizar la igualdad de condiciones dentro de la diversidad de esquemas de modulación de cada norma se emplea la velocidad de bits. Seleccionando dentro de las configuraciones definidas para cada norma la más robusta que admite determinada velocidad de bits.[10]

En cuanto a ruido aleatorio según [10] y como puede observarse en la figura 1.2 la cual refleja los niveles de relación señal a ruido C/N al alcanzar el TOV. ATSC y DTMB se comportan de manera similar en cuanto a ruido aleatorio mientras que DTMB es mejor en más de 3dB, respecto a DVB-T e ISDB-T.

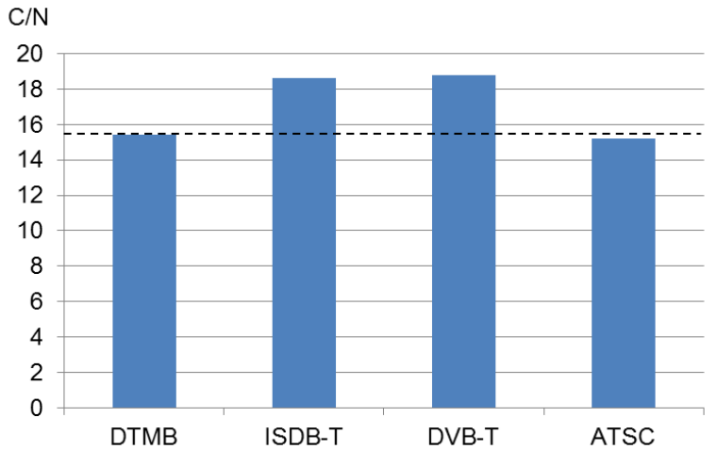

Figura 1.2: Análisis comparativo en cuanto a la inmunidad al ruido aleatorio[10].

En cuanto al parámetro de rango dinámico de la señal de RF en promedio, como puede observarse en la figura 1.3; los receptores DTMB son capaces de recibir señal y demodularla correctamente con 7.9 dB menos de señal que los receptores ATSC, el estándar más cercano. A pesar de que existe una pequeña diferencia de 0.2 dB entre DTMB y ATSC en cuanto al ruido aleatorio, el estándar DTMB logra mayor cobertura para un mismo nivel de potencia de transmisión[10].

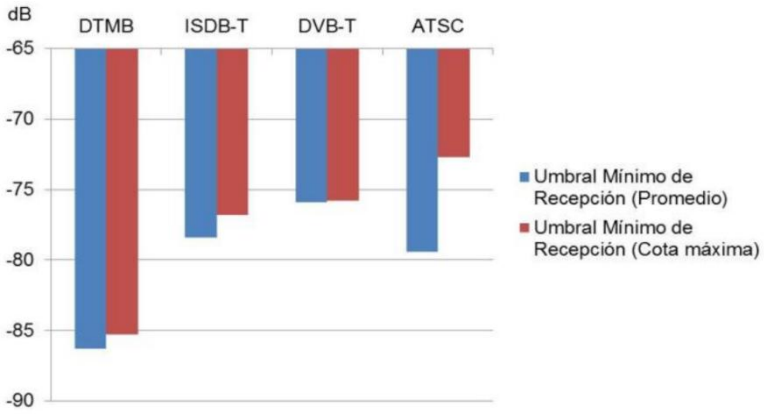

Figura 1.3: Análisis comparativo en cuanto al rango dinámico de señal RF[10].

Los resultados comparativos teniendo en cuenta la interferencia multitrayecto para una velocidad de bit de 19.492 Mbps en DTMB (FEC 0.8 y 32QAM) para la configuración de Rayleigh la relación señal a ruido es de 18.7, en el caso de DVB-T para una velocidad de bit de 18.7 Mbps la relación señal a ruido es de 21.5, es decir, se requieren casi 3 dB más en comparación con DTMB. En resumen, según [10] en las pruebas realizadas por LACETEL y de referencia para el estándar DTMB, éste muestra mejor desempeño que los estándares ATSC, DVB-T e ISDB-T, respecto a los datos oficiales del informe anexo a la recomendación UIT-R BT.2035-2.

1.2 Proceso de implementación de la TDT en Cuba.

La incursión de la TDT, televisión digital terrestre; en los países no comprende solamente consideraciones de carácter técnico, sino también legales, sociales y políticas, en donde se debe definir la tecnología a usar, cómo se realizará el proceso de transición, el tiempo definido para la migración hacia lo que se conoce como el apagón analógico[2].

En Cuba, desde la adopción del estándar de televisión digital DTMB en el año 2013, se ha ido desarrollando un proceso de implementación de esta tecnología. El programa de despliegue previsto tendrá tres etapas: una simultánea, en la que se mantendrán los dos tipos de transmisiones y que se prolongará durante unos cinco años; una segunda, que deberá comenzar a la altura del año 2016, y que incluirá la instalación del primer servicio de transmisión definitivo y el apagón analógico. En el último período se extenderá totalmente el servicio y se llegará a la alta definición. Para el año 2021 se prevé que concluya la transferencia en Cuba[12].

La elección de la norma china está determinada por diferentes cuestiones, dentro de las que se encuentra que la misma es producto de la evolución natural de la tecnología. Es la última

sacada al mercado y por tanto, se apoya en los éxitos y rectifica los problemas que presentaron en su despliegue las normas que la antecedieron. Se debe destacar que la norma tiene como característica que toda la propiedad intelectual es China y por este motivo, el gobierno tiene control sobre ella y nos ha garantizado el acceso a esta propiedad intelectual, incluso nos ha eximido del pago de royalty, regalías por el uso de estas patentes; lo que da una ventaja económica importante que se refleja en el costo del despliegue de la televisión digital en Cuba[13].

Además, se tuvieron en cuenta diferentes criterios técnicos producto de estudios realizados por LACETEL que demostraron un mejor desempeño de la norma DTMB sobre las demás variantes posibles.

Como resultado de los estudios realizados se han identificado debilidades y vulnerabilidades que pueden afectar la eficacia y eficiencia del proceso de implementación de la TDT. Principalmente se han presentado dificultades con el sistema de recepción de la señal, dígase antena, bajante o cable, conector, insuficiente oferta e incentivos que estimulen la migración, o sea servicios de valor agregado, precios asequibles, facilidades para la adquisición de los nuevos receptores, nuevos contenidos, etc. Además los recursos financieros para acometer las inversiones que demanda esta transición son limitados[2].

Para eliminar o mitigar estos riesgos la comisión técnica de la TDT ha propuesto líneas estratégicas de trabajo, las que serán implementadas con la participación de todos los organismos y entidades involucrados en el programa. Este programa que se propone tiene carácter nacional, aunque el Ministerio de las Comunicaciones es el Órgano de la Administración Central del Estado rector de las comunicaciones. Tiene un propósito real, de alta prioridad, orientado a la satisfacción de las necesidades de la población. El costo económico del proceso inversionista que lo respalda ha sido supeditado al interés de mantener un servicio de televisión gratuito y público. Se garantiza la participación de todos los sectores de la sociedad, en particular de la población que es sujeto de las transformaciones. Está articulado con los Lineamientos y posee interrelación con varios organismos del estado[2].

1.3 Características generales de la norma de televisión digital DTMB

El estándar de televisión digital DTMB está basado en el estándar chino GB20600- 2006 “Estructura de trama, codificación de canal y modulación para sistemas de televisión digital terrestre”[11]. Según este estándar el esquema en bloques de un modulador tomando como punto de partida la trama codificada MPEG-2, H.264 o AVS es el mostrado en la figura 1.4. El sistema convierte el flujo de datos de entrada en una señal de salida de radio frecuencia en banda VHF o UHF.

Este estándar soporta una razón de datos de carga útil de 4,813 Mbps a 32,486 Mbps, puede soportar televisión en definición estándar y en alta definición, receptores móviles y fijos, así como redes de múltiples frecuencias MFS (del inglés, ***Multiple Frequency Network***) y simple frecuencia SFN, (del inglés ***Single Frequency Network***)[11].

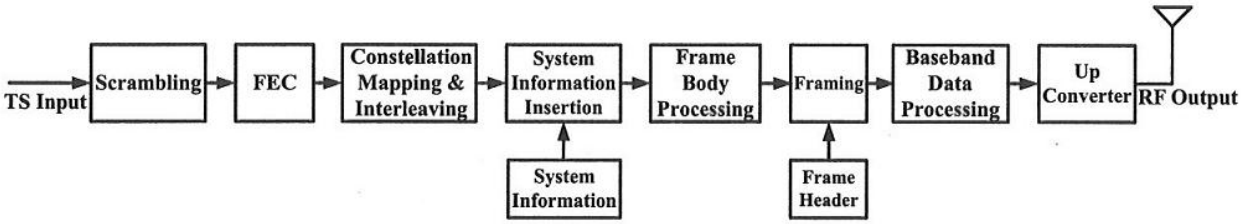

Figura 1.4: Estructura en bloques del modulador DTMB[14].

La estructura de las tramas de este estándar es jerárquica y posee 4 capas. La Trama de Señal es la unidad básica en la estructura de tramas de la norma. Una Trama de Señal está compuesta por 2 partes en el dominio de la señal en el tiempo, el Encabezado de Trama y el Cuerpo de Trama[11].

La razón de símbolo en banda base para el Cuerpo y el Encabezado es la misma. Como en Cuba el ancho de banda es de 6 MHz, la razón de símbolos pasa a ser 5.67 MHz, produciéndose una reducción de la carga útil a transmitir en una razón 3/4, en comparación con los 7.56 MHz que establece la norma para un ancho de banda de 8 MHz. La estructura de trama, sincronización, modulación, estimación de canal y corrección de errores son compatibles tanto en sistemas de 6 MHz como en 8 MHz[4].

El cuerpo de trama consta de 3744 símbolos de datos y 36 símbolos de información de sistema para un total de 3780 símbolos y una duración de 666.67 microsegundos, o sea, 3780/5.67 MHz. El encabezado puede tomar tres longitudes mientras que el cuerpo tiene un período de tiempo fijo, en todas las opciones de encabezado debe cumplirse que la razón de símbolos sea 5.67 MHz[11].

El estándar en su versión de 6MHz posee diferentes modos de transmisión según los servicios y ambientes de transmisión, cada modo puede variar en los siguientes parámetros:

Razón de código (FEC): 0.4, 0.6 ó 0.8.

Constelación de símbolos: 4QAM-NR, 4QAM, 16QAM, 32QAM ó 64QAM.

Profundidad del Entrelazado Temporal: 240 ó 720 símbolos.

Longitud de la cabecera: PN420, PN595 ó PN945.

Cantidad de portadoras: C = 3780 ó C = 1.

Existe un compromiso entre el modo a utilizar y la carga útil de información que puede ser transmitida, como es de esperar, un fortalecimiento de la señal representa una disminución de esta carga útil. Según [15] en el país en correspondencia con la programación que se quiere brindar se ha seleccionado el siguiente modo de transmisión:

Razón de código (FEC): 0.6.

Constelación de símbolos: 64QAM.

Profundidad del Entrelazado Temporal: 720 símbolos.

Longitud de la cabecera: PN420.

Cantidad de portadoras: C = 3780.

Este modo de transmisión permite una tasa binaria máxima de 18.2 Mb/s[15].

1.4 Codificación LPDC

Los códigos de chequeo de paridad de baja densidad LPDC (del inglés ***Low Density Parity Check)***, son unos poderosos esquemas de codificación los cuales pueden conseguir un buen desempeño en la corrección de errores bajo condiciones de baja relación señal ruido. Los mismos alcanzan un desempeño cercano al límite teórico máximo o límite de Shannon para bloques de transmisión extensos, además tienen una menor complejidad en el proceso de decodificación comparado con otros códigos FEC. Los códigos LDPC con un mayor tamaño tienen un mejor desempeño de corrección de errores con un alto costo de la implementación de hardware[16].

Para describir cómo funcionan los códigos LDPC es necesario considerar como funcionan los códigos de bloques lineales típicos, como por ejemplo, el código de Hamming. Para describir los códigos de bloques lineales la notación comúnmente utilizada es (n, k) donde k es el número de bits de información y n es el número de bits de la información codificada[17].

Un código de control de paridad de baja densidad es un código de bloque lineal dado por el espacio nulo de una matriz de control de paridad H (m×n) que tiene una densidad baja de unos. Un código LDPC regular es un código de bloque lineal cuya matriz de control de paridad H tiene el peso de columna g y de fila r, donde r=g * (n/m) y g << m. Si las filas y columnas de H no tienen un peso constante, entonces el código LDPC es irregular. Casi todos los procedimientos de construcción de H en estos códigos imponen que dos filas o columnas no tengan más de una posición en común que contenga un elemento distinto a cero, esta propiedad es llamada restricción fila-columna. El termino baja densidad no puede ser cuantificado con precisión, aunque una densidad de 0.01 o menor puede llamarse baja, o sea el 1% o menos de los bits de H son 1s. La densidad debe ser lo suficientemente baja para permitir una decodificación efectiva, siendo esto la clave de la invención de los códigos LDPC[18].

La representación gráfica de estos códigos se realiza mediante gráfico de Tanner, análogo a los códigos convolucionales de Trellis en que se proporciona una representación completa del código, lo cual ayuda en la descripción de decodificar algoritmos. Este es un gráfico cuyos nodos pueden ser separados en dos tipos, con bordes que conectan solo nodos de tipos diferentes. Los dos tipos de nodos en un gráfico de Tanner son; los nodos de variables o de bits de códigos y los nodos de chequeo denominados VN y CN respectivamente. En el gráfico CNi es conectado con VNj siempre que el elemento Hij sea un 1. De esta regla se define que hay “m” CNs, uno por cada ecuación de chequeo y “n” VN por cada bit de código. Las “m” filas de H especifican “m” conexiones CN y las “n” columnas especifican “n” conexiones VN[18].

La figura 1.5 mostrada a continuación ejemplifica un gráfico de Tanner, así como la figura 1.6 muestra la matriz correspondiente a dicho gráfico.

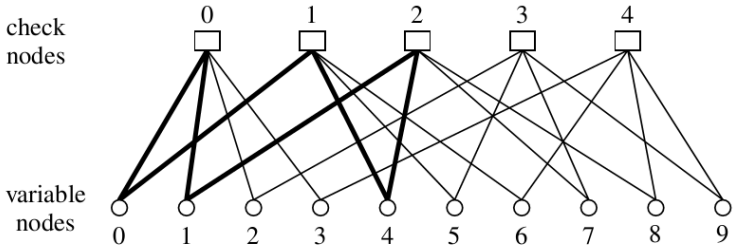

Figura 1.5: Gráfico de Tanner[18].

