Universidad de la República
Facultad de Ingeniería

Implementación de un Transmisor de ISDB-T Abierto Bajo el Paradigma de Radio Definida por Software

Memoria de proyecto presentada a la Facultad de Ingeniería de la Universidad de la República por

Javier Hernández, Santiago Castro

en cumplimiento parcial de los requerimientos para la obtención del título de Ingeniero Electricista.

Tutor

Dr. Federico La Rocca ..... Universidad de la República
M.Sc. Pablo Flores Guridi ..... Universidad de la República

Tribunal

Prof. Ing. Alicia Fernández ..... Universidad de la República
Dr. Víctor González-Barbone ..... Universidad de la República
Dr. Rafael Sotelo ..... Universidad de la República

Montevideo
lunes 10 diciembre, 2018

Implementación de un Transmisor de ISDB-T Abierto Bajo el Paradigma de Radio Definida por Software, Javier Hernández, Santiago Castro.

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.1).
Contiene un total de 80 páginas.
Compilada el lunes 10 diciembre, 2018.
http://iie.fing.edu.uy/

No es irrazonable que tengamos o que tropecemos con problemas. Pero hay decenas de miles de años en el futuro. Es responsabilidad nuestra hacer lo que podamos, aprender lo que podamos, mejorar las soluciones, y transmitirlas a nuestros sucesores. Es responsabilidad nuestra dejar las manos libres a las generaciones futuras.

Richard Phillips Feynman (1918-1988)

Esta página ha sido intencionalmente dejada en blanco.

Agradecimientos

A nuestras familias, que siempre pusieron todo lo que tuvieron a su alcance, para que podamos llegar hasta acá. A los compañeros de estudio con los que compartimos tardes interminables preparando exámenes, que terminaron forjando amistades tan profundas como inesperadas. A las barras de amigos que siempre estuvieron ahí cuando los resultados no acompañaron y necesitábamos un empujón anímico. A todos los docentes que se quedaron después de hora para contestar una duda. A los tutores que nos presentaron un desafío que nos puso a prueba en muchas mas ocasiones de las que hubiésemos imaginado antes de comenzar.

A Matilde, que estuvo desde el principio de todo, y en quien encontré siempre la palabra justa y la calma necesaria, para superar las adversidades.

Esta página ha sido intencionalmente dejada en blanco.

A Mati, a mis padres Hugo y Carolina, a mis hermanos Adrian y Cecilia, y a toda la barra del 1030.
A mi abuela, mis padres Cristina y Alfredo, y mi hermano Nacho.

Esta página ha sido intencionalmente dejada en blanco.

Resumen

La radio definida por software es un paradigma donde los sistemas de radiocomunicaciones, tradicionalmente implementados en hardware, son en cambio implementados por software. Esto típicamente se lleva a cabo a través de una computadora personal encargada de ejecutar el software, y un hardware genérico que se encarga de sintonizar, filtrar y muestrear la señal de radio a cierta tasa. De esta manera la señal puede ser procesada por la computadora. Hay una gran variedad de estos equipos que van desde unos pocos dólares hasta algunos miles de dólares. El paradigma de la radio definida por software permite lograr una implementación completa de sistemas de radiocomunicaciones de manera económica y sencilla en cierto sentido.

GNU Radio es un toolkit abierto y libre, ampliamente utilizado para la implementación de radios definidas por software. Provee bloques de procesamiento de señal de propósito general como filtros, re-samplers, moduladores y demoduladores analógicos y digitales. Estos bloques pueden ser combinados para lograr diferentes sistemas de comunicación. También es posible la implementación de nuevos bloques personalizados por parte de los usuarios, de manera de contribuir con el código fuente de GNU Radio. Esto hace que la comunidad de GNU Radio sea muy activa y resulte en una contribución continua de los usuarios tanto en el desarrollo de nuevas radios definidas por software o el testeo de otras implementaciones.

Integrated Services for Digital Broadcasting-Terrestrial (ISDB-T), es el estándar japonés para televisión digital terrestre que ha sido adoptado por la mayoría de los países en América del Sur. En Uruguay mediante un decreto de Presidencia, se adoptó ISDB-T como la norma nacional para la transmisión de televisión digital. Esto implicó la necesidad de desarrollar profesionales locales con un profundo dominio técnico sobre la norma, dando lugar también a importantes trabajos de investigación. Uno de esos trabajos es gr-isdbt, un receptor abierto de ISDB-T basado en radios definidas por software.

En este trabajo presentamos el primer transmisor de ISDB-T abierto, basado en radios definidas por software e implementado sobre GNU Radio. Se logró complementar el trabajo realizado en el proyecto gr-isdbt, para lograr una implementacion de un sistema de transmisión de televisión digital de punta a punta. La implementación lograda en gr-isdbt-tx [1] es capaz de modular por completo una señal ISDB-T fullseg, lo que significa que es capaz de multiplexar en frecuencia hasta tres flujos de transporte MPEG con múltiples programas de audio y video al mismo tiempo. Cada flujo de transporte es codificado de manera independiente logrando diferentes grados de robustecimiento de la señal, formando lo que se

denomina capa. Esto se consigue utilizando modulación OFDM y dividiendo el espectro en 13 grupos de portadoras idénticos, denominados segmentos. Cada capa cuenta con su propia codificación de canal y modulación, que es independiente del resto de las capas.

Una vez que la señal ISDB-T es conformada, se transmite por el aire y puede ser demodulada por cualquier dispositivo compatible con la norma, por ejemplo un TV comercial, un set top box ó una PC con gr-isdbt. Esto convierte a una PC, junto con un hardware SDR con posibilidad de transmitir, en un relativamente económico transmisor de TV digital flexible y totalmente configurable. De hecho quienes posean las licencias adecuadas podrían tener una estación transmisora de TV de manera relativamente sencilla. La aplicación resulta particularmente interesante para la televisión comunitaria en donde el aspecto económico es quizás la restricción más importante.

De manera alternativa, es posible simular todo el sistema concatenado transmisor-receptor dentro de GNU Radio. Esto genera una variada gama de posibilidades de su uso, por ejemplo puede ser utilizado con propósitos educativos mostrando de manera práctica cómo funciona un sistema de comunicación complejo. También podría ser utilizado para evaluar el desempeño de la codificación de las distintas capas jerárquicas al atravesar el canal, o evaluar el impacto de señales interferentes que podrían aparecer en el canal.

Daremos una descripción paso a paso de todos los puntos que hubo que resolver para lograr este transmisor, centrándonos en los aspectos claves que son necesarios dominar para conseguir una implementación exitosa. También describiremos las pruebas llevadas a cabo así como sus resultados.

Tabla de contenidos

Tabla de contenidos

Capítulo 1

Introducción

En Mayo de 2001, la ARIB (Association of Radio Industries and Businesses) presentó la primera versión de su estándar para la transmisión de televisión digital [2]. Coloquialmente denominada "Norma Japonesa de Televisión Digital", académicamente ISDB-T, por sus siglas en inglés provenientes de *Integrated Services Digital Broadcasting, Terrestrial*. La norma sintetiza un conjunto de requerimientos técnicos para la utilización eficiente del espectro radioeléctrico para la transmisión de datos multimedia, con la colaboración y el aval de todos los actores de la industria, como proveedores de tecnología, compañías de broadcasting, investigadores del área, entre otros.

Esta norma basa muchos de sus conceptos en la norma DVB-T, publicada por primera vez en el año 1997 por la organización europea DVB (Digital Video Broadcasting). La posterioridad de ISDB-T con respecto a ésta permitió que se robustecieran algunos de los aspectos más criticados de la norma europea, resultando en un estándar mas robusto para la transmisión. Discutiremos algunos de estos aspectos más adelante, cuando se presente la norma en el capítulo 3.

Actualmente, existen en el mundo cuatro grandes estándares comerciales. Además de ISDB-T y DVB-T, están la norma china DTMB (Digital Terrestrial Multimedia Broadcast) y la norma norteamericana de la ATSC (Advanced Television Systems Comitee). La elección de qué norma adoptar por parte de los gobiernos nacionales radica principalmente en decisiones políticas, cuyo análisis y discusión escapan a los objetivos de este documento.

En Uruguay, al igual que en gran parte de Latinoamérica, se adoptó en 2011 una versión de ISDB-T denominada ISDB-T International, la cual es a grandes rasgos idéntica a la primera, salvo por algunos cambios menores como el cambio en la codificación de fuente (pasa del estándar MPEG-2 a MPEG-4) y la elección de otro estándar de interactividad (se cambia de BML a Ginga).

Luego de la adopción del estándar, en el decreto 585/12 del Ministerio de Industria Energía y Minería (MIEM) se fijó el 21 de Noviembre del año 2015 como la fecha limite para el denominado "apagón analógico" [3], fecha en la cual se dejaría de transmitir televisión por vías analógicas, pasando exclusivamente a medios digitales, liberando los espectros asignados para los canales de TV abierta para otros fines.

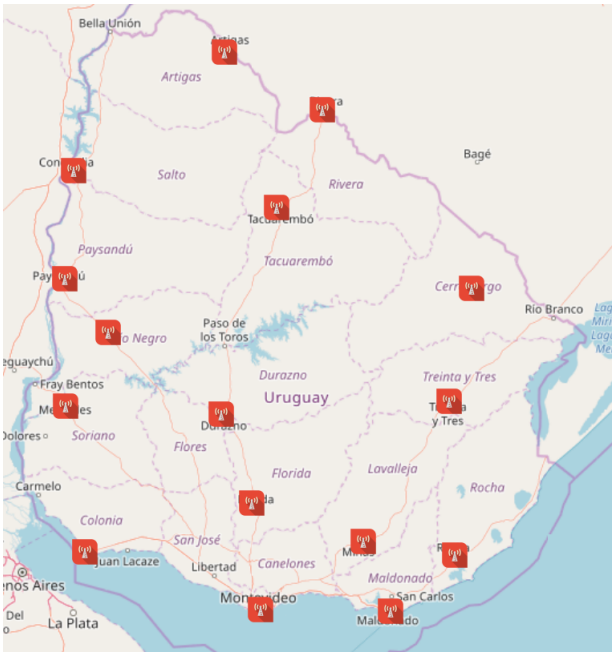

Figura 1.1: Distribución de las estaciones transmisoras de TV digital en el Uruguay.

Durante la implementación del marco legal de la nueva norma de televisión digital se entregaron 22 licencias para transmisión de contenidos bajo la norma ISDB-T International (en adelante ISDB-T por simplicidad). Al día de hoy, tres años después de la fecha límite para el apagón, en Montevideo todos los canales comerciales se encuentran transmitiendo con señal digital y todas las capitales departamentales cuentan con al menos un canal transmitiendo su señal digital. La asignación de bandas de frecuencia para los proveedores del servicio de televisión abierta, quedo trunco luego de disputas comerciales, en un fallo del Tribunal de lo Contencioso Administrativo (TCA), que dejó sin efecto el decreto 585/12 [4]. En esa sentencia, queda también sin efecto la aplicación del apagón analógico, lo que probablemente colaboro con la desaceleración de la transformación tecnológica por parte de los canales comerciales.

Relevamos la situación de las estaciones de transmisión de televisión digital a lo largo del país, mediante una consulta que elevamos a la Dirección Nacional de Telecomunicaciones (DINATEL). En la figura 1.1 presentamos un mapa con las estaciones activas al día de hoy.

La situación de los consumidores del servicio dista de la ideal, pues la televisión analógica sigue siendo la mayor puerta de acceso al medio. La Encuesta Continua de Hogares del Instituto Nacional de Estadística [5], cuyos indicadores son una muestra representativa de la situación de todos los hogares del país, dió a conocer que en el año 2017 una reducida parte de los hogares encuestados tienen recepción de TV digital abierta. Tanto es así, que el apagón analógico fue pospuesto por tiempo indeterminado luego del mencionado fallo del TCA. El alto costo del recambio de equipamiento, posturas sobre la democratización del acceso a la

información para personas de bajos recursos, fundamentan estas decisiones.

Durante el auge de la TVD, en los momentos posteriores a la adopción de la norma, surgió la necesidad de generar documentación técnica para la implementación de la misma. Mediante una colaboración entre la DINATEL y la Facultad de Ingeniería, se desarrolló el proyecto *gr-isdbt* [6]. Pablo Belzarena, Pablo Flores, Gabriel Gómez, Víctor González-Barbone y Federico La Rocca presentaron un receptor de televisión digital, de código abierto y bajo el paradigma de radio definida por software (SDR, paradigma de transmisión que explicaremos en detalle más adelante).

Nuestro proyecto complementa el trabajo iniciado en *gr-isdbt*, pues con la implementación del transmisor ISDB-T, se completa el sistema de transmisión de televisión digital de punta a punta. Ambos trabajos en conjunto permiten visualizar y ayudan a comprender el funcionamiento del sistema presentado en la norma, teniendo acceso completo a todo lo que sucede dentro del mismo, en cualquier punto de la cadena de transmisión y recepción.

Además, genera la posibilidad de recrear una planta de transmisión de televisión a muy bajo costo, permitiendo su implementación tanto en el hogar por entusiastas, en el aula por docentes o en la industria, por técnicos, lo cual puede colaborar con el mejoramiento de la calidad del servicio actual. A su vez, puede servir incluso como ejemplo para algunos de los cursos de la Facultad, generalmente catalogados por el estudiantado como muy teóricos y con poco alcance práctico.

Implementar un transmisor de televisión digital no es una tarea sencilla. El primer problema a enfrentar es el acceso a la información técnica. Existe poca documentación generada en el país para cumplir con las condiciones técnicas de un sistema complejo y que, además, ya lleva 7 años de vigencia como oficial. La norma presentada por la ARIB deja varias zonas grises, asume por conocidos conceptos clave, y no se explaya más de lo necesario en cuestiones de fondo.

También existen fuertes limitaciones económicas para hacerse con software o hardware comercial que resuelvan incluso algunas de las funcionalidades más básicas que exige la norma.

Esta tesis intenta resolver ambas, complementando el trabajo iniciado por el grupo ARTES con el receptor *gr-isdbt*. Se desarrolló a lo largo de este proyecto, un transmisor de televisión digital que cumple con las condiciones establecidas en la norma, y cuyas señales son decodificables por los televisores comerciales homologados por el LATU. El resultado es la posibilidad de implementar a pequeña escala y por un costo considerablemente menor, un sistema funcional de transmisión de TVD. Además, presentamos en este documento, un análisis de la norma y explicamos de qué forma se resolvieron los problemas de implementación y de las zonas “poco claras”que se mencionaron antes.

Contar con el trabajo presentado en *gr-isdbt* fue de una ayuda mayúscula, ya que disponer del código completo del receptor ayudó a comprender, sintetizar y testear los conceptos teóricos vertidos en la norma, lo que fué fundamental para la comprensión de las funcionalidades que sería necesario implementar para poder transmitir.