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Figura 1.6: Matriz correspondiente al gráfico de Tanner mostrado[18].

Los gráficos de Tanner de un código LDPC actúan como un patrón para el decodificador iterativo. La eficiencia del decodificador iterativo depende de las propiedades estructurales del gráfico en el cual está basado el decodificador. En el gráfico, una secuencia de puntos de conexión o bordes los cuales forman un camino cerrado son llamados ciclos, pequeños ciclos degradan el rendimiento de los algoritmos de decodificación iterativos. En el decodificador una alta densidad, o sea que cerca de la mitad de las entradas sean unos, hace que existan varios ciclos pequeños impidiendo el uso de un decodificador iterativo, siendo necesario que la matriz de H sea de baja densidad.

La distancia de un ciclo es igual al número de puntos de conexión que lo forman. El grafico mostrado anteriormente es un gráfico regular, donde la distancia de ciclo es 6. Cada VN tiene 2 puntos de conexión y cada CN tiene 4, o sea en grado de VN es 2 y CN es 4, lo cual está en concordancia con g=2 y r=4. Siendo claro que m * r = n * g para todos los códigos LDPC regulares e igual al número de puntos de conexión o bordes en el gráfico.

La matriz de chequeo de paridad H de un código cíclico es una matriz circular n x n; o sea cada fila es un cambio cíclico de una fila anterior, siendo la primera fila un cambio cíclico de la última. Este tipo de matriz H tiene una ventaja sustancial para los decodificadores LDPC pues cada ecuación de chequeo es relativamente cercana a la predecesora y sucesora. Sin embargo la complejidad de los códigos cíclicos LDPC es que la matriz H es n x n, independientemente de la razón de código lo que implica mayor complejidad en el decodificador y que los códigos tienden a tener pesos de filas grandes lo cual hace complicada la implementación del decodificador. Por otra parte, los códigos cuasi cíclicos poseen una estructura que conduce a simplificar los diseños del codificador y decodificador. Ellos permiten mayor flexibilidad en el diseño del código lo cual conduce a códigos mejorados con relación a los códigos cíclicos LDPC. La matriz H de los códigos cuasi cíclicos es generalmente representada como un arreglo de matrices circulantes[18].

$$\mathbf{H} = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & & \vdots \\ A_{M1} & \cdots & A_{MN} \end{bmatrix},$$

Figura 1.7: Matriz H de códigos cuasi cíclicos[16].

Cada elemento de la matriz A_{MN} es una matriz circulante q x q. Para códigos LDPC, las circulantes deben ser escasas de peso 1, esto significa que el peso de cada fila y columna es 1. Para efectuar irregularidad, algunas de las matrices pueden ser de todos ceros q x q usando una técnica denominada enmascaramiento.

Los códigos LDPC, así como las técnicas de construcción de los mismos pueden ser divididas en tres clases; cíclicas, cuasi cíclicas y aleatorios. Para las primeras técnicas de construcción se utilizan algoritmos computacionales, las segundas consisten en algoritmos finitos de matemáticas, incluyen algebra, combinatoria y teoría de gráficos. Las técnicas de construcción asistidas por ordenador pueden conducir a cualquier código LDPC arbitrario.

1.5 Aleatorizador o dispersor de energía

La norma DTMB presenta un primer proceso anterior a la codificación de los datos de información llamado aleatorizador o dispersor de energía.

La densidad espectral de potencia depende de la correlación entre los símbolos y la forma del pulso conformador que se transmite. En cuanto al pulso conformador, en caso de multiplexación por división de frecuencia ortogonal OFDM es un pulso rectangular. Respecto a la correlación de símbolos, si los símbolos no presentan ninguna correlación, el espectro depende únicamente de la transformada de Fourier del pulso conformador. Ahora bien, si los símbolos están correlados, por ejemplo, hay ráfagas de muchos 1s seguidos en el espectro de la señal aparecen rayas espectrales; picos que pueden saturar a los amplificadores de potencia y que, por otro lado desperdician potencia porque son picos que no aportan información. Por esta razón, el flujo de entrada debe ser tratado para convertirlo en pseudoaleatorio[1].

El Aleatorizador es un elemento que genera una secuencia de bits pseudoaleatoria PRBS (del inglés, ***Pseudo Random Bits Sequence***). El polinomio generador de la PRBS que establece la norma está dado por la Ecuación:

$$G(x) = 1 + x^{14} + x^{15} \quad (1)$$

La secuencia pseudoaleatoria se construye a base de un registro de desplazamiento con realimentación lineal o LFSR, (del inglés, ***Lineal Feedback Shift Register***). Un LFSR es un registro de desplazamiento cuya entrada es una función lineal de su estado anterior. La función lineal que utiliza es un OR exclusivo, así la entrada es manejada por esta operación lineal en la que intervienen varios elementos del registro[4].

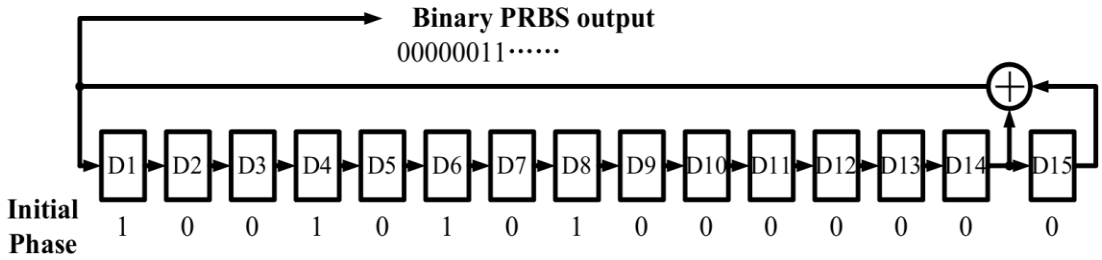

Figura 1.8: Estructura del LFSR del aleatorizador[11].

El estado inicial del registro de desplazamiento realimentado linealmente LFSR, es: "100101010000000" a partir del cual se genera la PRBS. El primer bit a la salida del generador PRBS será aplicado al primer bit del flujo de entrada mediante una operación XOR para aleatorizar los datos[6].

Este proceso será llevado a su estado inicial al comienzo de cada Trama Señal, o sea que la cantidad de bits a aleatorizar depende de la cantidad de bits necesarios para completar el cuerpo de trama; lo que a la vez depende de la razón de código y el mapeado que se emplee. En la siguiente tabla se relacionan la cantidad de bits a aleatorizar en dependencia de la razón de código y el mapeado.

En la siguiente tabla se relacionan esta información.

Tabla 1.2 Bits a aleatorizar en relación de razón de código y mapeo

	Razón de Código		
Mapeado	0,4	0,6	0,8
4 QAM	3008	4512	6016
16 QAM	6016	9024	12032
32 QAM	7520	11280	15040
64 QAM	9024	13536	18048

1.6 Codificación de canal de la norma DTMB

El bloque funcional corrector de errores progresivo FEC (del inglés, ***Forward Error Correction***), se encarga de agregar los bits de chequeo necesarios para proteger a los datos de información de errores provocados por diversos factores durante la transmisión.

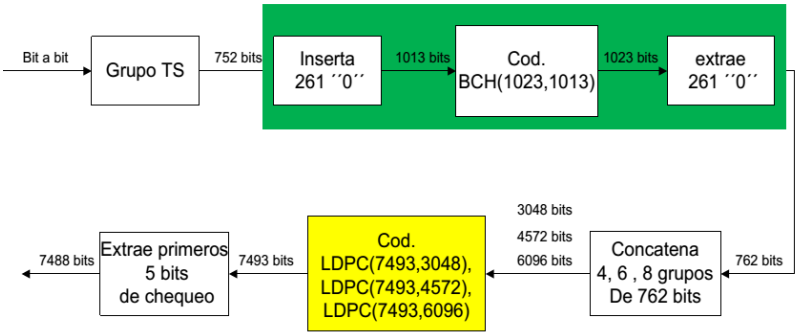

Figura 1.9: Proceso de codificación FEC utilizado en la norma DTMB[6].

El mismo se compone de un codificador externo BCH y un codificador interno de Chequeo de Paridad de Baja Densidad, LDPC.

1.6.1 Codificador externo BCH de la norma DTMB

El código BCH tiene 2 propósitos. El primero es adaptar la tasa de datos de entrada MPEG2 a las palabras LDPC según sus 3 posibles modos de trabajo. El otro propósito es reducir el umbral de error del sistema, mediante la corrección de errores esporádicos del codificador interno, en la forma de corrección de un error simple o detección de uno doble[6].

El codificador externo es del tipo BCH (762,752) correspondiente a 752 bits de información y 10 de chequeo; este se obtiene del código BCH (1023,1013), mediante la inserción de 261 ceros a los bits de información. Una vez obtenida la palabra de código se extraen los 261 ceros ubicados en la cabecera de la misma, esto es posible pues el código es sistemático, ubicando los bits de información por delante de los de chequeo. Una Trama de Transporte MPEG-2 contiene 188 Bytes o lo que es lo mismo 1504 bits. La codificación BCH se efectuará sobre 752 bits correspondientes a la mitad de dicha Trama de Transporte[6].

El código BCH es una generalización del código de Hamming, el cual permite la corrección de errores múltiples. El código BCH se describe por los elementos enteros: n, m y t, donde se deben cumplir las ecuaciones siguientes según se plantea en [18].

$$n = 2^m - 1 \quad (2)$$

$$t < 2^n - 1 \quad (3)$$

$$n - k = mt \quad (4)$$

$$m \geq 3 \quad (5)$$

$$dmín \geq 2t + 1 \quad (6)$$

Donde n es el número de bits de cada palaba de código, k es el número de bits de mensaje, t es la cantidad de errores que puede corregir, m la cantidad de bits de chequeo mínima para corregir t errores y dmín la distancia mínima entre dos palabras de código[18].

Analizando las ecuaciones anteriores se justifica la necesidad del relleno efectuado sobre los bits de información, pues el código BCH (762,752) no cumple con estas características.

El codificador externo BCH utilizado en la norma es común para las tres razones del codificador interno y es capaz de corregir un error simple o detectar uno doble, su objetivo es corregir errores esporádicos del codificador interno.

La norma describe el polinomio generador que establece las palabras válidas del código según la ecuación

$$G_{BCH}(x) = 1+x^3+x^{10} \quad (7)$$

El mismo es una subclase de los códigos cíclicos y se puede implementar con registros de desplazamientos realimentados, lo cual disminuye sustancialmente la complejidad del hardware.

1.6.2 Codificador interno LDPC de la norma DTMB

Luego de la codificación BCH, los datos son entregados al codificador LDPC correspondiente al FEC interno de la norma DTMB. Las tres razones de código que son 0.4, 0.6 y 0.8, están dadas por la cantidad de grupos de bits codificados BCH que tienen que concatenarse para entrar al codificador LDPC.

Tabla 1.3 Parámetros de la codificación LDPC

Longitud de bloques(bits)	Bloque de entrada	Bits de información	Razón de código
7488	3048	3008	0,4
7488	4572	4512	0,6
7488	6096	6016	0,8

La longitud de estos grupos será de 3048, 4572 ó 6096 bits, tal como muestra la tabla 1.3 la cual muestra la cantidad de bits que intervienen en el proceso de codificación LDPC según la razón de código que se utilize; a los mismos se le añaden los bits redundantes del codificador LDPC para formar palabras de longitud fija e igual a 7493 bits. Finalmente se retiran los primeros cinco bits de chequeo de la palabra codificada. La extracción se justifica con el objetivo de transmitir en una Trama Señal unidad básica de transmisión del estándar DTMB, o sea un número fijo de Tramas de Transporte (TS) de 188 bytes[7].

EL codificador LDPC es de baja densidad pues es un código de bloque lineal representado por una matriz de chequeo de paridad H (m x n) con baja densidad de 1s; se considera baja la densidad si es menor del 1% de todos los elementos de la matriz H[6].

Esta matriz está conformada por submatrices de (b x b) elementos con b = 127 y pesos de fila 1 cumpliendo con la condición de baja densidad, o sea, 1<< 127. La densidad de la matriz compuesta en su conjunto es del 0.26 % no igual a 1/127 debido a la existencia de submatrices todos ceros, con lo cual el valor es mucho menor que la máxima teórica del 1 %, siendo suficientemente baja como para permitir decodificaciones iterativas efectivas. Esto es en efecto, la clave innovadora detrás de la invención de los códigos LDPC[7].

La norma presenta la matriz generadora G_{qc}, donde I es la matriz identidad (127x127); 0 es una matriz ceros (127 x127) y $G_{i,j}$ es una matriz circulante (127x127) la cual cumple 0 ≤ i <k-

1 y 0 ≤ j< c-1. Dicha matriz generadora es de forma cuasi-circular sistemática. Las palabras de códigos obtenidas son de 7493 bits[11].

Las dimensiones de la matriz varían según la razón de código que se pretenda utilizar.

a) Razón de código 0.4 k=24 y c=35
b) Razón de código 0.6 k=36 y c=23
c) Razón de código 0.8 k=48 y c=11

$$\mathbf{G}_{qc} = \begin{bmatrix} \mathbf{G}_{0,0} & \mathbf{G}_{0,1} & \cdots & \mathbf{G}_{0,c-1} & \mathbf{I} & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{G}_{1,0} & \mathbf{G}_{1,1} & \cdots & \mathbf{G}_{1,c-1} & \mathbf{O} & \mathbf{I} & \cdots & \mathbf{O} \\ \vdots & \vdots & \mathbf{G}_{i,j} & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{G}_{k-1,0} & \mathbf{G}_{k-1,1} & \cdots & \mathbf{G}_{k-1,c-1} & \mathbf{O} & \mathbf{O} & \cdots & \mathbf{I} \end{bmatrix}$$

Figura 0.10: Matriz generadora del código LPDC[11].

En la norma se especifican los valores de cada una de las filas de las submatrices. Ver [11]

CAPÍTULO 2. Codificación de canal de la norma DTMB con FPGA de Xilinx

En este capítulo se abordan las diferentes estrategias de diseño de sistemas digitales con FPGA de Xilinx y se especifica cual se utiliza para el desarrollo de este trabajo. Se describe la herramienta de software ***Xilinx System Generator XSG***. Se realiza el diseño de las diferentes etapas de codificación de canal de la norma DTMB mediante el empleo de bloques funcionales de Xilinx System Generator.

Todos los diseños de este capítulo pueden consultarse en la carpeta LDPC.

2.1 Diseño de hardware de sistemas digitales

El diseño de hardware tiene un problema fundamental que no se evidencia en el diseño de software. Este problema es el alto costo del ciclo diseño, prototipo, verificación, donde el costo del prototipo por lo general es bastante elevado.

En busca de minimizar el costo de este ciclo se incluye la fase de simulación y verificación que elimina la necesidad de elaborar físicamente un prototipo donde las diversas herramientas permiten realizar simulación por eventos, funcional, digital o eléctrica considerando el nivel de simulación requerido.

Una de las herramientas que pueden utilizarse durante el diseño e implementación de hardware es la programación de dispositivos VLSI (del inglés, ***Very Large Scale Integrated Circuits***) donde se emplea lógica programable o también llamada configurable. Los dispositivos de lógica programable permiten implementar circuitos mediante la programación de los mismos, los que posteriormente pueden ser reutilizados en caso de querer modificar el diseño. Ejemplos de dichos dispositivos son los FPGA, (del inglés, ***Field Programmable Gate Arrays***) y PLD (del inglés, ***Programmable Logic Devices***).

Existen dos métodos de diseño de hardware digital con dispositivos VLSI de lógica programable: el **Bottom-Up** y el **Top-Down**.

La metodología Bottom-Up comprende la descripción del circuito mediante componentes que pueden agruparse en diferentes módulos, y éstos últimos a su vez en otros módulos hasta llegar a representar el sistema completo que se desea implementar. Esta metodología no implica una estructuración jerárquica de los elementos del sistema simplemente reúne componentes de bajo nivel para formar el diseño global[19].

Esta metodología de diseño es útil para diseños medianamente pequeños. Para diseños mayores ésta metodología resulta impráctica. El hecho de unir un número elevado de componentes entre sí, sin una estructura jerárquica que permita organizarlos dificulta el análisis del circuito, aumentando la posibilidad de cometer errores.

Por otra parte el diseño Top-Down consiste en capturar una idea con un alto nivel de abstracción, implementarla partiendo de la misma, e incrementar el nivel de detalle según sea necesario. El sistema inicial se va subdividiendo en módulos, estableciendo una jerarquía. Cada módulo se subdivide cuantas veces sea necesario hasta llegar a los componentes primarios del diseño[20].

La metodología Top-Down tiene como ventaja que la información se estructura en forma modular. El diseño se realiza a partir del sistema completo y se subdivide en módulos, esto permite que las subdivisiones se realicen de forma que los mismos sean funcionalmente independientes. El diseño Bottom-Up tiene la desventaja que no contempla la división en partes funcionalmente independientes. Además, estos diseños deben organizarse de tal forma que resulte fácil su comprensión. Una forma de organizar el diseño es la creación de un diseño modular jerárquico, el cual está constituido por niveles en donde cada uno es una especialización del nivel superior[19].

Debido a los elementos que se explican anteriormente, en este trabajo se decide emplear la metodología Top-Down.

Las herramientas actuales permiten utilizar en forma automática la metodología Top-Down, lo que permite a las herramientas de síntesis sofisticadas llevar a cabo la implementación de un circuito final, partiendo de una idea abstracta y sin necesidad de que el diseñador deba descomponer su idea inicial en componentes concretos[20].

Además al desarrollar cualquier sistema digital es importante seguir ciertas pautas de trabajo y tener en cuenta factores muy diversos para que el diseño pueda terminarse a tiempo y funcione correctamente. A medida que los diseños se hacen más complejos, la necesidad de seguir un método ordenado para lograr buenos resultados se hace más importante. Para poder atacar el problema del diseño de sistemas digitales complejos sobre FPGA, es importante tener una metodología de trabajo que permita planificar y ordenar el trabajo.

Las etapas de diseño de sistemas digitales en general son las mencionadas a continuación.

- *Especificación y Diseño*

Las especificaciones deben describir la solución de manera de cumplir con los requerimientos que se piden para el dispositivo. Una especificación debe comprender un diagrama en bloques del sistema que muestre los principales bloques funcionales, una descripción de las entradas/salidas, incluyendo interfaces lógicas, eléctricas y protocolos de comunicación. Estimaciones de tiempos que se deben cumplir, incluyendo tiempos de establecimiento y mantenimiento para las entradas, salidas y frecuencias de reloj[21].

- *Verificación:*

La verificación engloba varios pasos como la simulación, estas se hacen sobre pequeñas partes del sistema y sobre el sistema completo. Se debe llevar a cabo una simulación funcional, pero también puede incluir simulaciones de temporizado, consumo de potencia y otros parámetros. Otra etapa de la verificación es la implementación física y revisión; una vez que se ha aceptado el diseño se lleva a cabo la implementación física final del dispositivo y se verifica para asegurarse que su funcionamiento coincide con las simulaciones hechas

anteriormente. En este paso se deben también evaluar los tiempos, consumo de potencia y cualquier otro parámetro de importancia que al revisar pueden surgir cosas que obliguen a volver atrás hacia pasos anteriores[21].

2.1.1 Descripción de sistemas digitales basados en FPGA.

El aumento de la complejidad de los sistemas digitales ha ido acompañado por el desarrollo de múltiples herramientas de diseño. Existen diferentes métodos de descripción de un circuito en una FPGA en los que se diseña de forma independiente de la tecnología. O sea el diseñador no tiene que conocer la arquitectura interna de la FPGA, ni las características de los circuitos semiconductores.

La manera más intuitiva y usada en la descripción de un diseño de sistemas digitales basados en FPGA es la edición de esquemáticos. Esto es llamado captura de esquemáticos y en el proceso se describe el sistema completo, pudiendo usarse niveles de jerarquía o sistemas ya creados. Existe un formato estándar de intercambio de diseño electrónico EDIF, (del inglés, ***Electronic Design Interchange Format***) independiente de los fabricantes; este puede usarse para intercambio de esquemáticos, pero los sistemas están descritos en modo de texto[22].

Otro método es mediante los lenguajes de descripción de hardware HDL, (del inglés, ***Hardware Description Language***). Estos permiten describir sistemas digitales usando texto, facilitando su diseño y modificación. En los lenguajes de programación ordinarios las sentencias se ejecutan de forma secuencial, y se mantiene la dependencia entre ellas aunque haya una ejecución paralela. Mientras en los HDL las sentencias son concurrentes; es decir, se ejecutan de forma simultánea, tal y como se comportan un grupo de puertas lógicas en un circuito digital. Para ejecutar un grupo de sentencias de forma secuencial se han de incluir dentro de lo que se conoce con el nombre de proceso. Un proceso es la descripción de un sistema secuencial, que pasa por una serie de estados. Los procesos son concurrentes entre ellos; es decir, se ejecutan de forma simultánea[23].

Existen dos HDL estándar y ampliamente usados. Uno es VHDL que es un doble acrónimo ***Very High Speed Integrated Circuit Hardware Description Language***, ***VHSIC-HDL***. La traducción correcta de este acrónimo sería “lenguaje muy rápido de descripción hardware para circuitos integrados”. El otro HDL estándar es Verilog. VHDL y Verilog son lenguajes estándar definidos por IEEE y han sufrido diferentes revisiones. Verilog es más compacto que VHDL, y más fácil de aprender y usar. Esta facilidad se debe a su parecido con el lenguaje C. Los dos lenguajes se usan para el diseño en FPGA. Verilog es preferido para el diseño en sistemas ASIC por acceder a niveles más bajos en hardware, pero VHDL permite el uso de tipos de datos más complejos.

2.1.2 Estrategia de diseño de sistemas digitales con FPGA.

Cuando se diseña con lógicas programables, cualquiera sea el método usado para describir el diseño, el proceso desde la definición del circuito por el desarrollador hasta tenerlo funcionando sobre un FPGA implica varios pasos intermedios y en general utiliza una variedad de herramientas. A este proceso se lo denomina ciclo o flujo de diseño. Para cada

uno de estos pasos se utilizan herramientas de software diferentes que pueden o no estar integradas bajo un ambiente de desarrollo.

A continuación se describen los pasos del flujo de diseño según se plantea en [24].

Descripción del Diseño. En este paso se describe el diseño, usando un diagrama esquemático o estructural, un lenguaje de descripción de hardware como el VHDL, una máquina de estados o una tabla de entrada-salida. Muchas herramientas permiten utilizar todos estos métodos.

Generación o Traducción. Este paso tiene sentido cuando el diseño no se hace mediante lenguajes de descripción de hardware. En este paso se traducen todos los módulos a VHDL. Para los módulos ingresados como VHDL en esta etapa se hace un análisis del VHDL para verificar la sintaxis y semántica de los módulos.

Compilado. Los simuladores actuales compilan el código VHDL a un formato que permite una simulación más rápida y eficaz.

Simulación y verificación. En este paso se simula el comportamiento del diseño y se evalúa su comportamiento. La simulación puede hacerse en dos etapas diferentes del diseño. La primera es sobre el diseño original para verificar el correcto funcionamiento del mismo. La segunda es después de sintetizar el circuito donde se simula la implementación real sobre el FPGA. Esta es la más exacta, engorrosa y lenta, ya que incluye la información final lógica y temporal del diseño sobre el FPGA.

Síntesis. En este paso se traduce el diseño a su implementación con lógica digital, utilizando los componentes específicos del FPGA que va a utilizarse. Esta traducción puede llegar hasta el nivel más básico de elementos lógicos del FPGA en el que se pretenda implementar el diseño o hasta un nivel superior, en el que el diseño se presenta en módulos básicos estándar provistos en una librería por el proveedor del dispositivo.

Ubicación e Interconexión. El FPGA está compuesto por muchos bloques idénticos. En este paso cada componente del diseño sintetizado se ubica dentro del FPGA específico. También se interconectan los componentes entre sí y con los pines de entrada-salida. Puede definirse la interconexión de las señales con los pines físicos del FPGA.

Generación de Binarios. Después de la ubicación e interconexión se genera algún archivo ya sea para poder utilizar el sistema en un diseño más complejo o para programar un FPGA.

Configuración de FPGA. Con el archivo binario generado puede configurarse directamente un FPGA a través de alguna de las opciones de configuración. Estas opciones dependerán de las herramientas y del dispositivo que se esté utilizando.

Verificación final. Una vez integrado el FPGA con su programación a un sistema debe hacerse una verificación para controlar que el diseño funciona bien y que el FPGA se integra bien al sistema en el que está. Pueden usarse técnicas y herramientas de verificación automáticas para evaluar si el dispositivo y el diseño están funcionando como debieran.

2.2 Herramientas para el diseño de sistemas digitales con FPGA de Xilinx

Los suministradores de FPGA disponen de diferentes entornos de diseño estándar, en ellos se pueden describir los circuitos de las formas descritas anteriormente y siguiendo el flujo de diseño que anteriormente se describió, aunque algunos de estos pasos sean transparentes al usuario. Estas herramientas no son en principio gratuitas, aunque se pueden pedir como donación a instituciones académicas. También tienen versiones de demostración, por limitación en el tiempo de uso o por el tamaño del diseño. Estas herramientas permiten compilar y simular el diseño, para finalmente generar el fichero de programación para la FPGA del fabricante. Estos entornos gestionan el conjunto de ficheros generados en el diseño, asignan los pines de entrada-salida, seleccionan opciones para la compilación y dan información de los recursos de hardware necesarios.

La empresa Xilinx es la que acapara la mayor parte del mercado de FPGA a nivel mundial. Actualmente Xilinx dispone de dos entornos de diseño estándar; por un lado ISE, del inglés ***Integrated System Environment***; y por otro Vivado; siendo el primero el más utilizado, pese a que Xilinx anunció descontinuar dicho entorno en el año 2015. Junto a estos entornos se dispone de System Generator, que es la utilidad de diseño sobre Matlab/Simulink.

En el diseño de sistemas digitales basados en FPGAs se utiliza el entorno de diseño ISE ***Integrated Software Environment*** de Xilinx consistente en una herramienta de software que permite realizar un diseño digital basado en lógica programable, la misma permite recorrer las diferentes fases del proceso de desarrollo de un circuito lógico, desde el diseño hasta la implantación sobre una arquitectura reconfigurable; descripción, simulación, síntesis, implementación, simulación temporal y programación. Cada una de las etapas del flujo de diseño puede realizarse dentro del entorno integrado o bien utilizar herramientas de terceros, algunas de esas herramientas también son integrables en ISE si están convenientemente instaladas[21].

El entorno Matlab se ha convertido en un estándar tanto para la comunidad académica y científica, como para la industria, siendo Xilinx System Generator la herramienta que distribuye Xilinx para la realización de diseños con FPGA utilizando Matlab/Simulink.

En investigaciones anteriores consultadas, ver[21]; se describe una metodología de diseño mediante el entorno de ISE, la misma abarca el desarrollo de aplicaciones de mediana complejidad y de carácter académico; mientras en el diseño de sistemas digitales complejos que incluyen bloques funcionales típicos de la transmisión de datos, una de las maneras de más eficientes de abordar estos diseños es mediante ***Xilinx System Generator (XSG)***, ya que este incluye herramientas las cuales posibilitan la implementación a alto nivel del diseño directamente en una FPGA, permite la interacción de esta herramienta con Matlab/Simulink, lo que facilita la modelación y simulación a alto nivel de abstracción. Además, mediante XSG es posible realizar la simulación del sistema diseñado introduciendo los datos o variables al sistema desde Matlab/Simulink, así mismo se obtienen los datos y señales de salida del sistema para su análisis y revisión desde esta plataforma.

Otra de las cuestiones asociadas a las ventajas de utilizar esta herramienta, es que la compañía Xilinx desarrolla módulos o bloques funcionales de transmisión de datos los cuales solo es necesario parametrizar o configurar según el diseño a realizar; además los mismos

se van actualizando y se agregan nuevos bloques funcionales de diferentes sistemas de comunicaciones específicos los cuales pueden simplificar el diseño de un sistema de comunicaciones en general.