Para este grupo de trabajo es importante destacar lo valioso de la existencia de

proyectos de código abierto. Incontables veces encontramos en la comunidad puntos de vista, ideas y hasta algoritmos para resolver los problemas encontrados en el camino. Es por eso que esperamos poder contribuir con ella, poniendo a disposición de cualquier persona el transmisor *gr-isdbt-tx* [1], para que continúen con el trabajo de aprendizaje y la optimización del mismo por técnicos y estudiantes, seguramente con un mejor panorama del rubro, que el que tuvimos al implementar este proyecto.

Esperamos también, mediante el desarrollo de este documento, poder contribuir con la comunidad nacional de técnicos que trabajan en el rubro, y que no cuentan con documentación técnica generada por y para la norma nacional, con los problemas y las particularidades que la transmisión tiene en nuestro país y no tener que abstraer de trabajos de terceros, que resolvieron problemas similares en contextos diferentes. Entendemos que en este proyecto, los conceptos desarrollados por la norma se sintetizan en órdenes básicas al procesador, y al ser de código abierto y gratuito, se democratiza el acceso a una información a la que hoy por hoy solo se accede por medio de hardware y software propietario con licencias de costos elevados.

Para esta documentación, que acompaña el código presentado para el transmisor, definimos seis capítulos en los que se explica el desarrollo del mismo. En el capítulo 2 presentamos algunos de los conceptos fundamentales de telecomunicaciones sobre los que se construye la norma. El capítulo 3 realiza un breve pasaje por los puntos clave del sistema transmisor ISDB-T, los cuales son necesarios para comprender algunos de los bloques que conforman el sistema. Para profundizar más en los mencionados conceptos, invitamos al lector a revisar la tesis de maestría de Pablo Flores, que pueden encontrar en [7]. Luego, el capítulo 4 se detiene particularmente en el concepto de radio definida por software y presenta en detalle una implementación del mismo, en particular aquél sobre el cual se desarrolló el transmisor, que es GNU Radio en conjunto con un equipo USRP. En el capítulo 5 analizamos el código generado para implementar el transmisor, explicando en cada paso los conceptos del capítulo 2 y 3 que se necesita aplicar en cada bloque, y cómo se extrapolaron a C++, lenguaje en el que se escribió cada uno de los bloques de procesamiento. Más adelante, en el capítulo 6 mostramos el desempeño del transmisor como un todo, realizando las evaluaciones prácticas del mismo en función de los objetivos de este proyecto y se comentan los resultados obtenidos. Para terminar, en el capítulo 7, presentamos las conclusiones del proyecto en particular y planteamos algunos desafíos que sería interesante afrontar en un futuro.

Capítulo 2

Fundamento Teórico

Entendemos al proyecto *gr-isdbt-tx* como un complemento al trabajo de maestría presentado por Pablo Flores, por lo que nos parece reiterativo explicar en sumo detalle algunos de los fundamentos de telecomunicaciones sobre los que se construye la norma ISDB-T. Los mismos ya fueron explicados con gran claridad y nivel de detalle en la documentación de aquél proyecto. Quizás algunos de los lectores encuentren insuficiente el desarrollo de algunos de los conceptos incluidos en este capítulo. A ellos los invitamos a complementar la lectura de esta documentación con la tesis de maestría de Pablo Flores.

Es posible también que entre los lectores se encuentren técnicos vinculados al área de sistemas y programación. Encontrarán ellos, seguramente, mayor interés y profundidad de conceptos en los capítulos 4 y 5, donde detallamos el uso de GNU Radio como plataforma de SDR, y la estructura funcional de nuestro transmisor, así como los desafíos de programación a los que nos enfrentamos implementando cada uno de sus bloques.

Intentando contemplar el interés de todos los lectores, es que les presentamos en este capítulo un repaso sobre algunos de los conceptos que entendemos clave para comprender por qué son necesarios algunos de los bloques del transmisor.

Comenzaremos explicando aquí como se modela el canal inalámbrico para trabajar, a qué problemas se enfrenta uno cuando debe diseñar un sistema de transmisión por aire y cómo mitigar algunos de los efectos que el canal tendrá sobre nuestra señal. Repasaremos algunos de los códigos más utilizados de detección y corrección de errores. Luego vendrá una breve explicación de los sistemas OFDM, su surgimiento teórico y práctico, para contextualizar algunas de las funcionalidades de ISDB-T. Para terminar, explicaremos de qué forma se encapsulan los datos a transmitir. Para eso, repasamos algunos conceptos definidos en las normas MPEG, orientados hacia la transmisión de televisión, que ayudarán a modelar la fuente de datos del sistema.

2.1. Modelado del canal

Cuando una señal viaja a través del aire, sufrirá una serie de modificaciones a causa de muchos factores. El clima, el ruido generado por otras señales que también viajan por el mismo medio, los rebotes de la señal contra edificios, autos o cualquier objeto que se encuentre en el camino, son todos aspectos que afectarán de distinta forma la señal, por lo que la señal recibida será distinta de la transmitida.

De forma teórica, se podrían plantear las ecuaciones de Maxwell, definir condiciones de borde que modelen de manera fiel los obstáculos en el camino y calcular en cada momento y en cada lugar del espacio, la atenuación que afectará nuestra señal. Pero el canal cambia constantemente, las diferentes señales que comparten el medio también están en constante transformación, y por lo tanto no sería ni practico, ni útil seguir un camino como este para comprender el canal, mucho menos basarse en él para robustecer a la señal frente a los atenuantes que surgirán.

Es por eso que se realiza un modelado menos ambicioso del canal. Consideramos al medio como un objeto más simple, casi ideal, y la mayoría de las fuentes de interferencia se modelan dentro de una función de probabilidad a la que entendemos como "ruido", que induce errores en el receptor. Este modelo del canal podrá ser más o menos complejo, dependerá del técnico qué se enfrente al problema decidir qué herramientas incluir en función de los efectos que desea mitigar en su señal.

Consideraremos en este documento, tres modelos distintos de canal, que serán los modelos de Gauss, Rice y Rayleigh. El primero, se utiliza fuertemente para el cálculo de enlaces punto a punto. Por su parte, el modelo de Rice, es bueno para modelar los enlaces punto a multipunto, por lo tanto se utiliza ampliamente en el mundo del broadcasting. Finalmente el modelo de Rayleigh considera los efectos de la difracción y los ecos producidos por las múltiples trayectorias.

- Canal de Gauss

  El canal de Gauss es el modelo más sencillo de canal que se trabaja. Se modela al canal como un sistema cuyo efecto sobre la señal transmitida es la adición de un ruido blanco aleatorio (modelado mediante una distribución de probabilidad Gaussiana). En particular, es el modelo que se utiliza para calcular los enlaces punto a punto con línea vista. Generalmente, también se presenta la hipótesis de que el resto de las interferencias del canal, pueden modelarse como un ruido aditivo Gaussiano.

- Canal de Rice

  El modelo de canal de Rice considera que un receptor estará expuesto primero hacia un haz directo, en lo que puede ser una transmisión por línea vista, pero también considera otras trayectorias, debidas a los rebotes del haz original con los distintos obstáculos del entorno. Estos rayos secundarios generalmente vendrán acompañados de un determinado delay, de dimensión considerable.

- Canal de Rayleigh

En ISDB-T la transmisión en la capa A esta orientada a la recepción en dispositivos móviles. De estos temas hablaremos más adelante en el capítulo 3, pero por el momento, presentar esta idea nos da pie a plantear un modelo distinto de canal, uno en el que el receptor se mueve.

El canal de Rayleigh tiene en cuenta las atenuaciones que se producen por lo que se conoce como Difracción Combinada, esto es, señales que sortean obstáculos perdiendo mucha de su potencia inicial en el proceso. Este fenómeno es interesante, puesto que puede llegar a existir transmisión, pese a que no haya linea vista.

Este modelo de canal toma mucha relevancia en ISDB-T, no solo porque la mayoría de los televisores tienen antenas internas, sino que tenemos que modelar el canal para resolver el problema de la recepción móvil. Es posible robustecer a la señal de forma tal que un teléfono móvil pueda decodificar datos multimedia en alta definición y con poca latencia en todo momento, sin perder la continuidad del servicio.

Existen una serie de estrategias, implementadas en la norma ISDB-T, orientadas a resolver estos mismos problemas.

2.2. Modulación OFDM

En 1966, Chang presentó un método para lograr la multiplexión de canales de datos a través de un medio de frecuencia acotada, que elimina los efectos de interferencia intersimbólica e intercanal [8]. Implicó un cambio importante en la teoría de telecomunicaciones de la época, pues hasta entonces, los resultados existentes tomaban como funciones modulantes ortogonales, señales limitadas en el tiempo, lo que implica infinitos anchos de banda en frecuencia, lo cual en la práctica, con espectros de banda acotada se traducían en interferencias producto de los recortes en banda.

En el paper, Chang postula la idea de una nueva clase de funciones modulantes acotadas en frecuencia. Lo que permite transmitir a través de un medio lineal a la máxima tasa posible, eliminando además la interferencia intersimbólica e intercanal. Además, con estas nuevas funciones modulantes, es posible alterar de manera independiente las características de amplitud y fase, lo que abrió la puerta a la transmisión de números reales. En ese momento se sentaban las bases de una modulación OFDM (Orthogonal Frecuency-Division Multiplexing) que podría ser implementada de forma práctica.

Para lograro, consideremos en primera instancia una familia de funciones ortogonales, definida como sigue:

$$\phi_n = e^{j\omega_k t} \quad \text{donde:} \quad \omega_k = \omega_0 + 2\pi\frac{k}{t}$$

Recordemos que la ortogonalidad viene del cumplimiento de la condición:

$$\int_a^b \phi_p \phi_q^*(t)dt = \begin{cases} \text{K } si & p = q \\ 0 \ si & p \neq q \end{cases}$$

Las funciones modulantes ortogonales para OFDM serán entonces definidas de la mencionada familia de funciones, ajustando los parámetros para lograr un equiespaciado en frecuencia en el ancho de banda disponible W para las N portadoras, y agregando un factor de normalización en el periodo. Se obtiene entonces

$$\phi_n = \begin{cases} \frac{1}{\sqrt{T}} e^{j2\pi \frac{W}{N} nt} & \text{cuando} \quad t \in [0, T] \\ 0 & \text{en otro caso} \end{cases}$$

La frecuencia de cada portadora sera

$$f_n = f_0 + \frac{n}{T_{S_{OFDM}}} = f_0 + \frac{W}{N} n \quad \text{para} \quad n = 0, 1, 2, ...$$

Donde $T_{S_{OFDM}} = N.T_S$ y T_S es el tiempo de símbolo que depende del esquema de mapeo utilizado.

En cada una de esas portadoras, se mapeará un símbolo complejo $X_{n,m}$ donde n sera el número de portadora y m será el número de símbolo OFDM en la trama.

Una vez que todas las portadoras del símbolo hayan sido mapeadas, se suman todas las señales, para formar la onda que será transmitida sobre el canal para cada símbolo OFDM, que tendrá la forma

$$s_m(t) = \sum_{n=0}^{N-1} X_{n,m} \phi_n (t - mT)$$

Por lo que podemos escribir el caso general para una trama OFDM con todos sus símbolos concatenados como sigue:

$$s(t) = \sum_{m=-\infty}^{\infty} \Big[\sum_{n=0}^{N-1} X_{n,m} \phi_n (t - mT) \Big]$$

Si ponemos el foco en la ecuación anterior, notaremos que para el caso general, la señal $s(t)$ es aperiódica, lo cual dificulta lograr un buen sincronismo entre transmisor y receptor. El sincronismo perfecto entre las dos puntas del sistema es vital para eliminar el ISI, por lo tanto este problema debe ser atendido. En OFDM, la solución de sincronismo consta de la adición de un intervalo de guarda o prefijo cíclico.

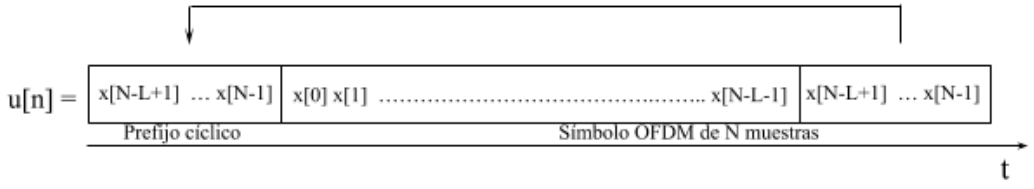

Figura 2.1: Inserción del prefijo cíclico.

Las últimas muestras del símbolo se copian al principio, de forma que al recibir, se calcula la autocorrelación del símbolo, que tendrá su pico en el comienzo de los datos válidos. Un ejemplo del prefijo cíclico se encuentra en la figura 2.1

La cantidad de muestras que se copian en el prefijo cíclico es un parámetro del sistema que los diseñadores pueden modificar según la aplicación. Generalmente se le denomina profundidad, y cuanto mayor sea la profundidad del prefijo, más robusto será el sistema frente a la ISI, pero dejará disponible un menor ancho de banda.

La implementación de OFDM en los años 60 era prácticamente imposible, uno de los mayores problemas de los sistemas multiportadora, era la gran dificultad para escalar en cantidad de portadoras. La complejidad y el costo de construir los osciladores para la cantidad de canales necesarias, mantenían la brecha entre la teoría y la ejecución práctica.

La solución a estos problemas, llegó cuando se logró programar computadoras capaces de procesar grandes cantidades de datos, mediante la implementación del algoritmo de "Fast Fourier Transform".

Weinstein y Ebert, resolvieron en su publicación de 1971 [9] el mencionado problema de la escalabilidad, mediante la utilización de los avances tecnológicos de la época. Discretizando las señales a transmitir, y modulándolas por computadora, lograron formar sistemas multicanal para la transmisión en tiempo discreto en lugar de recurrir a los bancos de osciladores que se usaban hasta el momento.

Su argumento provenía de la siguiente idea, una señal multitonal, puede ser vista como la transformada de Fourier de un tren de pulsos, y la demodulación coherente a su vez puede entenderse como la aplicación en tiempo continuo de una transformada inversa de Fourier. Entonces, probaron que muestreando la señal de origen, y mediante la implementación de un modem sobre una computadora que ejecute el algoritmo de la transofrmada rápida de Fourier, se pueden obtener aproximaciones suficientemente cercanas a los de la señal original.

En síntesis, el esquema de un sistema transmisor OFDM tiene una forma similar a la presentada en la figura 2.2. En la misma, el bloque CP identifica la inserción del prefijo cíclico, y el bloque ~~CP~~ identifica la remoción del mismo por parte del receptor.

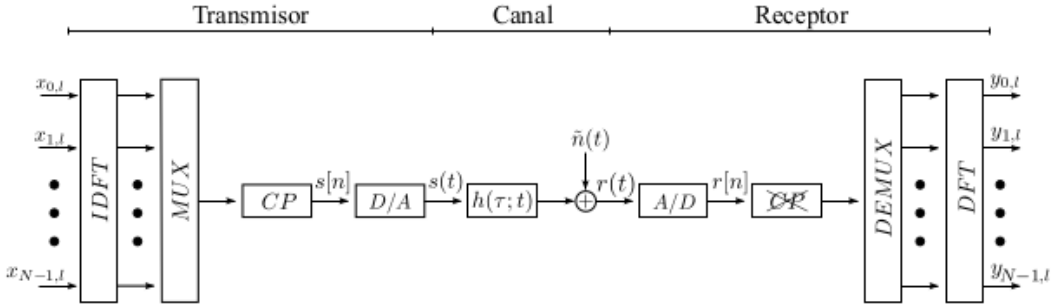

Figura 2.2: Esquema básico de un sistema OFDM.

2.3. Codificación de canal

2.3.1. Códigos de detección y correción de errores

La comunicación entre emisor y receptor puede modelarse mediante el proceso de la Figura 2.3. La situación es la siguiente, una fuente emisora envía mensajes m (palabras fuente) al receptor a través de un canal de comunicación. El mensaje debe ser traducido a algún mensaje que el canal esté capacitado para enviar, estos mensajes se conocen como palabras código.

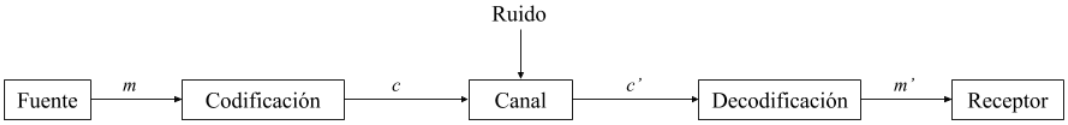

Figura 2.3: Esquema básico de codificación de canal.

Al otro lado del canal llega un mensaje codificado c', el cual seguramente sea erróneo, pues en todo proceso real de comunicación existe ruido e imperfecciones en los canales. El mensaje es decodificado en una palabra m', y generalmente $m' \neq m$.

Se desea que el receptor sea capaz de darse cuenta si el mensaje m' es realmente lo que se transmitió del otro lado, y más aún, poder corregirlo.

La Teoría de Códigos es un campo de la matemática aplicada que busca resolver los problemas de las etapas de codificación-decodificación y corrección, y que presenta su propia complejidad.

La transmisión inalámbrica de una señal la expone a diversas fuentes de ruido, con lo cual los tipos de errores generados pueden ser muy variados. Por ejemplo los errores en ráfaga, en los que un conjunto de bits consecutivos se ven alterados, son muy comunes en las comunicaciones inalámbricas. También podría suceder que el canal radioeléctrico presente distorsión en algunas portadoras en particular.

El estándar ISDB-T hace uso de distintas técnicas para la protección de los datos en transmisión. De hecho para proteger los datos en los ejemplos mencionados el estándar utiliza las técnicas de *entrelazamiento de bits y bytes* y la aplicación de códigos *forward error correction* o FEC como es el caso de Reed Solomon. Para la comprensión del estándar y el desarrollo de *gr-isdbt-tx* [1], es importante conocer el funcionamiento de estas técnicas. Profundizar en estos temas escapa los objetivos de este trabajo, por lo cual los detalles técnicos se pueden encontrar en las bibliografías mencionadas.

2.3.2. Códigos Cíclicos

El conjunto $GF(2) \triangleq \{0, 1\}$, con las operaciones de suma " + " y producto " $\times$ " usuales módulo 2, cumple con la propiedad de que cualquier elemento de $GF(2)$ distinto de cero tiene inverso. Esta propiedad se cumple trivialmente en este

conjunto y es la condición necesaria para que $GF(2)$ sea un *Campo de Galois*. Es común encontrar que a este campo también se lo llame *campo binario* y se lo denote como $\mathbb{F}_2$. Las operaciones de suma y producto definidas en $GF(2)$ son asociativas, conmutativas y distributivas, y llevan elementos de $GF(2)$ en elementos de $GF(2)$. Por esto $GF(2)$ también es un *anillo*. El conjunto de todos los polinomios con coeficientes en $GF(2)$ con las operaciones usuales de suma y producto forman un *anillo de polinomios* en $GF(2)$ y se denota como $GF(2)[x]$. Por ejemplo $g(x) = x^3 + x + 1$ es un elemento de $GF(2)[x]$.

Sea $\mathbf{c} = (c_0, c_1, ..., c_{n-1}) \in GF(2)$, con $GF(2)$ tal como se describió anteriormente. Un código $\mathcal{C}$ de bloque (n, k) se dice que es un *código cíclico* si para cada vector $\mathbf{c} = (c_0, c_1, ..., c_{n-1}) \in \mathcal{C}$ cualquier rotación circular a la derecha de C también pertenece a $\mathcal{C}$, es decir $(c_{n-1}, c_0, c_1, ..., c_{n-2}) \in \mathcal{C}$. Los códigos de bloque se caracterizan por codificar mensajes de longitud fija k en *codewords* de longitud fija n, con lo cual el tamaño del mensaje original se incrementa en $n - k$. Cada *codeword* del código $\mathcal{C}$ puede ser representada en una forma polinomial de la siguiente manera:

$$c(x) = \sum_{i=0}^{n-1} c_i x^i \tag{2.1}$$

A continuación se enumera una serie de propiedades de los códigos cíclicos, en [10] se puede encontrar una demostración detallada de cada una de ellas.

- Un código cíclico es un código lineal de bloque
- Cada *codeword* se corresponde con un polinomio
- Los polinomios del código forman un *ideal* en $GF(2)[x]/(x^n - 1)$
- Para un código cíclico existe un generador $g(x)$ que es divisor de $x^n - 1$ y que puede generar todos las *codewords* $c(x) = m(x)g(x)$

Se puede probar que esto implica la existencia de una *matriz de chequeo de paridad* $\mathbb{H} \in \mathcal{M}_{(n-k)\times n}$ tal que para toda *codeword* $\mathbf{c}$ de $\mathcal{C}$ se cumple $\mathbf{c}\mathbb{H}^T = \mathbf{0}$.

El proceso de codificación se realiza de la siguiente manera. Primero se construye el polinomio $x^{n-k}m(x)$ de grado n. Luego se divide entre el polinomio generador $g(x)$ y el resto de esa división es el polinomio de paridad $d(x)$ que se le agregará al mensaje:

$$x^{n-k}m(x) - q(x)g(x) = d(x) \tag{2.2}$$

La *codeword* se forma de la siguiente manera:

$$c(x) = x^{n-k}m(x) - d(x) = q(x)g(x) \tag{2.3}$$

Como se trata de un múltiplo de $g(x)$, entonces efectivamente es una *codeword* válida. La representación vectorial de la *codeword* queda de la siguiente manera:

$$\mathbf{c} = (-d_0, -d_1, ..., -d_{n-k-1}, m_0, m_1, ..., m_{k-1}) \tag{2.4}$$

En una situación en la que se recibe una palabra **r** cuyo mensaje es **m** y sus bits de paridad son **d**, el procedimiento para detectar si hubo error es codificar el mensaje **m** que se recibió con el mismo codificador utilizado por el transmisor (ambas partes deben conocer el polinomio generador), y luego comparar el **d'** obtenido con el **d** recibido. Si ambos difieren entonces hubo error. Por ejemplo, para un código cíclico (7, 4) con polinomio generador $g(x) = x^3 + x + 1$ se desea codificar el mensaje 1001. Los mensajes codificados tendran $n - k = 7 - 4 = 3$ bits de paridad. El mensaje en su forma polinomial queda $m(x) = 1 + x^3$. Los bits de paridad se obtienen calculando el resto de la division $x^{(7-4)}m(x)/g(x)$, los coeficientes de ese resto seran los bits de la paridad buscada. Operando se llega a que la paridad es 011 y el mensaje codificado queda 0111001.

2.3.3. Códigos Convolucionales

Los *códigos convolucionales* son un tipo de códigos lineales que debido a su estructura, la codificación mediante estos códigos puede ser vista como un proceso de filtrado o convolución, de ahí su nombre.

A diferencia de los códigos de bloque que toman bloques de k-símbolos y devuelven bloques de n-símbolos, los códigos convoulcionales funcionan como códigos de flujos. Esto significa que operan en flujos continuos de símbolos y no en paquetes discretizados. De hecho se define la tasa del código como el cociente k/n.

El funcionamiento del codificador convolucional consiste en una serie de k registros inicializados en algún valor. La salida de cada uno de estos registros ingresa a n sumadores binarios (xor). Los registros que intervienen en cada operación de suma están determinados por los polinomios generadores. Cada rama tiene asociada un polinomio generador y es común encontrar que se refiere a ellos en notación octal. Por ejemplo ISDB-T utiliza $G_1 = 171_{OCT}(1111001_b)$ y $G_2 = 133_{OCT}(1011011_b)$, significa que G_1 utiliza el bit de entrada, los primeros tres registros y el último registro.

2.3.4. Códigos de Reed-Solomon

Los códigos de Reed-Solomon son un tipo de códigos FEC no binario que corrigen los datos erróneos recibidos durante una transmisión. Se caracterizan por utilizar símbolos en lugar de bits, es decir que un conjunto de m bits consecutivos forman un símbolo. Se dice que el símbolo es erróneo si al menos un bit del símbolo es incorrecto. Las palabras del código estan formadas por $n = k+r$ símbolos, donde r son de paridad. La longitud del bloque de información es r y el código es capaz de corregir hasta $r/2$ símbolos erróneos. Debido a su capacidad de corrección de errores son especialmente indicados para tratar los errores en ráfagas.

2.4. La entrada de datos a ISDB-T

2.4.1. MPEG

El Moving Picture Experts Group (MPEG) [11] es un grupo de trabajo conformado por expertos internacionales, formado por la Organización Internacional de Normalización (ISO) en conjunto con la Comisión Electrotécnica Internacional (IEC), con el objetivo de desarrollar estándares para la codificación, compresión y transmisión de audio y video.

Uno de los estándares publicados por el MPEG, es MPEG-2. Consta de métodos para la compresión digital de contenidos audiovisuales, y abarca la difusión de los mismos a través de una amplia gama de tecnologías, desde el streaming de datos a través de la web, codificación de voz y video para telefonía y videoconferencias, comercialización de discos compactos (CD) y hasta formatos para la transmisión de televisión.

Es en este último punto donde se vincula con ISDB-T Internacional, puesto que para la codificación de fuente en la norma, fue seleccionado el Estándar MPEG-4 Parte 10 "Advanced Video Coding", también denominado H.264 [12].

Para garantizar que los receptores de televisión digital ISDB-T también sean compatibles con los transmisores tanto de ISDB-T como de ISDB-T International, se encapsulan los videos codificados en H.264 dentro de un formato denominado "Transport Stream" que se define en la norma MPEG-2 Parte 1 – Sistemas [13].

En el transmisor gr-isdbt-tx, tomamos como fuente de datos un archivo codificado como Transport Stream, para garantizar esta compatibilidad.

2.4.2. MPEG 2 Transport Stream

A la hora de transmitir televisión, los datos a transmitir son un conjunto de lo que se denomina *programas*, que son datos de video, audio, ocasionalmente subtítulos y otros datos de interés del broadcaster.

Cada programa es comprimido de forma independiente, formando lo que se denomina *Elementary Stream (ES)* o flujo elemental. Para poder ser multiplexados y transmitidos, los ES se paquetizan para formar estructuras denominadas *Packetized Elementary Stream (PES)*. Dependiendo del tipo de transmisión a realizar, los PES pueden sufrir dos tipos de modificaciones. En caso de orientarse a un entorno poco ruidoso, los PES se codifican en los llamados Program Streams, que en general son contenedores donde los paquetes de datos pueden ser de tamaño y bitrate variable. En caso de que la transmisión sea a través de un medio muy ruidoso, como es el caso de la televisión digital terrestre, los PES deben ser codificados en los llamados *Transport Streams*. Básicamente, para convertir un PES a un TS se reduce el tamaño de los paquetes a un contenedor fijo de 188 bytes, sobre los cuales se ejecutaran posteriormente códigos correctores de errores de tipo FEC, como los vistos en la sección anterior.

Al comienzo de la cadena de transmisión, en ISDB-T, se multiplexan varios TS, junto con tablas de información que contienen la información y posición de los programas en el flujo. Se crea entonces un único TS de transmisión sobre el

cual se van a realizar las tareas de codificación, robustecimiento ante pérdidas y posteriormente la modulación y otras técnicas que se verán mas adelante.

Cuando un receptor obtenga el TS en cuestión, no tendrá en principio información alguna sobre el contenido del flujo. Dicha información existe en las tablas, pero en un momento inicial de la recepción, tampoco están disponibles puesto que están multiplexadas con todos los demás paquetes del TS. Cada paquete contiene un identificador único denominado Packet Identifier (PID). Como las tablas contienen información vital para la decodificación, sus números PID son bien conocidos por el receptor, de forma que puede buscar entre todos los paquetes aquellos que le darán la información para comenzar la decodificación. La primera tabla que el receptor busca en el TS, es la tabla PAT, que tiene asignado el PID 0x0000.

2.4.3. Tabla PAT

La Program Association Table (PAT), es una tabla que contiene una lista de todos los programas disponibles en el TS. Está formada por valores de 16 bits denominados Program Number, asociados cada uno de ellos con el PID correspondiente a la tabla PMT de cada uno de los programas dentro del stream.

De este modo, el receptor que se conecta al stream en cualquier momento de la transmisión, lee la tabla PAT y obtiene un listado de los programas existentes en el TS y los PIDs de sus correspondientes PMTs.

2.4.4. Tablas PMT

Cada programa del TS tiene su tabla PMT (Program Map Table), identificada tambien por su PID único. En la tabla PMT, esta la información correspondiente a los PIDs de los paquetes que pertenecen al programa en cuestión.

Una vez obtenida la tabla PMT del programa que se desea decodificar, el receptor debe almacenar los PIDs que estan contenidos en ella. Luego, basta con filtrar los paquetes del TS para obtener solo los datos del programa a decodificar.

Además de la tabla PAT y las PMT, existen otros tipos de tablas en MPEG-2. Para el alcance de este documento, con la descripción de estas dos es suficiente, pero si nos interesa destacar otro tipo de paquete del TS con un PID bien conocido.