Una potencialidad que posee esta herramienta es la posibilidad de realizar una simulación de hardware llamada en la bibliografía Hardware Co-Simulación, donde es posible ejecutar el diseño directamente en la FPGA ya que permite comunicar las señales de Matlab con la placa que contiene la FPGA. Esta simulación de hardware permite realizar una corrida en tiempo real del sistema directamente en el FPGA desde la interfaz de Simulink, o sea es posible establecer una interacción entre Simulink y la FPGA de la placa de desarrollo. Esta es una manera de comprobar el sistema y verificar el comportamiento del mismo sobre la FPGA mediante las herramientas de Simulink, los datos y variables de entrada se suministran a través de Simulink donde también se visualizan los resultados pero en este caso los cálculos y funciones lógicas se realizan en el hardware de la tarjeta de desarrollo; esto permite realizar un análisis temporal del comportamiento del sistema desde su funcionamiento en el hardware.

2.2.1 Xilinx System Generator.

XSG es un conjunto de bloques integrados con MATLAB/Simulink. Es un software para traducir los modelos Simulink a una realización hardware del modelo. XSG mapea los parámetros definidos en Simulink en la versión hardware del modelo; puertos, señales y atributos, además produce de forma automática un fichero de comandos para la síntesis en la FPGA, genera un modelo de simulación en lenguaje de descripción de hardware y realiza la implementación, de tal forma no es necesario dejar el entorno grafico en ningún momento o etapa del diseño. Dicha herramienta nos permite simular funcionalmente los diseños y usar el entorno MATLAB para verificar los modelos a nivel de bit y ciclo con los resultados de los modelos de referencias. Estos resultados de referencias pueden estar generados dentro o fuera del MATLAB. En adición a esto la herramienta nos permite programar la FPGA desde el mismo entorno proporcionado por XSG y MATLAB[25].

XSG puede ser usado para el diseño y las pruebas de sistemas de procesamiento digital de señales para FPGA en un ambiente de flujo de datos visuales como MATLAB/Simulink. Es posible utilizar los bloques de XSG en el diseño y generar un diseño sintetizable que puede ser puesto en práctica usando el navegador de proyecto de ISE Xilinx. El diseño propiamente puede realizarse mediante la interconexión de los bloques lógicos de Xilinx que brinda la herramienta y mediante lenguaje de descripción de hardware VHDL a través de un bloque denominado Caja Negra disponible que permite realizar modelos basados en lenguaje de descripción de hardware VHDL o Verilog.

Las herramientas de síntesis de comportamiento leen las descripciones de alto nivel de las aplicaciones de procesamiento digital de señales escritas en MATLAB y automáticamente generan modelos sintetizables RTL (del inglés, ***Register Transfer Level***).

Los tipos de archivo de entrada al sintetizador son .V para Verilog o .VHD para VHDL, mientras el archivo de salida de Synplify es un fichero EDIF (***Electronic Data Interchange Format***); y el archivo salida de XST es un archivo NGD (***Native Generic Database***).[26]

Como en esta etapa no se ha trazado un mapa de componentes básicos específicos de Xilinx, las herramientas de síntesis no pueden brindar resultados temporales exactos; solo estimados. La salida de las herramientas de síntesis entonces alimenta la siguiente etapa del flujo de diseño, que es la llamada implementación en el flujo de diseño y es la herramienta principal de la suite de software ISE.

Antes de que se ejecute la implementación es necesario crear un archivo UCF (del inglés, ***User Constraints File***). La información principal de este archivo es la posición específica de cada uno de los pines de entrada y salida en el archivo de diseño HDL, la información de tiempo así como la frecuencia del reloj del sistema[26].

Los archivos de salida de la síntesis son los archivos con formato EDIF, NGD y UCF, los mismos representan un mapa optimizado del diseño, restricciones de tiempo y la asignación de pines del FPGA.

Todo este proceso y archivos que ayudan finalmente a la generación del archivo de configuración .bit son transparentes para el usuario. O sea, que mediante System Generator es posible realizar todos los pasos necesarios para llevar a cabo el diseño de un sistema digital hasta la generación del fichero de configuración .bit el cual es cargado a la FPGA en la placa de desarrollo que se pretenda utilizar.

En este trabajo se determina utilizar XSG en todas las etapas hasta antes descritas hasta la generación del fichero de configuración .bit, mientras que la descarga a la FPGA se plantea utilizar la herramienta de software Impact, la cual se instala mediante la instalación del paquete ISE Xilinx.

En la instalación de System Generator es necesario tener presente la compatibilidad de los diferentes paquetes que deben ser instalados. O sea es necesario que exista compatibilidad entre ISE, System Generator y Simulink de Matlab. La versión que se utiliza en este proyecto es Xilinx System Generator XSG 14.7. Para poder utilizar XSG es necesario haber instalado en orden, primero el Matlab, en este caso versión 2013a y luego el software ISE Xilinx 14.7.

La figura 2.1 muestra la interfaz de Xilinx System Generator, donde se pueden observar parte de los bloques que pueden ser utilizados en el diseño de un sistema.

Una vez instalado correctamente el System Generator se tienen como resultado tres set de bloques en Simulink los cuales a su vez están compuestos por diez bibliotecas las cuales contienen gran cantidad de bloques funcionales con los cuales se genera el diseño del sistema a implementarse en el FPGA. Cada uno de estos bloques posee parámetros configurables, estos pueden ser específicos o comunes en la mayoría de los bloques.

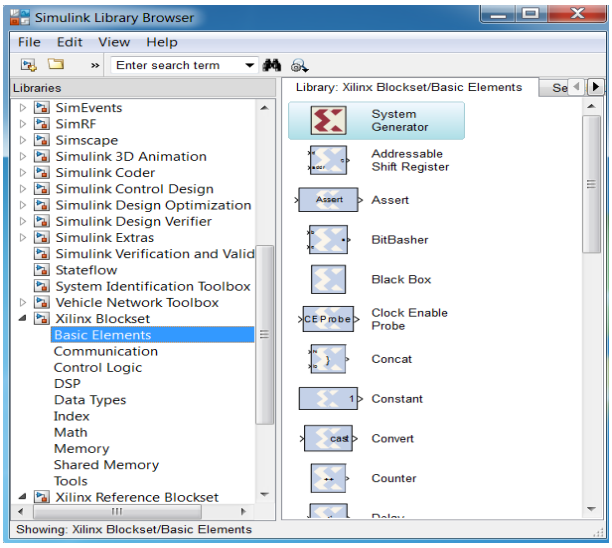

Figura 2.1: Bloques de Xilinx en XSG.

2.3 Diseño de etapas de codificación de canal de la norma DTMB

A continuación se describe el diseño de las etapas aleatorizador y codificación del canal de televisión digital terrestre TDT para la norma DTMB para ser implementado sobre plataforma FPGA de fabricante Xilinx, se pretende realizar el diseño mediante la herramienta Xilinx System Generator. La metodología utilizada se basa en realizar el diseño mediante bloques propios de Simulink y por otra parte mediante bloques de Xilinx; se sigue esta idea con el objetivo de tomar como referencia los resultados obtenidos mediante la primera variante que utiliza bloques de codificación propios de Simulink para comprobar los resultados que se obtienen mediante los bloques de Xilinx que son los que se pueden implementar en hardware.

El sistema diseñado se hace funcionar a una frecuencia de reloj de 54MHz pues según se especifica en [27] esta es la frecuencia mínima a la que se deben procesar los datos de entrada mpeg para no perder carga útil.

2.3.1 Aleatorizador

Para realizar el diseño del aleatorizador se utilizan bloques funcionales de registro de desplazamiento con realimentación lineal LFSR, (del inglés, ***Lineal Feedback Shift Register***) disponible en XSG.

En la configuración de los LFSR se define la cantidad de bits, el estado inicial de los mismos así como los bits que intervienen en la realimentación que en este caso se determina que sea el último solamente 0001h = 000000000000001b. La cantidad de bits es 14 y 15 respectivamente siendo el estado inicial 00A9h = 000000010101001b para ambos registros. La salida de los mismos se aplica a un or exclusivo obteniéndose la secuencia pseudoaleatoria; dicha secuencia es llevada a otro or exclusivo con la secuencia de datos de entrada con lo cual se consigue aleatorizar los datos. La figura 2.2 muestra el diseño realizado.

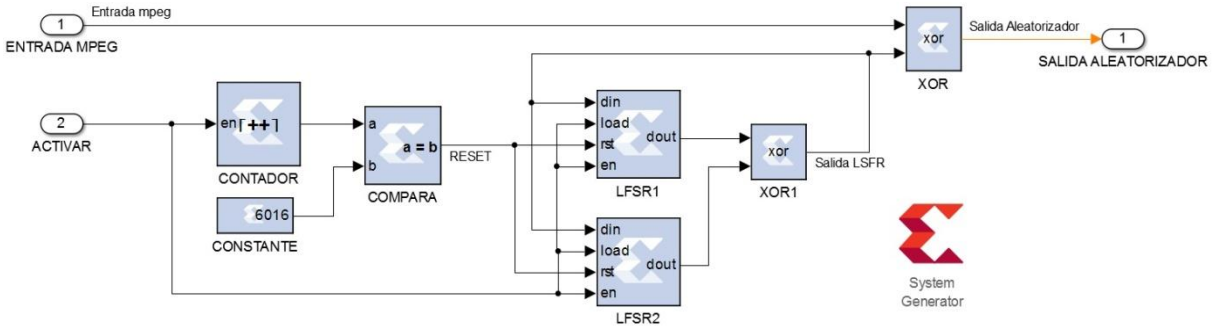

Figura 2.2: Diseño de aleatorizador.

Los bloques contador y comparador se utilizan con el objetivo de resetear ambos registros de desplazamiento a su valor inicial. El valor de la constante indica la cantidad de bits aleatorizados que se deben contar para resetear el LFSR. Este valor depende de la razón de código y del mapeo seleccionado. El valor seleccionado corresponde a la combinación de razón de 0.4 con mapeo 16QAM, esta combinación transporta 6016 bits de información en cada trama señal; ver tabla 1.2.

2.3.2 Etapa de codificación externa BCH con bloques de XSG.

El bloque de codificación BCH se realiza primeramente mediante bloques funcionales de XSG los cuales pueden ser implementados en un FPGA y llevados a su realización hardware, además se realiza el diseño mediante Simulink, donde se dispone de un bloque funcional de codificación BCH.

El diseño de la etapa de codificación BCH en XSG se divide en dos partes. Primeramente se diseña una etapa que acomoda los datos que provienen del aleatorizador y posteriormente una etapa de codificación propiamente según plantea el estándar. La figura 2.3 muestra este diseño.

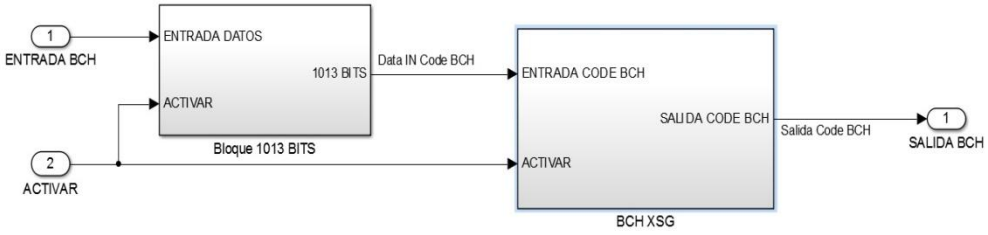

Figura 2.3: Diseño de codificación externa BCH.

En el diseño de esta primera etapa mediante bloques de Xilinx System Generator se utiliza un bloque de memoria FIFO, (siglas en inglés, ***first in first out***); en este se van almacenando los datos provenientes del aleatorizador, a su vez se implementa una etapa de control la cual maneja los tiempos de lectura de la FIFO y maneja el control del multiplexor. Esto logra la adicción de 261 ceros anteriores a los 752 datos que se toman desde la FIFO. De esta manera se pueden formar cadenas de 1013 bits los cuales son conducidos hacia el proceso

de codificación BCH tal como establece la norma. Este proceso se refleja en la figura 2.4 que a continuación se muestra.

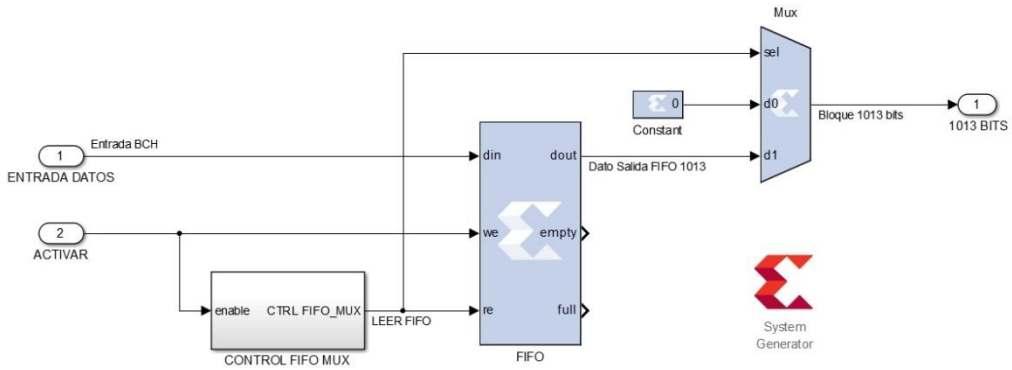

Figura 2.4: Diseño de bloque de 1013 bits.

System Generator no posee un bloque preestablecido de codificación BCH por lo que la proxima etapa se diseñó mediante la utilización de diferentes bloques de XSG.

La cadena de bits proveniente del bloque de 1013 bits es almacenada en una memoria FIFO de donde se extraen los bits teniendo en cuenta que se deben agregar al flujo de datos de salida los bits de chequeo de paridad del proceso de codificación BCH. El bloque Control Code maneja los tiempos de lectura de la memoria y el proceso de codificación. La figura 2.5 a continuacion muestra el diseño realizado.

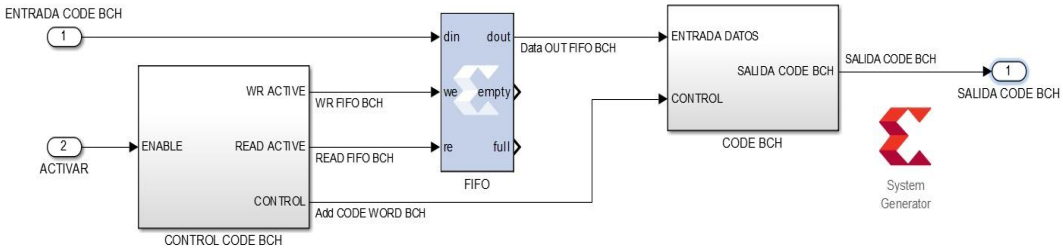

Figura 2.5: Diseño de etapa de codificación BCH con bloques Xilinx.