2.4.5. Paquetes Nulos

Resulta vital para un esquema de transmisión de televisión mantener el bitrate constante, puesto que esto mantiene el ancho de banda de tamaño constante. En una transmisión de datos multimedia, hay variabilidad en la tasa de bits todo el tiempo, por lo que puede ocurrir que en algún momento falten datos para contemplar esa tasa. Es por eso que se definen los paquetes nulos. Un multiplexor completa con paquetes “dummy“ cuando ocurre esta deficiencia, y así logra para mantener la tasa constante. El PID de los paquetes nulos es el 0x1FFF.

2.4.6. Broadcast Transport Streams

El estándar MPEG no está pensado para la transmisión en capas jerárquicas. Para lograr la característica de la transmisión jerárquica y la recepción parcial, ISDB-T cuenta con una solución propia denominada Broadcast Transport Stream. Esta solución consiste en el uso de un flujo de transporte propio, formado a partir de tres flujos MPEG multiplexados, lo que supone un procesamiento no tan sencillo. El nuevo flujo deberá incluir cierta información jerárquica, que no provee MPEG, que permita identificar los flujos con sus correspondientes capas jerárquicas. Es importante que el sistema funcione a una velocidad de reloj constante y determinada, por este motivo el flujo BTS deberá tener una tasa perfectamente definida y constante independientemente de los parámetros de cada capa.

La conformación del BTS se lleva a cabo en el bloque TS Remux. Es el encargado de agregar 16 bytes al final de cada paquete TS con la ISDB-T information y la paridad; posicionar los TSP dentro del BTS de acuerdo a cierto patrón de ordenamiento que se detallará, para posibilitar la transmisión jerárquica; insertar la cantidad necesaria de paquetes nulos para asegurar una tasa constante r_{BTS}.

$$r_{BTS} = 4 \times f_{IFFT} = \frac{2048}{63} \approx 32{,}508 \quad Mbps$$

La obtención de la fórmula anterior, en conjunto con una descripción mas detallada del Multiplex Frame para ISBD-T puede encontrarse en el análisis realizado por Pablo Flores y Federico La Rocca en [7].

Durante la división jerárquica la ISDB-T information de los TSP es removida por lo cual en las siguentes fases de procesamiento ya no se cuenta con la información jerárquica en los flujos de transporte. Es por eso, que resulta necesario definir un orden dentro del BTS de manera que el receptor pueda reconstruirlo perfectamente sin necesidad de la ISDB-T information.

Supongamos que tenemos un BTS como el de la figura 2.4 en el cual hay dos capas jerárquicas A y B, cada una con su respectiva tasa tal que $r_{BTS} > r_B > r_A$. Las capas con tasas altas implican un tiempo de transmisión de paquetes mucho más corto que las capas con tasas bajas, y viceversa.

Observando el diagrama de la figura 2.4 puede verse que en recepción el paquete B_1 es procesado antes de que termine de ser transmitido el paquete A_1, cuando en realidad el orden original era A_1 y luego B_1. Además se generan espacios de tiempo en los cuales no se entregan paquetes al sistema porque aún se están procesando paquetes que van arribando. Esos intervalos de tiempo deben ser rellenados con paquetes nulos a fin de mantener la tasa constante. Esto resulta en un desordenamiento del patrón original del BTS y de la pérdida total de la capacidad de reordenar los datos en recepción.

Para formar los patrones de ordenamiento en transmisión de manera tal que el receptor sea capaz de recuperarlos perfectamente, se agregan los denominados ajustes de atraso. En el ejemplo de la figura 2.5 se agrega un ajuste de atraso de 2 TSP y se introduce un paquete nulo en el BTS original. Como resultado, el flujo BTS reconstruído por el receptor resulta ser exactamente igual al BTS original. Dada una configuración jerárquica del sistema, existe un único patrón de

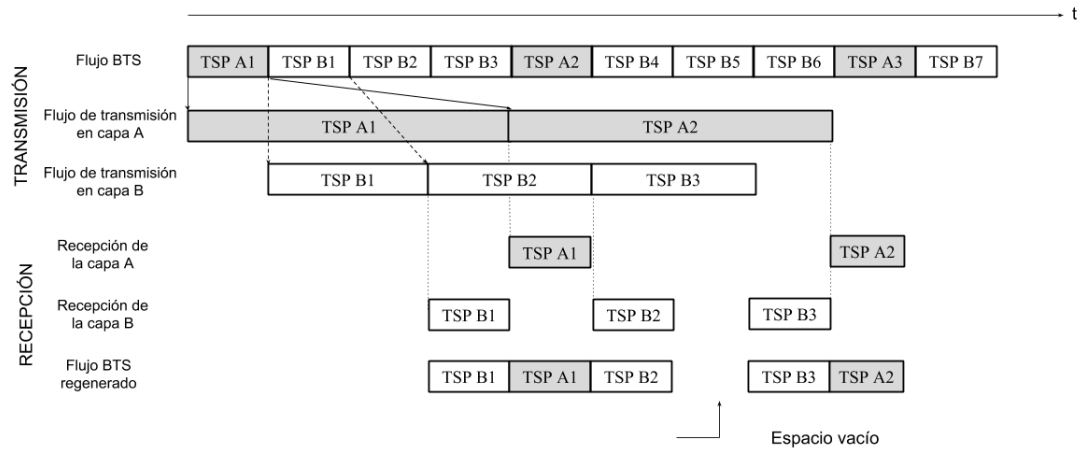

Figura 2.4: Patrón de ordenamiento incorrecto del BTS.

ordenamiento del BTS.

El transmisor implementado en este trabajo toma como flujo de entrada un BTS ya conformado, y con la información jerárquica de los TSP es que logra procesar cada capa por separado. De ahí en adelante las capas son entrelazadas y moduladas cada una de acuerdo a su propia configuración.

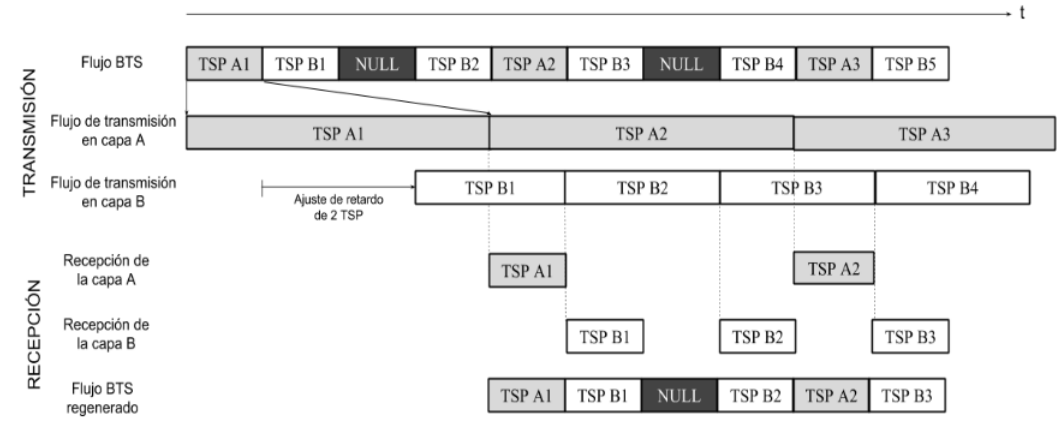

Figura 2.5: Patrón de ordenamiento correcto del BTS.

Capítulo 3

El Sistema de Televisión Digital Terrestre ISDB-T

El esquema de transmisión del estándar ISDB-T que se presenta en la Figura 3.1 puede ser caracterizado en cuatro grandes etapas que serán presentadas en este capítulo: el flujo de transporte BTS (*Broadcast Transport Stream*), la etapa de robustecimiento de la señal, la formación de los Cuadros OFDM con sus señales piloto y la puesta en el aire de la señal.

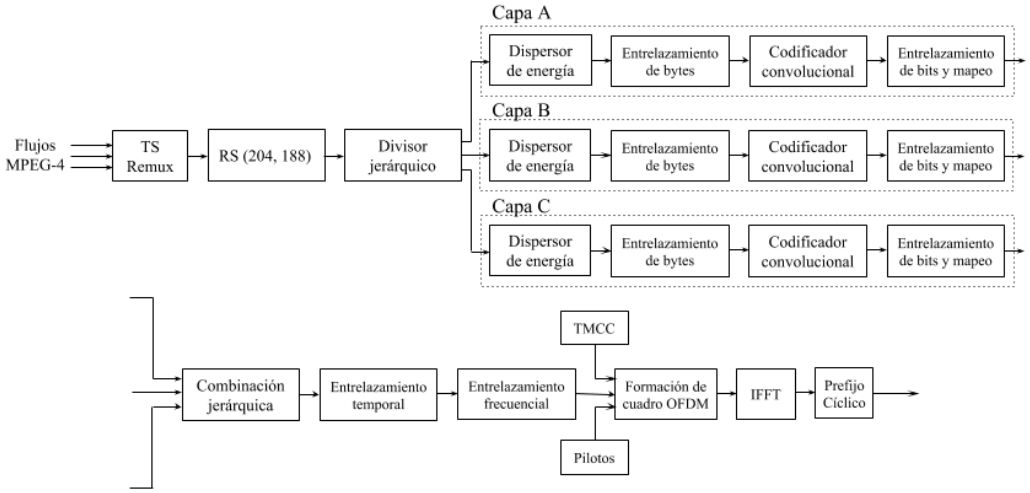

Figura 3.1: Diagrama de bloques del transmisor ISDB-T definido por la ARIB.

Cada una de esas etapas tiene a su vez varias subetapas, o bloques fundamentales, que se enfocan en resolver los distintos problemas discutidos que surgen al transmitir una señal inalámbrica. Esto dá como resultado una variedad de parámetros de operación presentados en la Tabla 3.1 que permitirán por ejemplo, la transmisión jerárquica en distintas capas, cada una con su propia configuración; o enfocarse en robustecer la transmisión frente al *multipath* o al efecto Doppler.

A nivel de la distribución en el espectro, el estándar utiliza 6 MHz de ancho de banda de canal distribuído en 14 segmentos de los cuales sólo 13 son utilizados para

enviar datos, el restante se utiliza como guarda a ambos lados del canal. También se incluye un piloto continuo que se trata de una portadora modulada en BPSK ubicada a continuación del Segmento 12.

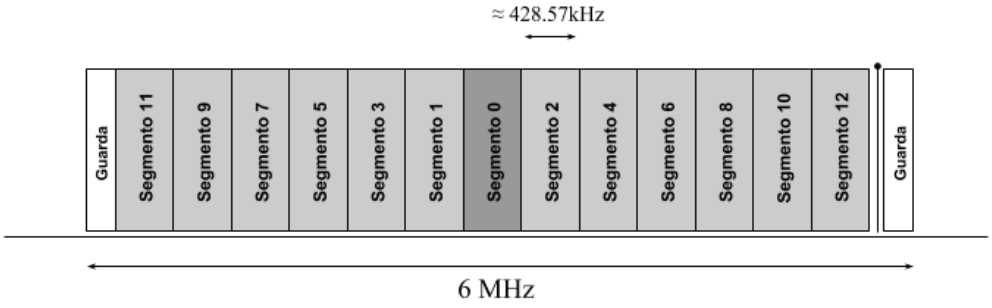

Figura 3.2: Distribución espectral de los segmentos OFDM. A la derecha, entre el segmento 12 y el intervalo de guarda, puede apreciarse la presencia del piloto continuo.

Tener un esquema de transmisión jerárquica permite asignar a cada capa jerárquica la cantidad de segmentos que uno quiera. El estándar de la ARIB denomina a estas capas como A, B y C, y es posible tener configuraciones con solamente 2 capas presentes tal como la de la figura 3.2 en la que la capa A corresponde al segmento 0 y el resto de los segmentos forman la capa B.

En los comienzos de la televisión digital en el Uruguay los operadores que prestaban servicios de televisión digital debían transmitir su señal en calidad HD, en SD y *oneseg*, es decir en tres capas jerárquicas. Esto era así porque se entendía que en el mercado circulaba una gran cantidad de receptores que no eran capaces de procesar la calidad HD. Posteriormente esa directiva fué suprimida y al día de hoy los operadores no están obligados a transmitir en SD.

El sistema puede operar en tres *modos de transmisión* diferentes que se caracterizan por la cantidad de portadoras utilizadas, siempre en el mismo ancho de banda de 6 MHz. Las portadoras utilizadas son $2^{10+modo}$ donde el modo puede ser 1, 2 o 3.

Se define la frecuencia de muestreo de la IFFT como $f_{IFFT} \triangleq \frac{512}{63\mu s} \approx 8{,}127MHz$, que a su vez coincide con el número de portadoras utilizadas sobre el tiempo de símbolo activo T_s. Por lo tanto la utilización de un número mayor de portadoras implica utilizar símbolos más largos para respetar la relación de la f_{IFFT}; aumentar el tiempo de símbolo activo implica reducir el ancho de banda ocupado por cada portadora.

Cada segmento a su vez está conformado por dos tipos de señales portadoras. Las primeras son las correspondientes a los datos, y las otras son las denominadas *portadoras piloto* que cumplen se utilizan para la estimación del canal y transmisión de parámetros de control e información del sistema. De manera general, los segmentos tienen $96 \times 2^{modo-1}$ portadoras de datos $12 \times 2^{modo-1}$ portadoras piloto dependiendo del modo de transmisión. En la Tabla 3.1 se evalúan estos parámetros para los distintos modos de transmisión.

Parámetro	Valor
Ancho de banda del canal	6 MHz
Cantidad de segmentos	13
Ancho de banda de cada segmento	$6000/14 \approx 428{,}57kHz$
Cantidad de portadoras activas por segmento	96 de datos y 12 pilotos (Modo 1) 192 de datos y 24 pilotos (Modo 2) 384 de datos y 48 pilotos (Modo 3)
Duración de símbolo activo	$252\mu s$ (Modo 1) $504\mu s$ (Modo 2) $1008\mu s$ (Modo 3)
Duracion del prefijo cíclico	1/4, 1/8, 1/16, 1/32 (fraccion del símbolo activo)
Tasa de código convolucional	1/2, 2/3, 3/4, 5/6, 7/8
Tasa de código Reed-Solomon	(188, 204)
Profundidad del entrelazamiento temporal	0, 1, 2, 4 (Modo 1) 0, 2, 4, 8 (Modo 2) 0, 4, 8, 16 (Modo 3)
Esquemas de modulación	DQPSK, QPSK, 16QAM, 64QAM
Frecuencia de muestreo (f_{IFFT})	$512/63 \approx 8.127$ MHz

Tabla 3.1: Parámetros relevantes en el estándar ISDB-T.

3.1. BTS como fuente de datos

El estándar ISDB-T admite la posibilidad de tomar hasta tres *Transport Streams* (TS) MPEG-2. A estos flujos se les debe agregar la información necesaria para la transmisión en capas jerárquicas. El bloque TS Remux es el que se encarga de multiplexar los tres flujos de transporte y a cada TSP agregarle 16 bytes de información. Una vez que se agregan estos datos los paquetes de cada capa son multiplexados según un patrón de ordenamiento que es único para cada configuración del sistema, en [14] se explica este patrón y se presenta un algoritmo para recuperarlo en recepción. El transmisor implementado en este trabajo toma como flujo de entrada un BTS ya conformado, y con la información jerárquica de los TSP es que logra procesar cada capa por separada. De ahí en adelante las capas son entrelazadas y moduladas cada una de acuerdo a su propia configuración.

3.2. Robustecimiento frente a las no idealidades del canal

Como primer medida para proteger los datos del BTS se aplica un código Reed-Solomon (204, 188). El proceso de codificación consiste en sustituir los últimos 16 bytes de los TSP por una paridad que permite corregir hasta 8 bytes de error en el paquete.

Este bloque remueve la información jerárquica agregada por el TS Remux con lo cual una vez que los TSP son codificados por el Reed-Solomon en principio ya no es posible distinguir a qué capa pertenece cada paquete, es decir que al llegar al bloque de división jerárquica ya no se cuenta con esa información. Una estrategia para sortear este problema, y de hecho la que se implementa en *gr-isdbt-tx*, consiste en separar primero los paquetes correspondientes a cada capa jerárquica y luego codificar los paquetes. Una vez encaminados los paquetes se puede prescindir de la información jerárquica. El diagrama de bloques presentado en la Figura 3.1 corresponde al esquema original de la ARIB en la que se define el estándar ISDB-T, mientras que la Figura 5.1 se encuentra la implementación utilizada en *gr-isdbt-tx*.

Para evitar los problemas de sincronismo que podría ocasionar una señal con muchos ceros o unos consecutivos, se utiliza un *dispersor de energía* que se encarga de generar un flujo pseudoaleatorio. Este proceso se define de manera tal que resulta ser totalmente invertible y por lo tanto aplicando el mismo proceso en recepcíon se obtiene la secuencia original.

Los canales inalámbricos son propensos a una multiplicidad de fuentes de error. Un tipo de errores muy comunes en estos canales son los *errores en ráfaga* provocando que una secuencia consecutiva de bits se corrompan. El impacto de estos tipos de errores crece con la velocidad de transmisión; a mayor velocidad de transmisión un mismo error afecta a una mayor cantidad de bits. Resulta necesario lograr algún tipo de inmunidad ante estos errores, es por ello que en muchos sistemas, y en particular en ISDB-T, se suelen concatenar códigos correctores de errores con entrelazamientos de datos, para dispersar los eventuales errores puntuales y aumentar la potencia de la corrección. David Forney en su trabajo *Concatenated Codes* [15] estudia el compromiso que existe en la utilización de sucesivos códigos concatenados, y en especial la performance que pueden llegar a alcanzar los códigos Reed-Solomon concatenados. El estándar ISDB-T utiliza este enfoque a través de la implementación de un código Reed-Solomon como *outer code* y un código convolucional como *inner code*. Como medida adicional para contrarrestar los efectos del canal se utiliza el *entrelazamiento de bits* que esencialmente consiste en introducir retardos variables en los bits que se transmiten.

Al transmitir en múltiples portadoras ortogonales se corre el riesgo de atravesar canales selectivos en frecuencia y por lo tanto que ciertas portadoras se vean continuamente afectadas. Para mitigar esto una estrategia puede ser rotar de portadora los contenidos que se transmiten, es decir hacer un *entrelazamiento frecuencial*. Con esto se logra que ante un canal selectivo, ya lo sea por sus características propias o por interferencias de señales espurias fuera de banda, las portadoras que son afectadas no resulte en una pérdida de bloques de información irrecuperables.

3.3. Formación de los cuadros OFDM

Es usual encontrar que en los sistemas de comunicación digitales se delimiten los datos transmitidos dándoles forma de cuadros con una duración fija. Esto

vuelve más sencillas las tareas de sincronización dado que en recepción se sabe en qué momento comienza un cuadro. Particularmente en los sistemas OFDM estos cuadros se denominan precisamente *cuadros OFDM*.

En ISDB-T la tasa de transmisión de datos, en bits por segundo, depende de las relaciones de codificación introducidas por los códigos correctores, de la modulación, de la cantidad de portadoras utilizadas y del tiempo de símbolo OFDM.

Se puede calcular esta tasa a partir de todos esos parámetros de la siguiente manera:

$$R = k_o k_i b_p \frac{(13 \times L_D)}{T_s} \tag{3.1}$$

Donde $k_o = 188/204$ es la relación del código exterior (Reed-Solomon), k_i la de código convolucional (1/2, 2/3, 3/4, 5/6 ó 7/8), b_p la cantidad de bits utilizados por la modulación (2-QPSK, 4-16QAM, 6-64QAM), L_D la cantidad de portadoras de datos por segmento y T_s el tiempo de símbolo OFDM.

Sea F la cantidad de símbolos OFDM que entran en un cuadro, entonces la duración de un cuadro OFDM está dada por $T_F = F \times T_s$.

La cantidad de bits de datos que se pueden transmitir por segmento está dada entonces por la siguiente expresión:

$$b_T = R \frac{T_F}{13} \tag{3.2}$$

Un TSP está conformado por 188 bytes de datos, con lo cual la cantidad de bits b_T expresada en términos de TSPs queda de la siguiente manera:

$$b_T = (188 \times 8) \times N \tag{3.3}$$

Igualando las dos expresiones para b_T se llega a que

$$N = \frac{k_i b_p L_D F}{204 \times 8} \tag{3.4}$$

De todos los casos posibles, y considerando que N debe ser entero, el valor más pequeño para un cuadro OFDM resulta ser $F = 204$ símbolos OFDM.

Luego de conformarse el cuadro OFDM los datos ya están listos para ser transmitidos mediante la modulación OFDM. En este punto los símbolos son procesados por el bloque de la IFFT y a continuación es necesario agregar el prefijo cíclico. Una vez agregado el prefijo las muestras ya estan prontas para ser dirigidas al conversor DAC, a una tasa de $\frac{64}{63} \times 8MHz \approx 8{,}127MSamples/s$, para la transmisión en ondas electromagnéticas.

3.4. Necesidad del prefijo cíclico

Cuando se pasa de trabajar en el dominio continuo al dominio discreto muchas cosas no funcionan igual. Si bien es cierto que en el dominio continuo la transformada de Fourier de un producto convolución coincide con la transformada

de cada factor, esto deja de ser válido en tiempo discreto; si $y[n] = x[n] * h[n] \Rightarrow DFT\{y[n]\} \neq X[q]H[q]$. Sería bueno poder tener una relación similar a la que ocurre en tiempo continuo y así poder separar fácilmente las muestras que se enviaron de la respuesta del canal.

Afortunadamente existe un tipo de producto convolución especial llamado *convolución circular* que nos devuelve esa propiedad de la transformada de Fourier. Supongamos que la señal que enviamos es $x[n]$ y la respuesta del canal $h[n]$ con $n = 0, .., N-1$. La convolución circular de dos señales $x[n]$ y $h[n]$ se define de la siguiente manera:

$$x[n] \circledast h[n] = \sum_{k=0}^{N-1} x[(n-k) \ \ mod \ \ N]h[k] \tag{3.5}$$

Para evitar sobrecargar la notación se suele utilizar la siguiente nomenclatura $x_N[n-k] \triangleq x[(n-k) \ \ mod \ \ N]$. Con lo cual la convolución circular queda expresada:

$$x[n] \circledast h[n] = \sum_{k=0}^{N-1} x_N[n-k]h[k] \tag{3.6}$$

La señal $x_N[n-k]$ no es otra cosa que la señal $x[n-k]$ periodizada cada N muestras. Tomemos la transformada discreta de Fourier de esta convolución circular y veamos lo que sucede.

$$Y[q] \triangleq DFT\{x[n] \circledast h[n]\} = \sum_{n=0}^{N-1} e^{-j2\pi\frac{nq}{N}} \sum_{k=0}^{N-1} x_N[n-k]h[k] \tag{3.7}$$

$$Y[q] = \sum_{k=0}^{N-1} h[k] \sum_{n=0}^{N-1} e^{-j2\pi\frac{nq}{N}} x_N[n-k] \tag{3.8}$$

Renombrando la variable m como $n-k$ se tiene:

$$Y[q] = \sum_{k=0}^{N-1} h[k] \sum_{m=-k}^{N-1-k} e^{-j2\pi(m+k)\frac{q}{N}} x_N[m] \tag{3.9}$$

Todo lo que no depende de m en la segunda sumatoria puede salir como factor

$$Y[q] = \sum_{k=0}^{N-1} e^{-j2\pi\frac{mq}{N}} h[k] \sum_{m=-k}^{N-1-k} e^{-j2\pi\frac{mq}{N}} x_N[m] \tag{3.10}$$

En la ecuación 3.10 puede apreciarse que la sumatoria en k corresponde a la $DFT\{h[k]\}$. La otra sumatoria resulta un poco más complicado de verla pero pensemos que $x_N[m]$ es la señal $x[m]$ de N muestras periodizada; la sumatoria se lleva a cabo en una ventana de N muestras por lo cual esto coincide con la transformada discreta de Fourier de la señal $x[n]$. Dicho de otra manera, la sumatoria de la ecuación 3.10 es una manera de calcular la DFT en la que se suman los términos en otro orden, pero al final la suma es la misma.

Con esto queda demostrado que la DFT de la convolución circular de dos señales coincide con el producto de sus transformadas discretas, es decir que $DFT\{h[n] \circledast x[n]\} = H[q]X[q]$.

Pasemos a ver qué sucede con la señal OFDM cuando se propaga por el canal inalámbrico. En principio el canal podría variar con el tiempo, con lo cual el modelo discreto de la señal recibida queda de la siguiente manera:

$$r[n] = \sum_{k=-\infty}^{\infty} h_k[n]x[n-k] + w[n] \tag{3.11}$$

Donde $w[n]$ corresponde al ruido gaussiano aditivo que introduce el canal, y $h_k[n]$ es la respuesta al impulso del canal que en principio puede ser una función del tiempo. Para el estudio que realizaremos se asumirá que el canal es un sistema lineal invariante en el tiempo y su respuesta está definida por L muestras, con lo cual se elimina la dependencia de n:

$$r[n] = \sum_{k=0}^{L-1} h_k x[n-k] + w[n] \tag{3.12}$$

La señal resultante tendrá $N + L - 1$ muestras dado el efecto que tienen las muestras de la respuesta al impulso del canal. Esto hace que al enviar dos símbolos OFDM consecutivos haya un solapamiento de $L - 1$ muestras en recepción. Como primera medida podría separarse los símbolos enviados la cantidad de muestras necesarias. Sin embargo existe otra solución que consiste en el agregado de un *prefijo cíclico*. La idea es utilizar esos L espacios para transmitir las últimas L muestras del símbolo siguiente tal como se muestra en la figura 2.1.

El estándar ISDB-T ofrece la posibilidad de utilizar distintos largos de prefijo cíclico. Una manera de establecer sincronismo, y también poder conocer los parámetros básicos de transmisión desde el lado del receptor como el modo y el largo del prefijo cíclico, consiste en tomar símbolos OFDM consecutivos que van arribando y retardarlos una cierta cantidad de símbolos. Luego se calcula una función de verosimilitud propuesta por Van de Beek en [16] y según su característica es que se puede determinar el largo del prefijo cíclico y el modo de transmisión.

La nueva señal conformada a partir de $x[n]$ con su prefijo cíclico queda expresada de la siguiente manera:

$$u[m] = \begin{cases} x[N-L+m+1] & \text{si } m = 0, ..., L-2 \\ x[m-L+1] & \text{para } m = L-1, ..., N+L-2 \end{cases} \tag{3.13}$$

Ahora con esta nueva señal que hemos definido, en recepción llegará la siguiente señal:

$$v[n] = \sum_{k=0}^{L-1} h[k]u[n-k] + w[n] = \sum_{k=0}^{L-1} h[k]x[n-L-k] + w[n] \tag{3.14}$$

La superposición de las primeras $L-1$ muestras de v hace que deban descartarse. Por lo tanto se considerarán las muestas con $n = L, ..., L+N-1$ y entonces es posible considerar un n' el cual contempla que las muestras indicadas ya han sido descartadas. Como este es un proceso cíclico símbolo a símbolo, la ecuación 3.14 queda así:

$$v[n'] = \sum_{k=0}^{N-1} h[k]x_N[n'-k] + w[n'] \text{ con } n' = [0, ..., N-1] \tag{3.15}$$

Esta última ecuación no es otra cosa que la convolución circular $h \circledast x$ y que, como se vió anteriormente, su transformada de Fourier resulta ser $Y[q] = H[q]X[q]$. De esta manera el receptor una vez que realizó su estimación del canal es capaz de separar las muestras $X[q]$ y así recuperar la señal transmitida.

3.5. Las portadoras y la modulacion

Cada segmento OFDM está conformado por una cantidad de portadoras en función del modo de transmisión. La mayor cantidad de portadoras está destinada a la transmisión de los datos mientras que algunas otras son utilizadas como mecanismos para la estimación del canal, sincronización, transmisión de parámetros de funcionamiento entre otros.

Scattered Pilot

Las portadoras piloto son señales moduladas en BPSK que se generan a partir de una secuencia pseudoaleatoria PRBS (Pseudo Random Binary Sequence). El procedimiento para generar estas portadoras consiste en evolucionar el circuito de la figura 3.3 y tomar W_i para la modulación BPSK. El índice de la portadora SP dentro del símbolo OFDM es el que determina la cantidad de veces que se debe evolucionar el registro para obtener W_i. Por último la modulación se realiza como $(4/3, 0)$ si $W_i = 0$, y $(-4/3, 0)$ si $W_i = 1$.

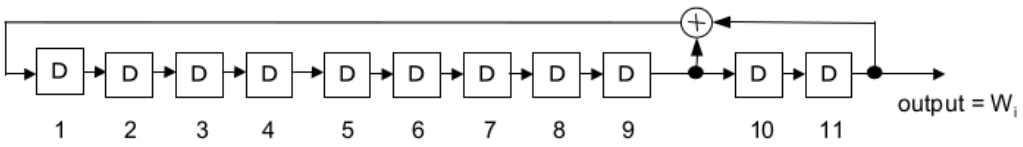

Figura 3.3: Circuito generador de la PRBS para la scattered pilot.

Las SP se ubican cada 12 portadoras y a medida que transcurren los símbolos OFDM las mismas van rotando $3 \times (n \ mod \ 4)$, donde n es el número de símbolo OFDM. Esto es así ya que el objetivo de estas portadoras es la estimación del canal y por lo tanto conviene rotarlas para evitar que caigan siempre en un lugar que presente fade en frecuencia.