De esta manera se realiza la codificacion de los 1013 bits de informacion y se agregan 10 bits de chequeo de paridad. El proceso de codificacion propiamente se realiza mediante un circuito que efectua la división de polinomios el cual se muestra en la figura 2.6.

En este proceso, primeramente, la señal **control** permanece en nivel bajo durante las primeras 1013 rotaciones, el multiplexor 1 permanece cerrado a su entrada **d0** permitiendo la transmisión de los bits de mensaje dentro del registro de desplazamiento. El **multiplexor 2** permanece tambien en su posición baja durante las primeras 1013 rotaciones permitiendo la salida de los bits de mensaje. Después de las primeras 1013 rotaciones, la señal **control** pasa a nivel alto los conmutadores 1 y 2 se cierran a su entrada **d1**; de manera que se introducen ceros durante las restantes 10 rotaciones que limpian al **registro** y se extraen los 10 bits de chequeo por la **salida**. El total de rotaciones es igual a 1023.

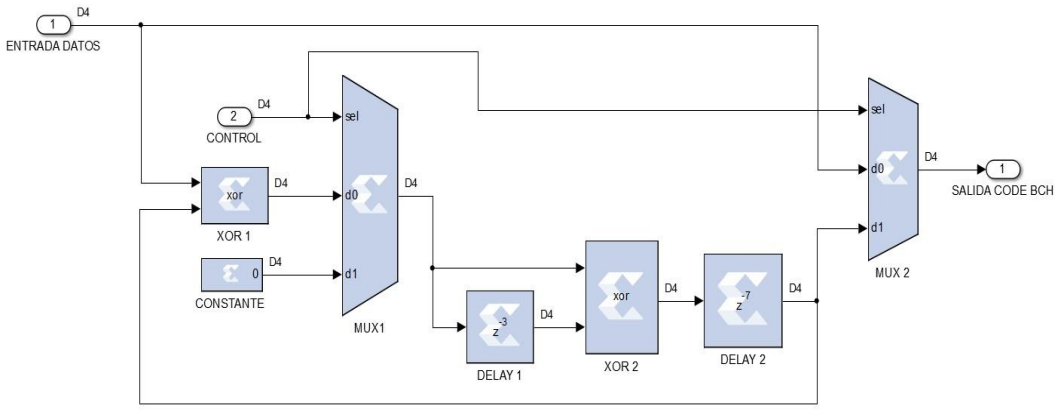

Figura 2.6: Proceso de codificación BCH.

De esta manera se obtiene el diseño de la etapa de codificación BCH donde se generan bloques de 1023 bits, 1013 de información y 10 de chequeo de paridad.

2.3.3 Etapa de codificación BCH mediante Simulink.

La realización de la codificación en Simulink parte de la idea de realizar una comparación y comprobación del diseño realizado mediante bloques funcionales de XSG y el diseño mediante Simulink, el cual se toma como patrón.

Para realizar el diseño mediante Simulink se utiliza un buffer que forma bloques de 752 bits los cuales son conducidos a un bloque **Pad** el cual agrega 261 ceros de manera que 1013 bits pasan al bloque codificador BCH, el cual agrega 10 bits de chequeo.

La interfaz de configuración del bloque **BCH Encoder** se muestra en la figura 2.7, en el campo *Code Word length*, N especifica la longitud de la palabra codificada y en el campo *Message length*, K la cantidad de bits de mensaje. Aunque este bloque brinda la opción de especificar el polinomio generador del código, también puede inferirlo insertando los parámetros N y K, esta opción es la escogida para el diseño.

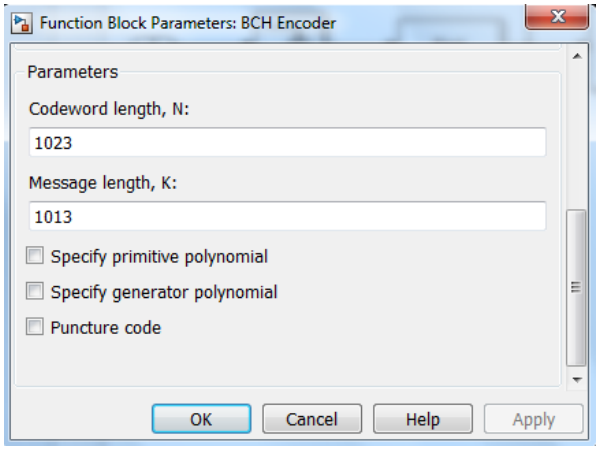

Figura 2.7: Diseño de etapa de codificación BCH en Simulink.

Finalmente se extraen los 261 ceros mediante un bloque **Pad** obteniéndose de esta forma un bloque de 762 bits, 752 de información y 10 de redundancia a la salida del codificador externo tal como establece la norma. La figura 2.8 muestra la estructura del proceso de codificación mediante Simulink.

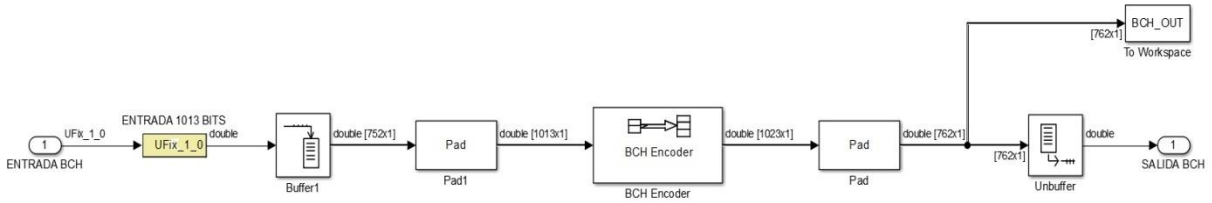

Figura 2.8: Diseño de etapa de codificación BCH en Simulink.

2.3.4 Codificación interna LDPC con bloques de XSG

Luego de la codificación BCH se realiza el proceso de codificación interna de la norma DTMB. Para ello es necesario eliminar los 261 ceros iníciales que se incorporan en la codificación BCH y se agrupan 4, 6 u 8 bloques de 762 bits según la razón de código que se utilice. Estos bits se codifican mediante la matriz generadora del código LDPC de manera que se obtiene un bloque de 7493 bits. En este trabajo se plantea utilizar una razón de código de 0,4 por lo que se obtienen 3048 bits de información y 4445 bits de chequeo de paridad.

Luego de la codificación BCH esta cadena de 1023 bits se guarda en una memoria FIFO de manera que la escritura de la memoria comienza luego de 261 bits con el objetivo de eliminar los ceros iníciales, por lo que se guardan 762 bits en la memoria FIFO. Este proceso se repite con el objetivo de agrupar 4 grupos de 762 bits para un total 3048 bits teniendo en cuenta la razón de código 0.4 que se determina utilizar.

El diseño del codificador interno LDPC utiliza el proceso de codificación de códigos Cuasi-Cíclicos, donde se plantea dividir los bits de información nombrados U en secciones de 127 bits U_0, U_1, U_2…..U_{23} y se realiza la multiplicación de estos vectores con las submatrices de la matriz generadora G_{qc}, se pueden obtener los bits de chequeo V del código como $V = U \times G_{qc}$, siendo necesario implementar un circuito que emule la multiplicación de matrices.[7]

Para obtener 7493 bits tal como establece la norma, la matriz generadora para una razón de código de 0.4 consta de 24 filas y 35 columnas de submatrices cada una de 127 por 127 bits. El bloque de información de 3048 bits guardados se divide en 24 grupos o secciones de 127 bits los cuales se utilizan en el proceso de codificación. Estas secciones coinciden con la cantidad de filas de submatrices de la matriz generadora[6].

La sección de chequeo de paridad es formada con un Registro de Desplazamiento, Sumador, Acumulador Cíclico o CSRAA (del inglés, ***cyclic shift register adder accumulator***) el cual emula la multiplicación de matrices cuyo circuito se muestra en la figura 2.9.

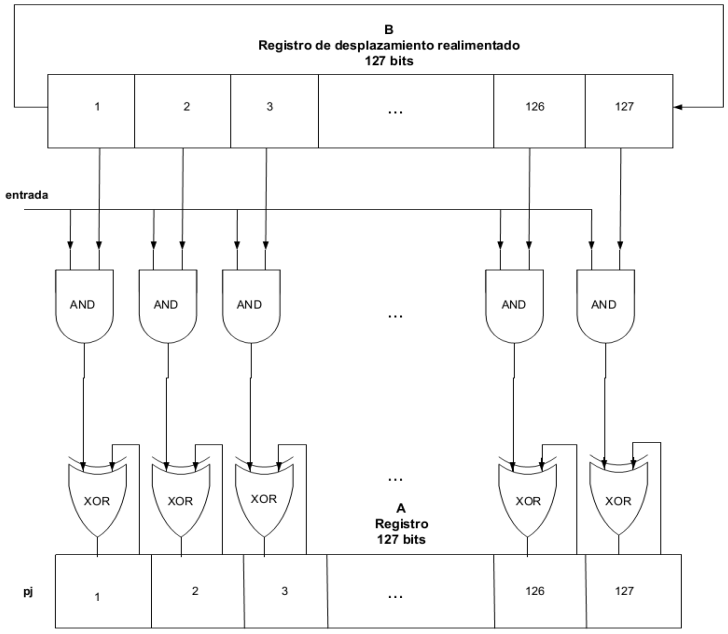

Figura 2.9: Registro de Desplazamiento, Sumador, Acumulador Cíclico[7].

Las secciones u= (u_0, u_1, u_3,…..u_{23}) son rotadas en el codificador y la suma acumulada que constituye el aporte de la sección de los bits de chequeo es S = ($u_0 g_0 + u_1 g_1 +$…….$u_{23} g_{23}$)

En el primer paso el primer elemento de la primera fila de la matriz del generador $g_{0,0}$ es almacenado en el registro realimentado B y el contenido del registro A es reseteado. Cuando el bit de información $u_{0,0}$ es rotado en el codificador, el producto $u_{0,0}g_{0,0}$ es formado a la salida de la compuerta AND y este es sumado al contenido almacenado en el registro A. La realimentación del registro de desplazamiento B es rotada un lugar a la derecha, y el nuevo contenido en B es $g_{0,1}$. Cuando el próximo bit de información $u_{0,1}$ es rotado en el codificador el producto $u_{0,1}g_{0,1}$ es formado a la salida del AND y es sumado al contenido del registro A que es $u_{0,0}$ $g_{0,0}$. La suma $u_{0,1}$ $g_{0,1}$ + $u_{0,0}$ $g_{0,0}$ + es almacenada en el registro A, y el proceso de rotar, sumar y almacenar continúa. Cuando el último bit de información $u_{0,\,127}$ de la sección u_0 es rotado en el codificador, el registro A almacena la suma parcial u_0g_0 la cual es la contribución de los bits de chequeo de paridad de la sección de información u_0[5].

En este tiempo $g_{1,0}$, o sea la primera fila del circulante G_1, es cargada en el registro B y el proceso de rota-suma-almacenamiento se repite. Cuando la sección de información u_1 ha sido completamente rotada en el codificador, el registro A almacena la suma parcial u_0g_0 +$u_1g_1$que es la contribución de los bits de chequeo de paridad de las secciones de información u_1 y u_0. El proceso anterior se repite hasta que la última sección de información u_{23} ha sido completamente rotada en el codificador. En este momento el registro A contiene la sección de chequeo de paridad[5].

Para una razón de código 0.4 es necesario formar 35 secciones de chequeo de paridad, en general el circuito de codificación consiste en 35 circuitos CSRAA. Las secciones chequeo de paridad son formadas al mismo tiempo, en paralelo y entonces serán rotados en la salida de forma serie.

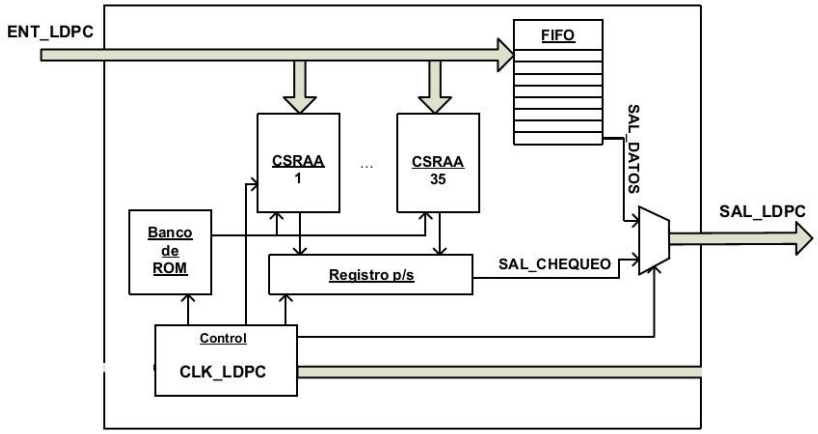

Figura 2.10: Estructura de codificación LDPC[7].

La estructura de la etapa de codificación se realiza mediante una memoria ROM donde se almacena la matriz generadora para esa razón de código. La característica Cuasi-Cíclica de estas matrices reduce de forma significativa la capacidad de almacenamiento necesaria, pues solo es preciso almacenar la primera fila de cada submatriz cíclica. Los datos de entrada son procesados por los CSRAA al mismo tiempo que son almacenados en una memoria FIFO. Es necesario almacenarlos debido a que la estructura de codificación establece los bits de chequeo antecediendo a los bits de información. Este proceso es reflejado esquemáticamente como se muestra en figura 2.10 mostrada anteriormente.

Según el diagrama descrito anteriormente el diseño de la etapa de codificación realizada mediante bloques de XSG se realiza como lo muestra la figura 2.11; el bloque ***Control Sistema*** gestiona y controla todas las señales de control.

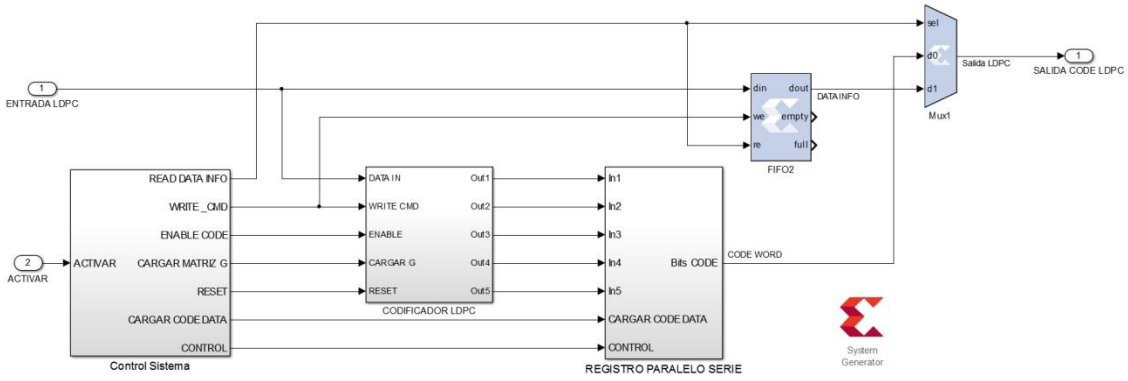

Figura 2.11: Diseño de etapa de codificación LDPC en XSG.

2.3.4.1 Bloque codificador LDPC

El bloque ***CODIFICADOR LDPC*** es dividido en dos etapas: **codificador** y **concatenar**. La etapa **codificador** agrupa 35 secciones de chequeo de paridad o circuitos CSRAA; de manera que se obtienen 35 salidas de 127 bits de palabras de código, como se observa en figura 2.12. En cada circuito CSRAA se introducen los datos que son guardados previamente

en una memoria FIFO, luego de 3048 bits se realiza un reset del sistema. El proceso de escritura en la FIFO se realiza de forma que no se guardan los 261 ceros al inicio de cada bloque de 1023 bits de la codificación BCH, esto hace que se guarden solo bloques de 762 bits que son utilizados en el proceso de codificación.

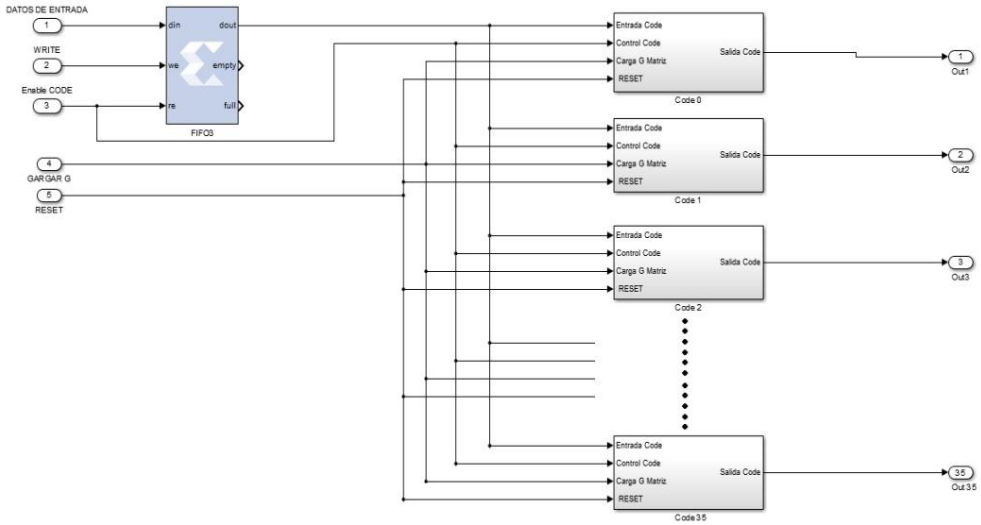

Figura 2.12: Etapa de codificación LDPC

El proceso de codificación de cada circuito CSRAA se muestra en la figura 2.13. En este proceso el bloque *Control ROM,* carga en un registro de desplazamiento, cada 127 ciclos; el valor en 127 bits de la primera fila de la matriz generadora desde una memoria ROM que guarda las primeras filas de cada matriz generadora, así comienza la rotación de la primera fila de la matriz. Por otro lado el bits de información que se lee de la memoria FIFO es conectado a la entrada de control de un multiplexor lo que permite que a la salida hayan 127 bits todos '0' o '1'. Esto permite que se proceda a realizar el proceso de codificación sobre los 127 bits de cada fila de la matriz generadora.

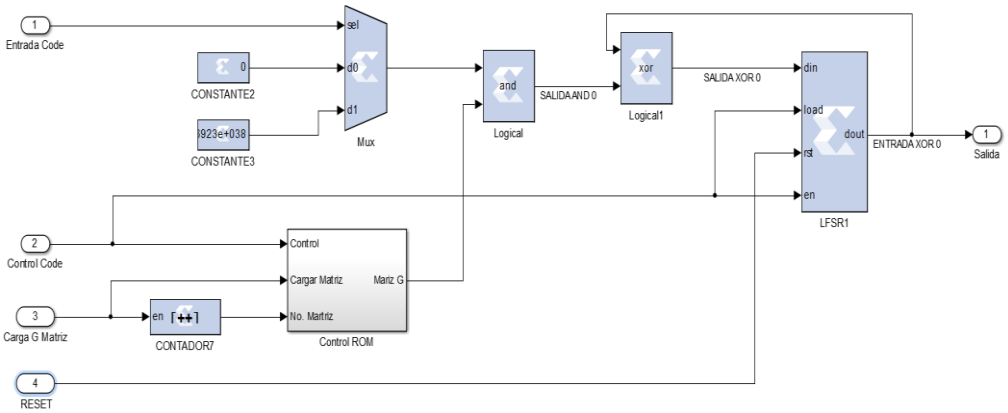

Figura 2.13: Proceso de codificación de circuito CSRAA

En la etapa **concatenar** se agrupan esas salidas en grupos de 7 y se introducen en bloques **concatenar**, disponibles en XSG donde los datos se introducen en forma paralelo y se extraen en forma serie. El bloque **concatenar** está limitado en la cantidad de bits que puede

asumir por lo que es posible agrupar 7 entradas en cada bloque de manera que se obtienen 5 salidas 889 bits de palabras de código para un total de 4445 bits. Este proceso es mostrado en la figura 2.14 que se muestra a continuación.

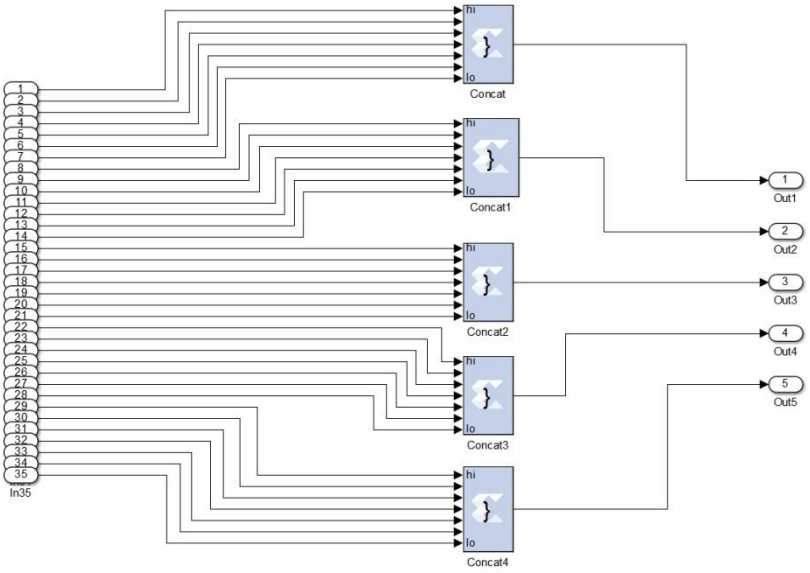

Figura 2.14: Etapa concatenar de bloque LDPC

2.3.4.2 Registro paralelo serie

Este bloque o etapa paralelo serie se utiliza para poder llevar las 5 salidas del proceso de codificación a una sola salida serie de los bits de chequeo de paridad. La figura 2.15 muestra el diseño realizado.

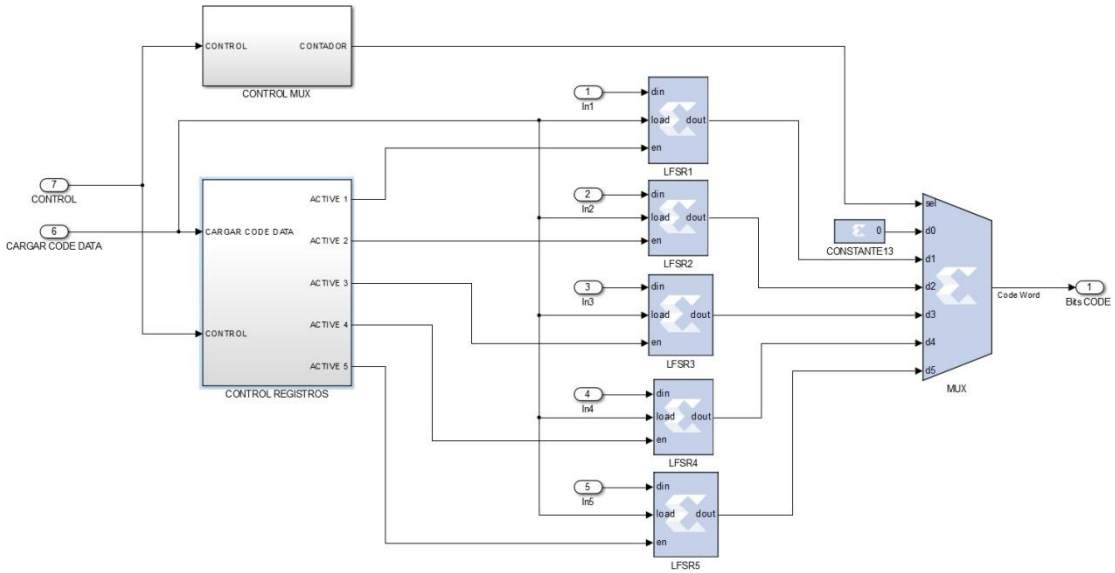

Figura 2.15: Diseño de bloque paralelo serie

En este proceso se cargan de forma paralelo los registros de desplazamiento con los 889 bits que proceden de cada salida del proceso de codificación. Mediante una máquina de estados se realiza la activación de estos registros para obtener los datos de forma serie. Mediante un multiplexor se van sacando los datos que provienen de cada registro de manera que se obtienen 4445 bits de chequeo de paridad.

Finalmente, como puede observarse en la figura 2.10 la salida del bloque paralelo serie es llevada a un multiplexor que extrae los bits de chequeo de paridad y luego los 3048 bits de información que son guardados en una memoria FIFO.

2.3.5 Codificación LDPC en Simulink.

El proceso de codificación LDPC se realiza también a través de Simulink, para ello se utiliza un buffer el cual almacena 3048 bits. Luego este bloque de bits es pasado a un bloque de codificación LDPC que posee Simulink el cual entrega un bloque de 7493 bits al que se les extraen los cinco primeros bits que son de chequeo mediante un bloque ***Pad*** obteniéndose finalmente 7488 bits tal como establece la norma. La figura 2.16 muestra dicho esquema de codificación en Simulink.

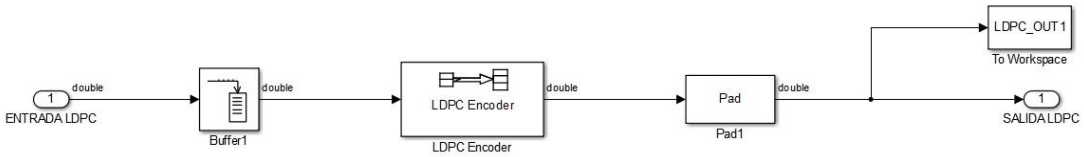

Figura 2.16: Etapa de codificación LDPC en Simulink.

El bloque LDPC Encoder codifica de forma paralelo los bits de entrada. En el campo *Parity-check matrix* se especifica la matriz de chequeo de paridad del código LDPC la cual varía según la razón de código. La norma DTMB define las matrices generadoras por sus submatrices cíclicas, es decir, sólo define la primera fila de cada submatriz, el resto de las filas se obtienen de la rotación de la primera.

A través de Matlab se confecciona una función para la obtención de las matrices de chequeo de paridad desde un fichero de texto que contiene la primera fila de cada submatriz las que se especifican en el estándar según establece [11]. Ver en anexo 1 la función de Matlab utilizada.

De esta forma se logra obtener la matriz de chequeo de paridad según la razón de código 0,4 que se plantea utilizar en este trabajo.

CAPÍTULO 3. Resultados de simulación de las etapas diseñadas

En este capítulo, mediante Simulink; se realiza la simulación de cada una de las etapas diseñadas por separado y se realiza un análisis de las cadenas de bits obtenidas como resultado del sistema. Finalmente se realiza una simulación de todo el diseño y se verifica el correcto funcionamiento del mismo mediante una comparación de los resultados que se obtienen mediante los bloques de codificación propios de Simulink y los resultados que se obtienen mediante el diseño de las etapas de codificación mediante bloques funcionales de XSG que pueden ser implementados y llevados a hardware.

3.1 Simulación y resultados de las etapas de codificación

Para realizar una simulación del sistema se utiliza un bloque Generador de Bernoulli disponible en Simulink el cual genera una cadena de bits que simula el flujo mpeg. Además se utiliza una entrada que activa el sistema. En la figura 3.1 a continuación se muestra este esquema de diseño.