Portadoras TMCC

Las portadoras TMCC son las que transportan toda la información de las capas jerárquicas y la información de control. Si bien en los cuadros OFDM se transmiten varias TMCC estas tienen exactamente el mismo contenido, lo que da al receptor la posibilidad de ponderar todas las TMCC que recibe para quedarse con las que tienen mayor porcentaje de coincidencia entre sí. Se compone de 204 bits modulados en DBPSK de los cuales el primero es un bit de referencia para realizar la modulación diferencial. Este bit coincide con el valor de la W_i correspondiente al primer símbolo del cuadro OFDM. Los siguientes 16 bits son utilizados por una señal de sincronismo que puede tomar los valores $w_0 = 0011010111101110$ y $w_1 = 1100101000010001$, alternándose entre una y otra en cada cuadro OFDM. A continuación se encuentran los datos de la información jerárquica y otros parámetros para los que se dedican 105 bits. Estos 105 bits son protegidos por un código cíclico acortado (200,118) con lo cual se dedican los siguientes 82 bits para la paridad.

Portadoras Auxiliares

Las portadoras auxiliares están pensadas para transmitir información complementaria del control de transmisión. Utilizan modulación diferencial DBPSK con referencia W_i al igual que la TMCC. Su uso es opcional y el sistema puede trabajar perfectamente prescindiendo de ellas; en caso de no ser usadas los bits correspondientes a las portadoras AC se rellenan con ”1”.

Piloto Continuo

Al igual que las SP, se agrega un piloto continuo a la derecha del segmento 12 con propósitos de sincronismo en recepción. Su modulación también es BPSK y toma el valor de $(-4/3, 0)$, $(4/3, 0)$, $(4/3, 0)$ para los modos 1, 2 y 3 respectivamente.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 4

Radio definida por Software

Cuando en telecomunicaciones se habla de radio, se hace referencia al hardware necesario para transmitir o recibir información, haciendo uso de la banda del espectro radioeléctrico ubicado en radiofrecuencia (RF).

La mayoría de los sistemas de comunicación de hoy en día necesitan de un radio: la telefonía móvil, la navegación por satélite, la aviación comercial, la televisión abierta, son todos ejemplos. En general los radios varían en dimensiones y costos según la aplicación, pero son muy poco versátiles en cuanto a sus capacidades. Con cada actualización tecnológica que reciben las redes, por ejemplo la modernización de la red móvil de 3G a LTE, implica una renovación total del equipamiento de radio.

El concepto de Radio Definida por Software no es nuevo, algunas de sus ideas fundacionales se publicaron en la década de los 80, impulsadas por algunos de los trabajos de Joseph Mitola [17]. Pero ha sido el masivo desarrollo de la industria tecnológica el que ha impulsado en los últimos años la implementación de sistemas basados en SDR.

Para implementar un equipo de radio definida por software, solo es necesario contar con una antena de radiofrecuencia, capaz de transmitir/emitir señales analógicas, y un conversor analógico-digital que alimenta de muestras a un procesador de propósito general. Es aquí donde radica la diferencia con los radios implementados en hardware, en los cuales se implementa toda una circuitería electrónica para realizar el procesamiento. La masividad de los procesadores y el auge de la programación de hoy en día, permiten implementar fácilmente un SDR, y realizar todo tipo de proyectos de distinta escala, reutilizando el equipamiento.

Durante la realización de este proyecto, utilizamos como entorno de desarrollo para crear los bloques de procesamiento de señales al software GNU Radio [18], y como hardware utilizamos un equipo B200 del fabricante Ettus Research [19] junto con una antena telescópica genérica.

4.1. GNU Radio

GNU Radio es un entorno de desarrollo orientado a procesamiento de señales, gratuito y de código abierto. Mediante la interconexión de bloques de procesamiento, puede usarse en conjunto con antenas de RF para desarrollar SDRs, o en una versión local en modo de simulación de sistemas.

La aplicación por defecto trae una amplia gama de bloques funcionales para trabajar, desde herramientas simples como filtros y ecualizadores, hasta estructuras complicadas, como lo es un transmisor DTV completo. Existe además una gran comunidad, muy activa, que está permanentemente publicando nuevos contenidos, para ampliar la gama de herramientas existentes en la actualidad.

Esto se da porque al ser de código abierto, es relativamente sencillo crear bloques nuevos. El entorno soporta desarrollos de código en Python y en C++, siendo este ultimo el lenguaje con el que hemos implementado nuestro transmisor gr-isdbt-tx. Si bien puede parecer un poco complejo comprender la metodología de trabajo en GNU Radio en un principio, una vez que comenzamos a trabajar con él y atravesamos la curva de aprendizaje, nos encontramos con una herramienta muy poderosa, y con la que es más sencillo comprender algunos de los conceptos teóricos establecidos en la norma. Durante la carrera, nos habíamos encontrado con GNU Radio en algunas ocasiones, donde pusimos en práctica algunos conceptos básicos de sistemas de comunicación. No fue hasta esta instancia, en la que pudimos encontrarnos con el potencial total de la herramienta.

Pasaremos a desarrollar algunos de los conceptos clave para comprender el uso de GNU Radio, ejemplificando con nuestro transmisor gr-isdbt-tx

4.1.1. Flowgraphs

El flowgraph es la estructura de datos más básica del programa. Puede entenderse al mismo como si fuera una mesa de trabajo. Dentro de un flowgraph, vamos a insertar nuestras muestras desde la fuente, las vamos a procesar y las vamos a exportar, ya sea hacia un archivo o hacia el canal a través de una antena.

4.1.2. Bloques

En GNU Radio, los datos siempre se mueven a través de bloques. Un bloque puede tener la fuente de datos, realizar operaciones sobre las muestras, puede exportar datos hacia el exterior del entorno, o puede tener toda una estructura de bloques adentro. Existen cuatro tipos de bloques, y se diferencian por la tasa de muestras que atraviesan el mismo.

Bloques Síncronos (1:1), este bloque permite implementar funcionalidades que consuman y produzcan la misma cantidad de muestras por puerto. Los bloques síncronos pueden tener cualquier cantidad de entradas y salidas. En particular, cuando un bloque síncrono tiene 0 entradas, se tiene una fuente de datos (source). Cuando, al contrario, tiene 0 salidas, decimos que es un sumidero (sink). El bloque dispersor de energía por ejemplo, es un caso de bloque síncrono de gr-isdbt-tx. Volveremos sobre esto en el capítulo 5.

Bloques Decimadores (N:1), son los que producen menos muestras de las que consumen, y de forma análoga, existen también los *Bloques Interpoladores (1:M)*.

Finalmente tenemos el caso general, *Bloques Generales (N:M)*, en el cual no existe a priori una relación entre muestras de entrada y salida, y cada usuario debe definirlo para el bloque en particular que esta creando mediante una función llamada forecast. Uno de los más claros casos de bloque general en nuestro transmisor, es el bloque que crea el cuadro OFDM. En tiempo de compilación, no sabemos los parámetros de transmisión que determinan la tasa de muestras. Es en tiempo de ejecución que estos valores quedan definidos, y el propio motor de GNU Radio se encarga de orquestar el flujo de datos para que todo funcione en régimen.

Para implementar el transmisor, tuvimos que separar las funcionalidades en bloques de procesamiento, e implementar aquellos bloques que no estuviesen ya presentes en el software. Algunos ejemplos como conversores serie a paralelo, el codificador Reed Solomon, el codificador Viterbi, existían ya como bloques funcionales del programa, y fueron re utilizados.

4.2. El flujo de datos en GNU Radio

Para el desarrollo de los bloques de gr-isdbt-tx fué necesario comprender el modo en el que los datos se mueven entre bloques en GNU Radio. El motor del programa realiza llamados de forma periódica a los bloques en el flowgraph, dependiendo de la cantidad de muestras que tenga en cola para procesar en cada bloque, y de la tasa de muestras que busque mantener constante a través del sistema, le comunica a cada bloque la cantidad de datos que necesita que le dejen en salida.

Al momento de crear un nuevo bloque, debemos especificar en la firma de la función, los parámetros necesarios para que el motor central pueda controlar el flujo de datos de nuestro nuevo bloque. Para realizar esto, se utilizan ciertos parámetros, precargados en la estructura de funciones de la clase, orientados a la entrada/salida de datos, estos son, noutputitems y ninputitems.

Mediante estas variables, el motor de procesamiento puede obligar a los bloques a ejecutarse más de una vez en cada llamado, pues exige la cantidad de salidas que necesita para mantener la tasa. Para esto, le notifica al bloque en la variable noutputitems, la cantidad de “tandas” de datos que la instancia actual del bloque necesita devolverle al motor de procesamiento para mantener la tasa.

La variable ninputitems, mantiene el control de la cantidad de datos que hay en cada uno de los puertos de entrada. Y utilizándola como variable, podemos especificar cuantos datos de entrada precisamos consumir para obtener un dato en el puerto de salida.

Esta operación depende bastante de la naturaleza del bloque, por ejemplo, en los bloques síncronos, se sabe de antemano que la tasa de muestras se mantendrá constante, por lo que alcanza con especificar el tipo y la cantidad de muestras que atravesarán el bloque en una ejecución.

Los bloques de tipo general, son más complicados, pues es necesario especificar explícitamente la relación entre la cantidad de muestras a la entrada y a la salida.

Este control de flujo se realiza desde un método, precargado en la clase con la que se crea el bloque, denominado forecast.

4.3. Hardware

Universal Software Radio Peripheral (USRP) es una línea de plataformas de hardware para SDRs diseñados y comercializados por la empresa Ettus Research. Gran parte de la arquitectura de estos equipos es de código abierto, y puede ser descargada desde la web de la empresa. Con estos equipos se logra poner en funcionamiento soluciones de radio definida por software por costo accesible, ya que la gama de precios de los equipos de la compañía oscila en un entorno de 1000 dólares.

Tambien existen otros proveedores de menor renombre, con equipos que presentan un desempeño un poco más ruidoso, como es el caso del HackRF que comercializa Great Scott Gadgets [20], en el entorno de los 300 dólares. Durante el desarrollo de gr-isdbt-tx experimentamos con el modelo HackRF One, con magros resultados, por lo que no consideramos pertinente detallar más sobre el mismo. Aunque para otras aplicaciones de SDR, se ha probado que funciona de manera correcta.

En líneas generales, se componen de una placa madre en la que conviven varios subsistemas, como lo son un generador y sincronizador de señales de reloj, conversores analógico digitales y FPGA, que se encargan de realizar tareas de procesamiento de señal en banda base. Se complementa con una placa modular denominada "daughterboard" que realiza tareas como filtrado, acondicionamiento de señal y modulación las señales en frecuencias de hasta 6 GHz.

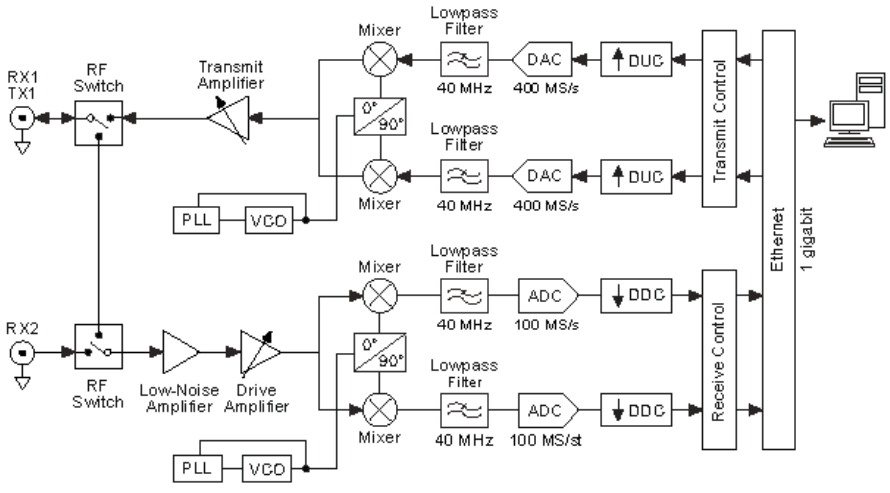

Figura 4.1: Arquitectura general de un equipo USRP.

Durante este proyecto, trabajamos con el modelo USRP B100 de Ettus Research. A continuación presentamos una breve reseña del equipo.

4.3.1. USRP B100

En este modelo, las muestras llegan al equipo a través de una interfaz USB 2.0 y tiene una capacidad para transmitir en velocidades de hasta 128 MS/s (Mega Samples per Second), cuando se trabaja con muestras de 14 bits. Esta limitacion del equipo viene del conversor DAC con el que convertiremos las muestras para transmitir, para trabajar en recepcion, el limite se da por el conversor ADC que soporta hasta 64 MS/s con muestras de hasta 12 bits. Por ultimo, el ancho de banda con el que se puede trabajar con el B100 es de hasta 8MHz, trabajando con muestras de 16 bits. [21]

Este modelo en particular esta actualmente discontinuado, pero su precio en el mercado rondaba los 700 dólares. Para el desarrollo del proyecto, recibimos en calidad de préstamo uno de estos equipos para las pruebas

Para poder levantar el equipo desde una PC, es necesario instalar el driver universal de Ettus, "USRP Hardware Driver". La mejor forma de hacer esto es instalarlo desde su repositorio oficial [22]. El driver es gratuito y de código abierto, y soporta todos los equipos de radio de Ettus.

Interfaz con GNU Radio

Para intercomunicar al equipo con el software de procesamiento de datos, es necesario incluir en el flowgraph los bloques del complemento gr-uhd (en las versiones actuales de GNU Radio ya vienen incluidos). El bloque sink de este complemento vincula al flujo de datos del software con el puerto receptor de datos del equipo, y permite especificar los parámetros de transmision como frecuencia central y ancho de banda.

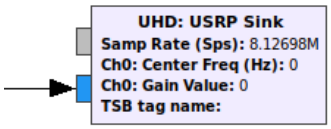

Figura 4.2: Bloque sink al equipo USRP

Los argumentos con los que funciona son la tasa de muestras, la frecuencia de destino en la que se desea transmitir y el valor de ganancia aplicado a la señal de entrada. Es importante verificar que los parámetros a seleccionar sean soportados por el hardware. En el caso de nuestro transmision, tiene una tasa de muestras objetivo de:

$$f_{IFFT} = \frac{portadoras}{T_s} = \frac{512}{63} \approx 8{,}127MHz$$

Para verificar que la tasa de muestras sea soportada por el transmisor, primero debemos verificar el tamaño de los datos. En nuestro caso, el tipo es *grc::complex*,

el cual es un tipo propio de GNU Radio, formado por una unión de dos variables *std::complex* cada una de 32 bits representadas en punto flotante. Es decir que cada una de las muestras de nuestro transmisor se compone de 64 bits.