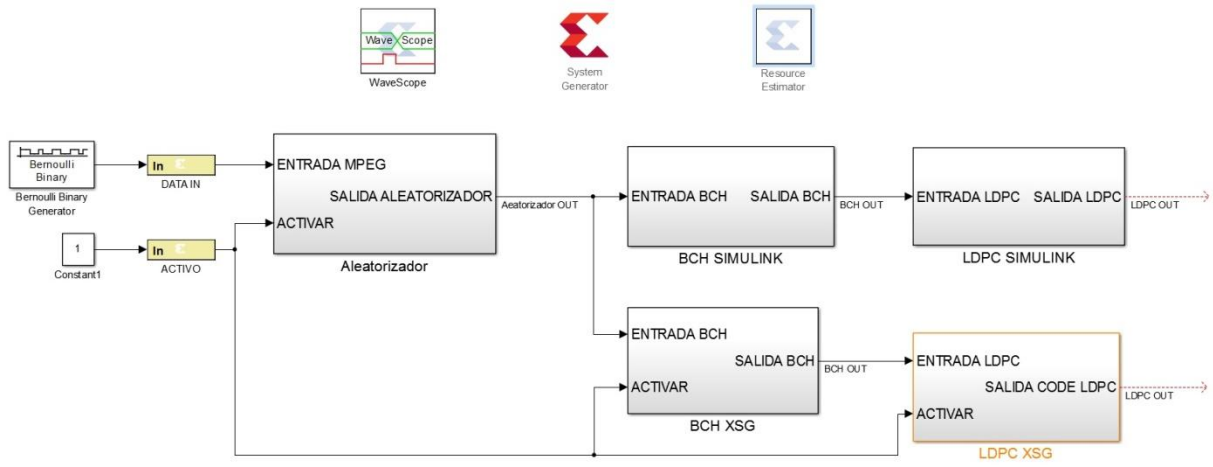

Figura 3.1: Sistema diseñado en Simulink.

En lo adelante se realiza un análisis de las cadenas de bits obtenidas a la salida de cada bloque que permite verificar que los resultados obtenidos están en correspondencia con el funcionamiento de la norma DTMB

3.1.1 Simulación y resultados de bloque aleatorizador.

En las figuras 3.2 y 3.3 se muestran las cadenas de bits de entrada, la secuencia pseudoaleatoria, la salida del aleatorizador así como la señal de reset de los registros de desplazamiento realimentados la cual se comprueba activa cada 6016 bits. Realizando un

análisis de la cadena de bits de la secuencia pseudoaleatoria podemos observar que la misma cumple con lo establecido en la norma, ver figura 1.8.

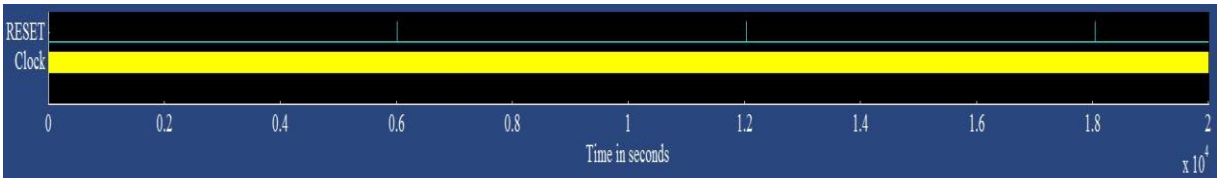

Figura 3.2: Señal de reset de aleatorizador

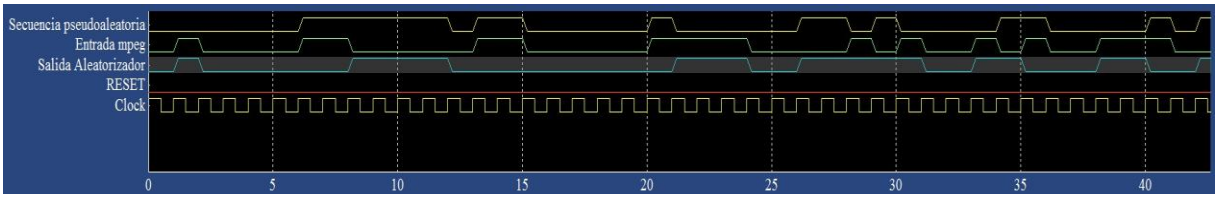

Figura 3.3: Secuencia pseudoaleatoria y salida del aleatorizador

3.1.2 Simulación y resultados de la etapa de codificación BCH.

Los resultados de la etapa de codificación como se detalla en el capítulo anterior se dividen en una etapa que conforma los bloques de 1013 bits que pasan a ser codificados; por lo que los resultados se muestran de la misma forma.

En la figura 3.4 se puede apreciar la señal de lectura de la FIFO, así como la cadena de 752 bits que se extraen luego de 261 ciclos en los que se agregan ceros de manera que se conforman los bloques de 1013 bits que pasan a ser codificados.

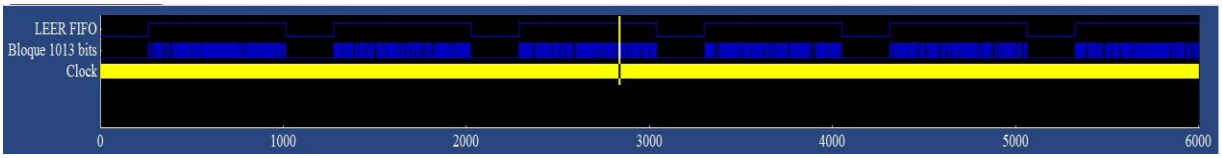

Figura 3.4: Señal de lectura de FIFO

La próxima figura 3.5 muestra con detalle que luego de 261 ceros comienza la lectura que termina al completar los 752 bits de datos y completar el bloque de 1013 bits.

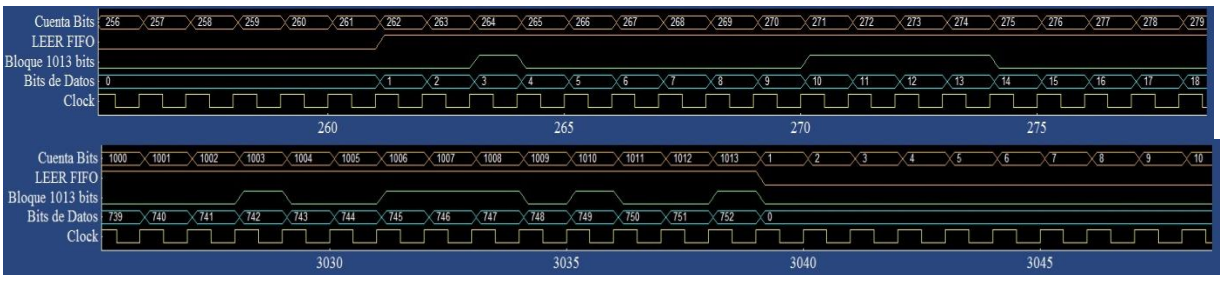

Figura 3.5: Bloque de 1013 bits

Luego este bloque de bits pasa a ser codificado de forma serie, de manera que se obtienen luego de 1013 ciclos los 10 bits de chequeo de paridad, obteniéndose un total de 1023 bits. Se puede observar además la señal que lee los datos de información así como la señal que

adiciona a la cadena de datos los bits de chequeo de paridad. La figura 3.6 muestra esta información.

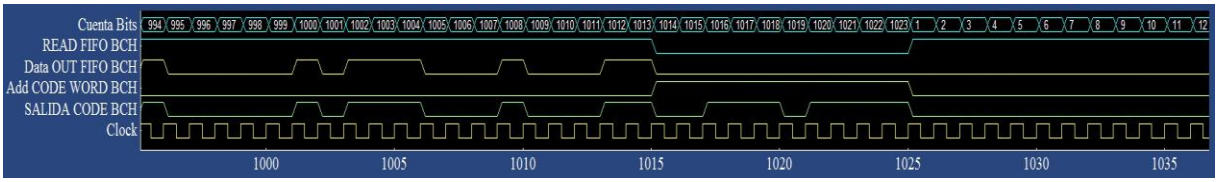

Figura 3.6: Salida de datos de bloque de codificación BCH

Se verifica que los bits de chequeo de paridad del primer bloque de bits son '0011101111'. A la vez que se realiza la simulación del sistema diseñado mediante bloques propios de Simulink de donde se obtienen los resultados que a continuación se muestran en la figura 3.7.

Edit data for Out_BCH

a:1006	Data:1007	Data:1008	Data:1009	Data:1010	Data:1011	Data:1012	Data:1013	Data:1014	Data:1015	Data:1016	Data:1017	Data:1018	Data:1019	Data:1020	Data:1021	Data:1022	Data:1023
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	1	0	0	1	1	1	0	1	1	1	1
1	0	0	1	1	1	0	0	1	0	0	1	0	0	1	1	1	0
1	1	1	0	1	0	0	1	0	1	0	0	0	1	1	1	1	0
0	0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	1	0
0	0	1	0	0	0	1	1	1	1	1	0	1	0	1	0	1	0

Figura 3.7: Datos de codificación BCH en Simulink.

De esta manera se comprueba mediante la comparación de las cadenas de bits que se obtienen que los 10 bits de chequeo de paridad son iguales para ambos diseños. Esto demuestra que el diseño de la etapa de codificación BCH funciona de manera correcta.

3.1.3 Simulación y resultados de la etapa de codificación LDPC.

En lo adelante se muestran los resultados de las cadenas de bits y señales de control de la etapa de codificación LDPC.

Inicialmente la próxima figura 3.8 muestra la cadena de datos de entrada LDPC, la señal de escritura de las memorias FIFO luego de 621 ciclos y activa durante 762 ciclos; esto permite guardar los bloques de 762 datos de información que serán codificados. Además se muestra la señal que permite la lectura de los datos a codificar y activa el proceso de codificación así como la cadena de 3048 bits de información a codificar.

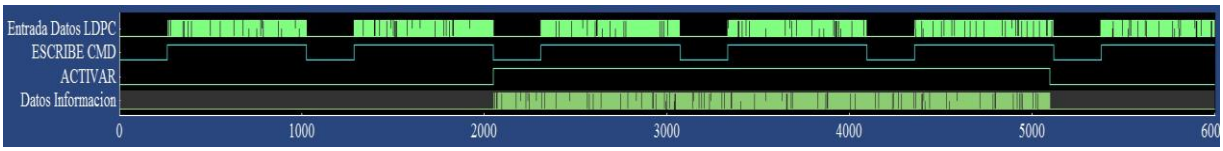

Figura 3.8: Señal de escritura, activación y datos de información.

La siguiente figura muestra la señal *Cargar Matriz* que ordena cargar la primera fila de cada matriz generadora que es especificada en la norma. Esta señal se repite cada 127 ciclos o bits como se observa en figura 3.9.

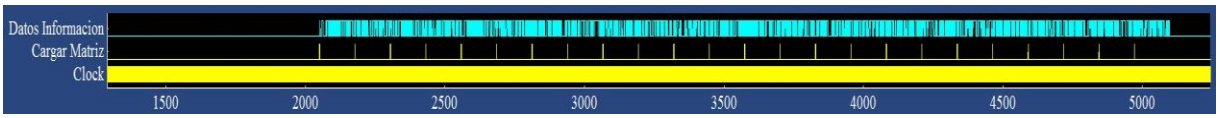

Figura 3.9: Señal Cargar Matriz.

Según el proceso de codificación a través de los circuitos CSRAA que se observa en la figura 2.13 se obtienen los resultados que a continuación se detallan en la siguiente figura 3.10.

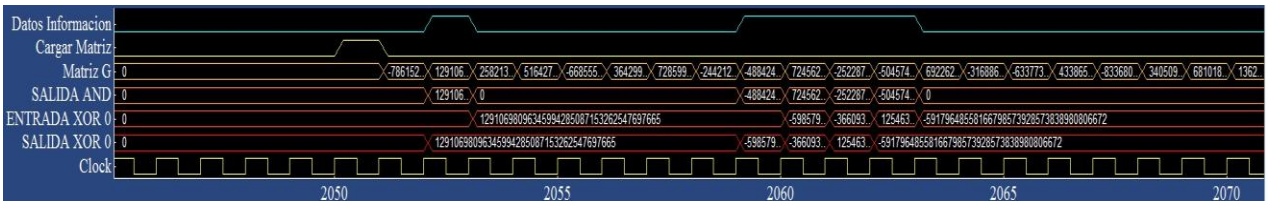

Figura 3.10: Proceso de codificación LDPC

Así, una vez que se realiza el proceso de codificación a los 3048 bits, se obtiene a la salida de cada circuito CSRAA una salida de 127 bits. Estas 35 salidas son llevadas al bloque *CONCATENADOR* donde se agrupan en grupos de 7 y se obtienen cinco salidas de 889 bits.

La siguiente figura 3.11 muestra datos y señales que se obtienen como resultado del bloque *Registro Paralelo Serie*. En la misma puede observarse la señal que ordena cargar los registros de desplazamiento así como las 5 salidas de datos o cadenas de 889 bits que son cargados y que luego se extraen de forma serie.

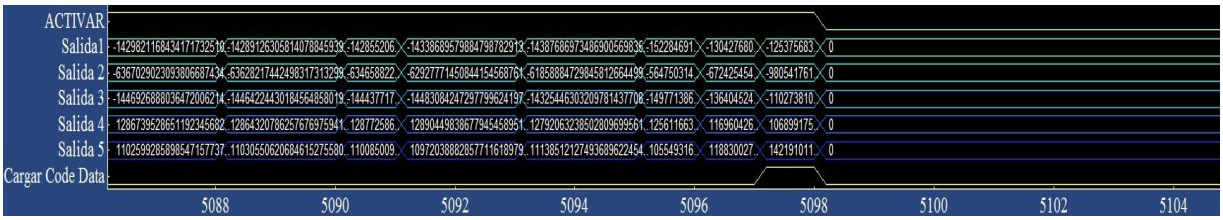

Figura 3.11: Señal cargar palabras de código y salidas de 889 bits

Una vez cargados los registros de desplazamiento se procede a extraer los datos en forma serie. Para esto se procede a activar cada uno de estos 5 registros de desplazamiento uno tras el otro. Se puede verificar que en el momento que se activa la señal que ordena la carga de los registros también se activan todos los registros; esto se configura de esta manera debido a que los registros deben activarse para cargar datos en ellos. Esto se refleja en la figura 3.12 que se muestra.

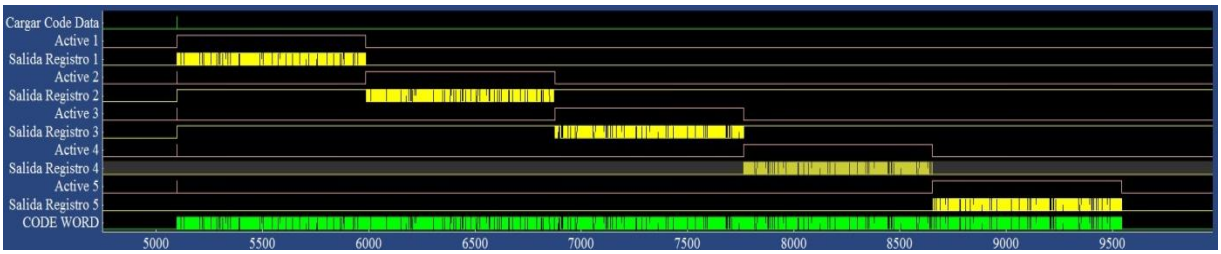

Figura 3.12: Palabra de código de 4445 bits.

Una vez que se obtienen los 4445 bits de chequeo de paridad LDPC se procede a extraerlos y luego los 3048 bits de información obteniéndose 7493 bits. Esto de detalla en la siguiente figura 3.13. De esta manera se realiza el proceso de codificación LDPC.

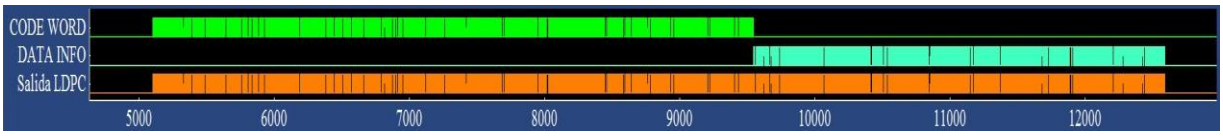

Figura 3.13: Bits de salida de proceso de codificación LDPC

Por otra parte se realiza la simulación del sistema LDPC diseñado mediante bloques propios de Simulink. Estos datos se visualizan en Matlab, parte de los mismos se muestran en la figura 3.14 y se realiza una comparación de los datos de chequeo de paridad que se obtienen mediante las dos vías.

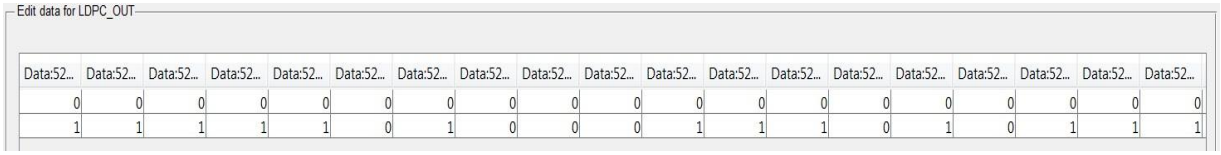

Figura 3.14: Bits de salida de codificación LDPC en Simulink

De esta manera se verifica que el funcionamiento y los bits de la palabra de código que se obtiene mediante el diseño de la codificación LDPC en XSG son correctos verificándose que el diseño realizado cumple con el funcionamiento de la norma DTMB.

3.1.4 Estimación de recursos

Una vez que se ha comprobado el correcto funcionamiento del diseño del codificador para la norma DTMB, se procede a realizar una estimación de los recursos que se necesitan para implementar el diseño en un kit de desarrollo basado en un FPGA. Esto se logra mediante el bloque un bloque disponible en XSG llamado *Estimador de recursos.*

En la siguiente figura 3.15 se muestran los resultados que se obtienen de la estimación de recursos, estos datos son necesarios para seleccionar que tarjeta de desarrollo o FPGA se necesita para poder implementar el diseño.

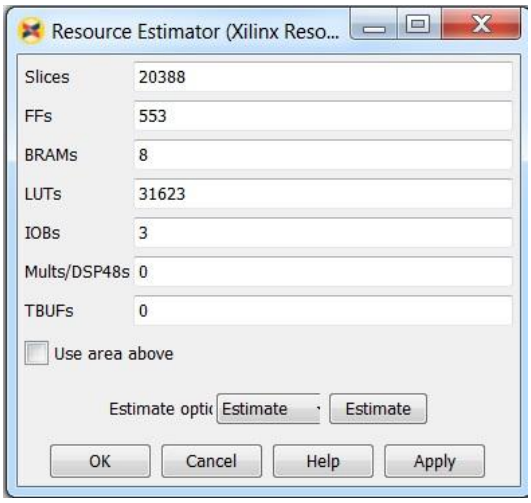

Figura 3.15: Recursos necesarios para implementar el diseño.

El kit de desarrollo Nexys 3 que se dispone posee una FPGA de Xilinx Spartan 6 XC6SLX45 que según se puede consultar en hoja de datos de esta familia [28], la misma no posee los recursos suficientes para poder implementar en ella este diseño por lo que no es posible en este momento realizar la simulación de hardware e implementación del diseño realizado.

Se considera que una vez que se tenga una tarjeta o kit de desarrollo que tenga los recursos suficientes para implementar este diseño y se realice el proceso de simulación de hardware donde se ejecutan los procesos directamente en el FPGA haya que ir atrás en el diseño y realizar algunos ajustes teniendo en cuenta los tiempos y demoras en el hardware. Hasta este punto mediante este trabajo, se tiene el diseño de un codificador de la norma DTMB que funcionalmente satisface los requerimientos de la norma y que es posible implementarlo en hardware basado en FPGA de Xilinx.

CONCLUSIONES

- Para realizar el diseño de sistemas digitales complejos basados en FPGA una manera eficiente de abordar el diseño es mediante la herramienta de software Xilinx System Generator.
- Mediante la herramienta de software Xilinx System Generator es posible realizar el diseño de todas las etapas de un codificador para la norma de televisión digital DTMB.
- Se valida el correcto desempeño del sistema diseñado respecto al funcionamiento de la norma DTMB mediante simulación funcional del diseño realizado mediante bloques de Xilinx System Generator, realizándose una comparación con los resultados que se obtienen mediante el diseño del sistema en Simulink.
- El sistema de codificación de la norma DTMB realizado puede ser implementado por completo en hardware FPGA.
- Se logra la implementación en hardware del proceso de codificación LDPC, en la bibliografía consultada se aborda cómo funciona el proceso de codificación pero no como realizar el diseño digital y la implementación de hardware del mismo.
- Este trabajo proporciona los primeros pasos en función de lograr realizar un diseño completo de un modulador de televisión digital para la norma DTMB, lo cual contribuya a lograr una apropiación de tecnología e independencia tecnológica.

RECOMENDACIONES

- Realizar el proceso de simulación de hardware mediante el empleo de cable JTAG y un kit de desarrollo que cuente con los recursos necesarios para implementar el diseño.
- Continuar el diseño del resto de las etapas que conforman un modulador de TDT para la norma DTMB.

BIBLIOGRAFÍA

[1] A. Sierra Romero, "DISEÑO DE LA CODIFICACIÓN DE CANAL Y MAPEO DE UN MODULADOR DVB-T," ed. La Habana: 1er Foro Internacional de Television Digital, 2013.

[2] G. E. Reyes León, "PLAN DE ACCIÓN DEL PROCESO DE DIGITALIZACIÓN DE LA TELEVISIÓN TERRESTRE EN CUBA," ed. La Habana: Ministerio de Comunicaciones, 2014.

[3] Y. del Sol González. (2017, 28 de junio). LACETEL Desde la ciencia, aportar también al desarrollo de las telecomunicaciones. *Granma*. Available: http://www.granma.cu/cuba/2017-06-28/desde-la-ciencia-aportar-tambien-al-desarrollo-de-las-telecomunicaciones-28-06-2017-21-06-18

[4] N. García Rodríguez and A. Rey Domínguez, "Diseño FPGA de un modulador DTMB para canalización de 6MHz " *RIELAC,* vol. XXXIII 2/2012, pp. 17-28, 2012.

[5] A. Rey Domínguez and L. G. Raymond Rodríguez, "Diseño del Aleatorizador, la Codificación de Canal, Mapeo y Entrelazado de un modulador DTMB.," Master, FACULTAD DE ELÉCTRICA., INSTITUTO SUPERIOR POLITÉCNICO JOSÉ A. ECHEVERRÍA, La Habana, 2011.

[6] A. Rey Domínguez, "DISEÑO DEL ALEATORIZADOR, LA CODIFICACIÓN DE CANAL, MAPEO Y ENTRELAZADO DE UN MODULADOR DTMB," L. G. R. Rodríguez, Ed., ed. La Habana: 1er Foro Internacional de Television Digital, 2013.

[7] L. G. Raymond Rodríguez and A. Rey Domínguez, "PROPUESTA DE DISEÑO DE LA CODIFICACIÓN DE CANAL DE LA NORMA DE TELEVISIÓN DIGITAL DTMB," ed. La Habana: 1er Foro Internacional de Television Digital, 2013.

[8] UIT, "Informe UIT-R BT.2035-2: Directrices y técnicas para la evaluación de sistemas de radiodifusión de Televisión Digital Terrestre incluida la determinación de sus zonas de cobertura.," in *Serie BT*, ed. Ginebra, 2009.

[9] J. F. Castillo León, "Decodificador Esférico de Complejidad Reducida para Sistemas de Comunicación MIMO," Postgrado en Ingeniería Electrónica, Facultad de Ciencias, Universidad Autónoma de San Luis Potosí, 2012.

[10] A. Martínez Alonso and R. Martínez Alonso, "EVALUACIÓN COMPARATIVA DE ESTÁNDARES DE RADIODIFUSIÓN DE TELEVISIÓN DIGITAL TERRESTRE," R. M. Alonso, Ed., ed. La Habana: Decimotercera semana tecnológica, 2013.

[11] National Special Working Group for Chinese Digital Television Terrestrial Broadcasting Standard, "Framing Structure, Channel Coding and Modulation

for Digital Television Terrestrial Broadcasting System (DTMB)," ed. China, 2006.

[12] Y. Orta Rivera, "La nueva TV," *Suplemento especial JUVENTUD REBELDE,* 2013.

[13] D. Vega Muguercia*, et al.* (2013, 20 febrero 2017). Television Digital en Cuba, el cambio que implica una nueva tecnologia. Available: http://www.cubadebate.cu/especiales/2013/07/10/dialogo-esclarecedor-sobre-la-tv-digital-en-cuba/#.WMwe301-7r8

[14] C. Zhang*, et al.*, "The Technical Analysis on the China National Standard for Digital Terrestrial TV Broadcasting," ed. Beijing, China: Beihang University.

[15] L. G. Raymond Rodríguez, "DESARROLLO DE SOLUCIONES INFORMÁTICAS PARA LA PLANIFICACIÓN DE LA TELEVISIÓN DIGITAL TERRESTRE Y OTROS SERVICIOS EN CUBA ", ed. La Habana: Decimotercera Edición Semana Tecnologica, 2013.

[16] W. E. RYAN, "An Introduction to LPDC Codes," ed. University of Arizona: Deparment of Electrical and Computer Engineering, 2003.

[17] M. Pregara, "Low Density Parity Check Code Implementation ", Z. Saigh, Ed., ed. Bradley University: Department of Electrical and Computer Engineering 2013.

[18] W. E. RYAN, "Channel Codes Classical and Modern," S. LIN, Ed., ed. New York: Cambridge University Press, 2009.

[19] R. Schweers Joachim, "Descripción en VHDL de arquitecturas para implementar el algoritmo CORDIC," ed. Buenos Aires: Facultad de Informatica, Universidad Nacional de La Plata, 2002, p. 143.

[20] F. Pardo Carpio and J. a. Boluda Grau. (2011, 20 marzo 2017). *VHDL. Lenguaje para sintesis y modelado de circuitos (3ra ed.)*. Available: http://www.ra-ma.es/libros/VHDL-LENGUAJE-PARA-SINTESIS-Y-MODELADO-DE-CIRCUITOS-3-EDICION-ACTUALIZADA/27946/978-84-9964-040-2

[21] R. O. Arias Zamora, "Metodología para el diseño de aplicaciones de media complejidad en FPGAs de Xilinx," Facultad de Electrica, Universidad Central de Las Villas, Santa Clara, 2010.

[22] International Electrotechnical Commission, "Electronic Design Interchange Format (EDIF) Part 2 Version 4.0.0," ed, 2000.

[23] S. T. Perez*, et al.*, "Metodologías de diseño para dispositivos digitales programables," presented at the I Jornadas Iberoamericanas de Innovacion Educativa en el ambito de las TIC, Las Palmas de Gran Canaria, 2014.

[24] G. Guichal, "Diseño Digital Utilizando Logica Programable," ed: Universidad Tecnologica Nacional. Facultad Regional Bahia Blanca, 2005.

[25] M. Rodriguez Valido and M. Gutiérrez Castañeda, "Metodología de diseño en FPGA usando Xilinx System Generator," E. M. Castello*, et al.*, Eds., ed. Tenerife, España: Universidad de La Laguna, 2012.

[26] T. Flaih Hasan, "FPGA Design Flow for SDR Transceiver using System Generator," in *International Journal of Engineering and Advanced Technology* vol. 3, ed: Blue Eyes Intelligence Engineering & Sciences Publication Pvt. Ltd, 2014.

[27] E. Cabrera Abizaid and J. Rodríguez Rodríguez, "DESARROLLO DE MÓDULOS PARA UN RE-MULTIPLEXOR DE TRAMAS DE TRANSPORTE MPEG-2 SOBRE FPGA.," presented at the 1er Foro Internacional de Televisión Digital, LACETEL, La Habana, 2013.

[28] Xilinx Inc, "SPARTAN 6 Family Overview," vol. 2.0, ed, 2011.

ANEXOS

- Anexo 1

```
function [H]= mgeneric(generador,razon)
switchrazon
case 1
alfa = 24;
beta = 35;
case 2
alfa = 36;
beta = 23;
case 3
alfa = 48;
beta = 11;
otherwise
disp('Entrada Incorrecta')
end
fdimen = alfa*127;
cdimen = beta*127;
M = 1;
N = 1;
f = zeros(1,alfa);
f(1,[1:alfa])=127;
c = zeros(1,beta);
c(1,[1:beta])=127;
G = zeros(fdimen,cdimen);
G = mat2cell(G, f, c);
for k = 1:length(generador)
fila = generador(k);
fila = cell2mat(fila);
bin = zeros(1,128);
fori = 1:length(fila)
decimal = base2dec(fila(i),16);
hex = dec2base(decimal,2,4);
bin(1,[((4*i)-3):(4*i)])=hex;
end
bin = bin-48;
binario = bin(1,[2:128]);
m = zeros(127,127);
m(1,[1:127])=binario;
for j=2:127;
m(j,[1:127])=circshift(m((j-1),[1:127]),[0 1]);
end
m = [29];
G(M,N)=m;
N = N+1;
if N == (beta+1)
M = M+1;
N = 1;
end
```

```
end
G = cell2mat(G);
I = eye(fdimen);
MG = [G,I];
H = gen2par(MG);
H = sparse(H);
end
```