Un aspecto a tener en cuenta con el transmisor, es verificar que la amplitud de las muestras esté normalizada, a modo de evitar un efecto de *clipping* a la salida, esto es, que todas las muestras superiores a 1, terminaran con la misma amplitud en recepción, lo que nos dejará una señal trunca en amplitud. Evitar este problema resulta sencillo, basta con agregar una ganancia a la salida del sistema controlada por una variable, y verificar que el factor sea el adecuado para que la señal no sobrepase la unidad.

Capítulo 5

gr-isdbt-tx: un transmisor ISDB-T implementado en GNU Radio

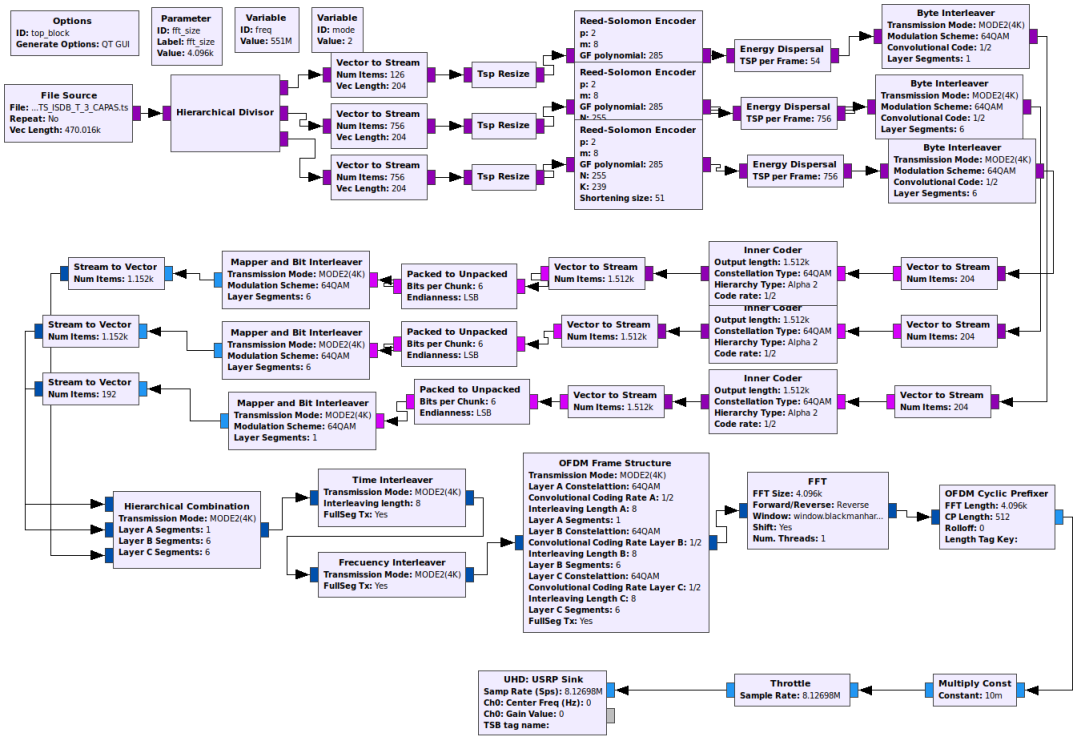

Figura 5.1: Flowgraph del transmisor gr-isdbt-tx.

En la figura 5.1 se muestra el diagrama de bloques que conforman el transmisor gr-isdbt-tx. En este capítulo repasamos las distintas funcionalidades que fueron implementadas para lograr la transmisión.

Los bloques de este capítulo están separados por funcionalidades y no por orden de aplicación en el flujo de datos establecido por la norma. Describir de esta

manera las funcionalidades desarrolladas nos permite comparar algunos bloques de funcionamiento similar y que a la hora de desarrollar necesitaron de soluciones de programación similares.

5.1. Obtención de los TS por capa

Como se explicó anteriormente, en un solo flujo de transporte coexisten paquetes pertenecientes a las distintas capas de transmisión, cada una con sus programas, sus tablas y sus propias características de transmisión.

Es necesario separar esta información al comienzo del transmisor, para procesar individualmente cada capa, afín de lograr mantener los datos de cada capa en un mismo flujo, pues como vimos, las capas pueden tener distintas modulaciones y distintos retardos.

Cada paquete tiene en su encabezado IIP, un parámetro denominado *layer_information*, que contiene información sobre la capa de transporte a la que pertenece. El parámetro ocupa los primeros cuatro bits del segundo byte del encabezado de cada paquete, y se decodifica según la siguiente tabla

Valor $b_7b_6b_5b_4$	**Capa**
0000	Null TSP
0001	Capa A
0010	Capa B
0011	Capa C

Tabla 5.1: Parámetro indicador de capa en el IIP.

El primero de los bloques de gr-isdbt-tx, es el bloque *Hierarchical Divisor*, es un bloque que se encarga de realizar esta tarea. Dado que conocemos de antemano la cantidad de TSPs que están contenidos en un cuadro múltiplex de nuestro BTS de fuente, definimos un bloque con un puerto de entrada y tres puertos de salida. En cada ejecución, se recorre un cuadro múltiplex, analizando para cada uno de los paquetes el parámetro *layer_information*, decodificamos a qué capa pertenece, y se enrutan según corresponda.

Entendemos que la implementación que presentamos de este bloque, existen cosas a mejorar. No las incluimos en este release del código por cuestiones de tiempo. De primera mano, si la cantidad de TSP por cuadro múltiplex fuese ser calculada en tiempo de ejecución, el transmisor soportaría cualquier BTS como fuente de datos. Hoy esos parámetros están fijos para el BTS de pruebas con el que trabajamos, y eso limita la robustez del transmisor. Obtener un BTS válido para realizar testing no fue tarea sencilla, puesto que buscamos uno que tuviese datos válidos para las tres capas de trabajo de la norma, y los canales nacionales actualmente no están emitiendo One-Seg. Discutiremos esta carencia junto con otras mas a fondo en el capítulo 7, cuando analicemos el trabajo a futuro.

Luego del bloque *Hierarchical Divisor*, los TSP de 204 bytes se encaminan hacia el codificador Reed Solomon. El bloque que se utilizó para este propósito,

espera en la entrada paquetes de 188 bytes, por lo que implementamos un bloque llamado *TSP Resize* que se encarga de remover los últimos 16 bytes antes de que los paquetes lleguen al codificador. Es importante mencionar que los datos que se eliminan en este bloque, no tienen en este momento del flujo información relevante, ya que en esa posición están los encabezados IIP que el codificador Reed Solomon sobrescribe con la codeword generada, según lo establecido en la norma.

5.2. Codificaciones de Canal

5.2.1. Reed Solomon

La norma ISDB-T utiliza una implementación similar a la de DVB-T del codificador Reed Solomon. En cada TSP se sustituyen los últimos 16 bytes del IIP por la palabra de redundancia de un código RS(204,188). Para lograr esta distancia, lo que se hace es llevar el payload hacia un tamaño mas grande, agregando 51 bytes en 0 al comienzo del TSP. Luego, el nuevo payload de 239 bytes, se inyecta en un codificador RS(255,239) para obtener los 16 bytes de codeword.

El cambio de tamaño se realiza principalmente para utilizar un algoritmo más eficiente. Una vez obtenida la palabra código del mismo, se remueven los primeros 51 bytes nulos, y obtenemos el código RS(204,188) que entra en el tamaño de un TSP, y además es capaz de corregir errores hasta en 8 bytes.

Para la implementación de esta funcionalidad en ISDB-T, no fué necesario desarrollar ningún bloque. Dentro de los complementos de GNU Radio está el paquete *gr-dvb* [23] (originalmente un proyecto de terceros, que en 2015 fué incorporado al repositorio oficial de GNU Radio), que contiene los bloques necesarios para implementar tanto un transmisor como un receptor bajo la norma DVB. Al compartir el codificador Reed Solomon con ISDBT, bastó con utilizar el bloque de gr-dvb.

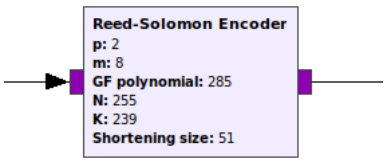

Figura 5.2: Bloque Reed Solomon Encoder de gr-dvb.

En la figura 5.2, mostramos el bloque y la configuración paramétrica del mismo, tal cual se implementó en gr-isdbt-tx. Los parámetros **p**, **m** y **GF polynomial** son los que configuran el polinomio generador que es la base del algoritmo. Para configurar el polinomio necesario para RS(255,239), debemos cargar en el bloque el polinomio:

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

5.2.2. Viterbi

Otro de los códigos de canal implementados por ISDB-T es el código Viterbi. Este código, es convolucional con puncturing, su código madre tiene una tasa de $\frac{1}{2}$ y constante $k = 7$, lo que permite obtener en salida datos codificados a cualquier tasa m/n en transmisión, sin aumentar la complejidad de la decodificación en recepción.

Esto sucede, porque tanto el lado transmisor como el receptor, conocen la llamada matriz de puncturing. En esa matriz, se especifica para cada tasa de código buscada los bits redundantes, que serán eliminados al transmitir para lograr la tasa deseada. Vale recordar que estos bits deben ser reingresados por el decodificador en recepción, pues de lo contrario aumentaría fuertemente la complejidad de la decodificación.

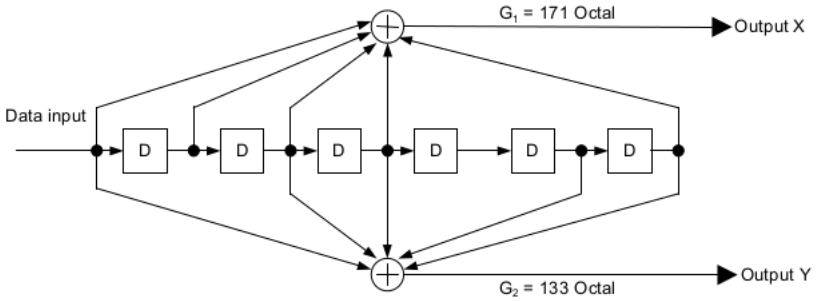

Figura 5.3: Circuito del código madre del Viterbi que implementa ISDB-T.

Agregar este tipo de códigos a la cadena de transmisión, aumenta la resistencia ante las perdidas y mantiene una tasa de bits constante, lo que colabora con el mantenimiento del sincronismo del sistema.

DVB utiliza el mismo código en su cadena de transmisión, con los mismos parámetros, por lo que en gr-isdbt-tx, no fué necesario realizar un desarrollo del bloque sino que reutilizamos el existente en gr-dvb.

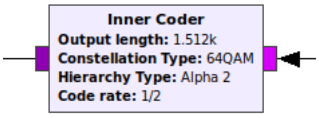

Figura 5.4: Bloque de gr-dvb que implementa el codificador Viterbi.

5.3. Dispersor de Energía

Es muy probable que los flujos de transporte contengan largas secuencias de unos o ceros. Esto puede presentar un problema a la hora de sincronizarse con el receptor, ya que generalmente, los cambios de flancos al medir los datos en

recepción suelen utilizarse para sincronizarse con el reloj de transmisión. Por otra parte, los patrones repetitivos en las cadenas de bits darían lugar a acumulaciones de energía en ciertos puntos del espectro, dejando espacios subutilizados en la banda. Para evitar estas cosas es que ISDB-T utiliza un dispersor de energía.

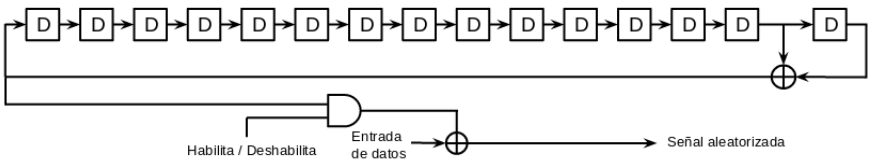

Figura 5.5: Circuito generador de la secuencia pseudoaleatoria.

El funcionamiento del dispersor consiste en la generación de una secuencia pseudo-randómica de bits. Esto se logra realizando un XOR de los bits de entrada contra un circuito que evoluciona constantemente. El circuito se inicializa con la palabra "100101010000000", que es conocida tanto por el transmisor como por el receptor. Para el byte de sincronismo, el XOR no se realiza, pues este byte es una referencia también para otros bloques y permanece incambiado, pero de todos modos el circuito se continúa actualizando. Para recuperar la secuencia original, el receptor utiliza exactamente el mismo circuito. Supongamos que $r = d \oplus g$ es un bit randomizado producto del XOR entre un bit de entrada y el bit del circuito generador. Si en recepción se utiliza el mismo circuito y está sincronizado, entonces el bit de-randomizado será $r' = r \oplus g = (d \oplus g) \oplus g$. Como la operación XOR es asociativa tenemos que $r' = d \oplus (g \oplus g) = d \oplus 0 = d$, y de esta manera se recupera el bit original. Es necesario que tanto el transmisor como el receptor estén sincronizados en la evolución de la palabra, pues de lo contrario los bits decodificados no serán los mismos que los originales. Para asegurarnos de eso, es que se reinicia el circuito generador en cada inicio de cuadro.

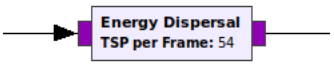

Figura 5.6: Bloque dispersor de energía de gr-isdbt-tx.

En el release actual de gr-isdbt-tx, el bloque dispersor de energía solicita como parámetro de entrada la cantidad de TSPs que conforman el frame de cada layer, para poder reiniciar con éxito el circuito generador. Durante el desarrollo del código, este parámetro fue de mucha ayuda ya que no nos resultó sencillo sincronizar transmisión y recepción, y además sirvió como herramienta de debugging en distintas pruebas que se realizaron.

Como trabajo pendiente, al igual que en el divisor jerárquico, nos proponemos eliminar este parámetro y obtener la cantidad de TSP por frame y por layer en tiempo de ejecución, de nuevo, para lograr más flexibilidad en cuanto a las fuentes de datos que pueda tomar este transmisor.

5.4. La modulación

Para resolver la modulación en nuestro transmisor, decidimos crear un solo bloque que resuelva en conjunto los problemas de interleaving de bits y modulación. Esto resultó bastante conveniente, pues bastó con crear una variable que contenga un selector de modulación, y en base a él se crean una serie de colas, cuyos tamaños responden a lo explicado sobre interleaving en 5.7.

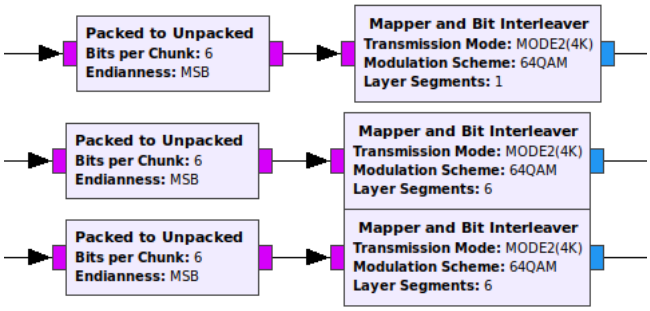

Figura 5.7: Implementación de la modulación e interleaving a nivel de bits en gr-isdbt-tx.

El problema que nos encontramos en esta etapa, fué el de particionar la información en un byte, pero conservar el resto. Un ejemplo claro de esto se dá cuando la modulación es *64-QAM*, pues cada símbolo en este caso se construye con 6 bits. Ahora, el parseo de datos entre bloques, GNU Radio lo hace encapsulados en datos de tipo *byte*. Por lo tanto, necesitábamos una forma de resolver el problema de separar 6 bits utiles de un dato, y guardar los 2 siguientes para el próximo símbolo. En principio, no parecería ser un problema tan grave, pero podría pasar que los 2 bits que nos quedan pendientes, se correspondan con un símbolo que se construirá en una próxima llamada al método modulador del objeto "bloque mapper". A nivel de programación, esto implica que la memoria volátil de la instancia del objeto, se borra, por lo que habría que utilizar algún almacenamiento que permanezca en memoria estática, y actualizarla en cada símbolo que se procesa.

Recorrer ese camino nos pareció además de muy laborioso, poco eficiente en términos de CPU. Encontramos la solución en un conjunto de bloques propietarios de GNU Radio, que lo resuelven de manera eficiente. Los bloques *Packed to Unpacked* nos resuelven el problema. Basta con tener en cuenta a la hora de utilizar el transmisor, definirle el parámetro de *Bits per Chunk* de forma consistente con la modulación que se este utilizando.

Modulación	**Bits por símbolo**
QPSK	2
16-QAM	4
64-QAM	6

Tabla 5.2: Bits por símbolo según modulación.

A la salida de los bloques *Packed to Unpacked*, tendremos un byte completo de 8 bits, pero rellenado con la cantidad de bits requerida por la modulación, comenzando por el bit mas significativo. Esto nos permite normalizar la recepción de datos del bloque modulador, para cada caso, sabremos perfectamente como esta compuesto el byte de datos.

Una vez allí, alcanza con crear una cola con el retardo correspondiente, y enrutar los bits hacia la misma de forma ordenada. La sección de modulación del bloque, funciona de la siguiente manera.

Se obtiene de la cola los bits a procesar, y se arma, manteniendo el orden especificado en la norma, una palabra de N bits, donde $N = 2, 4, 6$ según corresponda en la tabla 5.2. Una vez formada la palabra, se re interpreta como un número decimal, y se busca en un arreglo el complejo que se corresponda con la palabra recién formada.

5.5. El uso de los entrelazamientos

5.5.1. Entrelazamiento frecuencial

El entrelazamiento frecuencial consiste en permutar las portadoras de un símbolo OFDM en el dominio de la frecuencia. Dada la característica de ISDB-T de utilizar un gran número de portadoras, ocurre que frente a canales selectivos en frecuencia se pueden ver afectadas una cantidad importante de portadoras consecutivas, con lo cual la capacidad de corrección de los códigos resultaría superada.

Realizando un entrelazamiento frecuencial las portadoras que pudieran verse afectadas en un canal selectivo son desentrelazadas en recepción y, por lo tanto, los errores que pudieran haber quedan distribuídos facilitando la tarea de los códigos correctores.

El procedimiento del entrelazamiento en ISDB-T se divide en dos etapas: el *enterlazamiento inter-segmentos*, y el *entrelazamiento intra-segmentos* .

Primero se separan los segmentos en tres grupos: los destinados a recepción parcial (*one-seg*), los que utilizan *modulación coherente*, y por otro lado los que utilizan *modulación diferencial*. Como en el caso de *gr-isdbt-tx* sólo se utiliza modulación coherente, los caminos que pueden tomar los segmentos son únicamente dos.

Posteriormente viene la etapa de entrelazamiento inter-segmentos. Precisamente consiste en intercalar las portadoras entre los segmentos que comparten la modulación. Para el segmento destinado a la recepción parcial este proceso no tiene sentido, por lo cual pasa directamente a la etapa de enterlazamiento intra-segmento.

El entrelazamiento intra-segmento se realiza permutando las portadoras de cada segmento entre sí. Se comienza por rotar las portadoras de acuerdo a la siguiente expresión:

$$S_{i,k}^{rotada} = S_{(k+i)modC,k} \tag{5.1}$$

Donde i representa el número de portadora dentro del segmento y puede tomar valores entre $[0, C-1]$, con $C = 96 \times 2^{modo-1}$ la cantidad de portadoras de datos en cada segmento; k es el número de segmento y toma valores entre cero y la cantidad de segmentos que utilizan esa modulación.

Luego se realiza una aleatorización que está determinada por *Look Up Tables* que dependen del modo de transmisión y puden consultarse en la norma [2].

5.5.2. Entrelazamiento temporal

El entrelazamiento temporal consiste en distribuír los símbolos complejos entre distintos símbolos OFDM. Esta distribución o entrelazamiento se realiza para cada portadora, es decir que se realiza en el dominio del tiempo.

Esta técnica actúa como mecanismo de protección frente a ruidos impulsivos que típicamente se caracterizan por una corta duración en el tiempo. La profundidad o largo del entrelazamiento temporal es el parámetro que rige este proceso de entrelazado y puede ser seteado de manera independiente para cada capa. Está íntimamente ligado a la dispersión temporal que se realiza en los símbolos; a mayor profundidad, los símbolos de una misma portadora son retardados un tiempo mayor.

La figura 5.8 describe esquemáticamente el mecanismo de entrelazamiento. Consiste en aplicar un retardo distinto para cada símbolo dentro de un mismo segmento; los segmentos de una misma capa son retardados de igual manera según la profundidad de entrelazamiento elegida.

La implementación de estos retardos en *gr-isdbt-tx* es llevada a cabo por medio de colas en las que, de manera secuencial, se empujan los símbolos que van arribando correspondientes a las distintas capas jerárquicas. Desde el otro extremo se extrae un símbolo de cada cola también de manera secuencial.

Para una capa jerárquica con profundidad de entrelazamiento I_L, que puede ser 0, 1,2 ,4, 8, 16 según el modo, los retardos estipulados por ISDB-T para cada uno de sus segmentos se define de la siguiente manera:

$$q_L(i) = I_L \times m_i = I_L \times ((i \times 5)\ mod\ 96) \tag{5.2}$$

Donde $q_L(i)$ representa el tamaño de las colas o búfers; m_i es definido por el estándar como $m_i = (i \times 5)\ mod\ 96$, para $i = 0, 1, .., n_c$ con $n_c = 96$, 192, o 384 según el modo de transmisión.

El proceso de entrelazamiento introduce un retardo que debe ser ajustado de manera que el retardo total introducido por el proceso sea un múltiplo de cuadro OFDM. Para ello *gr-isdbt-tx* conforma colas más largas de las especificadas en el esquema de la figura 5.8 logrando así un retardo total de un cuadro OFDM.

Este proceso genera un atraso que en símbolos OFDM está dado por la siguiente expresión:

$$95 \times I_L\ mod\ 204 \tag{5.3}$$

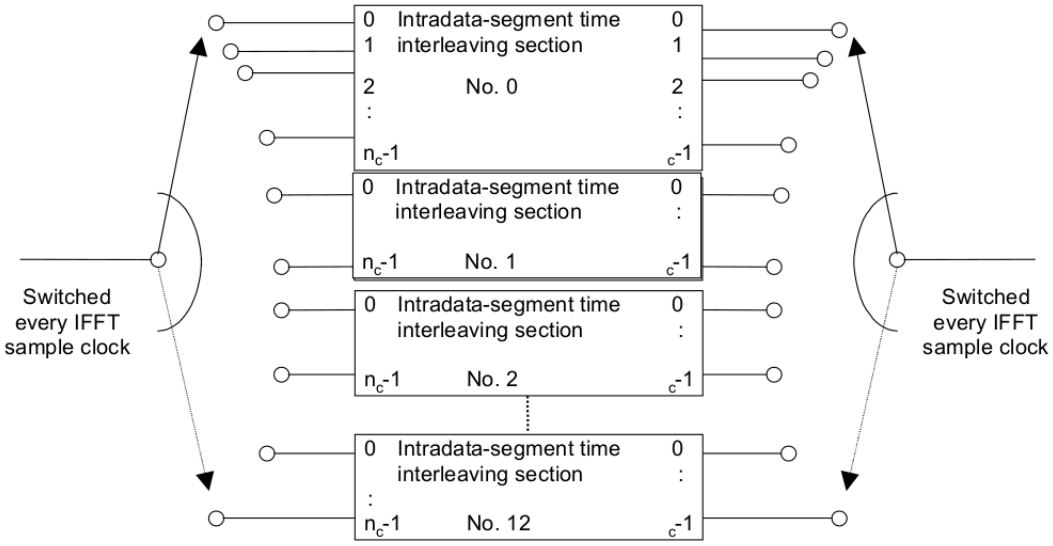

Figura 5.8: Mecanismo de entrelazamiento temporal.

Entonces para completar un cuadro OFDM hacen falta $d_I = 204 - (95 \times I_L \ mod \ 204)$ símbolos OFDM. Por lo tanto la cantidad de cuadros OFDM de retardo será $N_L = (95 \times I_L + d_I) \ mod \ 204$.

La implementación en el transmisor de este ajuste de atraso consiste en empujar los símbolos que arriban al bloque a cada una de las colas de tamaño $I_L \times m_i + d_I$. Del otro extremo de las colas se toman los símbolos secuencialmente y se tiene la secuencia entrelazada. Este mecanismo, al igual que otros entrelazamientos, hace que en el arranque del sistema se tengan datos espurios en los registros.

5.6. Entrelazamiento de bytes

El bloque de entrelazamiento de bytes se utiliza para aumentar la capacidad correctora de errores del código Reed Solomon. Como ya se explicó en el capítulo 3, esto se hace implementando un sistema de colas de retardos de largos variables, por las que se van enrutando los bytes uno a uno.

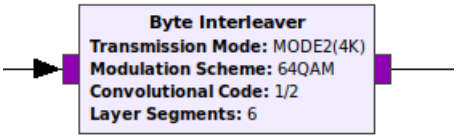

Figura 5.9: Bloque entrelazamiento de bytes.

A nivel de código, esto se implementó mediante un vector de 12 elementos. Cada uno de los elementos se define como una cola de bytes, de tamaño $17 * i$

siendo i el número de elemento. La cola se crea en el comienzo de la ejecución, cuando se crea una instancia del bloque. Luego, en régimen permanente, los bytes simplemente llegan en orden y van siendo asignados uno a uno a su cola correspondiente.

Como se vió anteriormente, es necesario realizar un ajuste de atraso para que el retardo total del bloque sea igual al período de un cuadro OFDM. Exploramos distintas alternativas para implementar esta funcionalidad, y la que nos pareció más eficiente fué crear un elemento extra en la cola de vectores, que contenga una cola más larga, correspondiente al ajuste de atraso. El bloque solicita como entrada los parámetros para calcular la cantidad de bytes de atraso, como se ve en la figura 5.9, y crea la cola del elemento 13 del vector, del tamaño necesario, que recordamos, está dado por la siguiente ecuación:

$$D_{bytes} = \frac{204 \times m_L \times FEC_L \times 96 \times 2^{modo-1}}{8 \times 204} - 11 \tag{5.4}$$

Donde m_L y FEC_L son el esquema de modulación y la tasa de código convolucional utilizados para la capa jerárquica L-ésima. Al final del cociente, se le restan los 11 bytes de delay que son agregados por el multicamino de las colas del entrelazado.

Un detalle que no está definido en la norma, es la equivalencia entre esquema de modulación utilizado y el valor del parámetro m_L a insertar en la ecuación.

Modulación	m_L
QPSK	2
16-QAM	4
64-QAM	6

Tabla 5.3: Asignación de valor de m_L según modulación

Por lo tanto, el flujo de los bytes en el bloque queda constituido por dos colas en serie, una de atraso, por la que circulan todos los bytes ni bien entran al bloque, y otra determinada por el entrelazamiento tal cual está establecido en la norma.

El efecto total del bloque es el atraso y entrelazamiento que define el estándar. Pudimos probar durante el desarrollo de gr-isdbt-tx, que si el valor del atraso no es el correcto, no sólo se pierde el sincronismo con el desentrelazador de recepción, sino que también se pierde el sincronismo con el reseteo del PRBS del dispersor de energía. Los TSP entonces vuelven a recorrer el dispersor, pero contra un circuito distinto al recorrido en transmisión, ocasionando que los datos a la salida del bloque estén completamente cambiados. Cuando esto sucede, los datos siguen su camino hacia el decodificador Reed Solomon con codewords inválidas, que son interpretadas como paquetes con una cantidad de errores superiores a la capacidad del código. En ese caso, el bloque Reed Solomon entiende que los paquetes están muy corrompidos para ser recuperados y a la salida del receptor, no se obtienen datos.

5.7. Entrelazamiento de bits

En gr-isdbt-tx, implementamos el entrelazamiento de bits en conjunto con el bloque modulador. Esto significa que el bloque tiene a cargo dos tareas, primero entrelazar los bits que le ingresan provenientes de una determinada capa jerárquica dando lugar a un nuevo flujo de bits entrelazados. Y luego agrupar estos bits para formar símbolos complejos según la modulación que tenga la capa.

El proceso de entrelazamiento se lleva a cabo de forma idéntica al caso de bytes, utilizando un vector de colas, pero en este caso, de booleanos en lugar de bytes. La diferencia es que en este caso, la cantidad de colas a crearse, y el tamaño de las mismas, dependen de la modulación seleccionada por el usuario.

Al igual que en el caso anterior, entrelazar utilizando este mecanismo de colas de retardos variables implica agregar un retardo total correspondiente al camino más largo que deben atravesar los bits, es decir un retardo de 120 símbolos. Según la modulación que utilice la capa, estos 120 símbolos se traducen en un mayor o menor retardo en el tiempo; para QPSK corresponde a esperar que ingresen 120×2 bits en la entrada, mientras que para 64QAM deben pasar 120×6 bits.

Con el objetivo de tener un mismo retardo para lograr sincronismo en todas las capas se realiza un *ajuste de atraso*. En este caso, los retardos deben ser múltiplos de un símbolo OFDM; para el entrelazamiento de bits el retardo propio del proceso de entrelazamiento sumado al ajuste de atraso debe totalizar dos símbolos OFDM.

El tamaño de un símbolo OFDM corresponde a:

$$N_{OFDM} = m_L \times N_L \times 96 \times 2^{modo-1} \quad \text{bits} \tag{5.5}$$

N_{OFDM} depende de la cantidad de segmentos utilizados N_L, del modo de transmisión y de la modulación de la capa, m_L, que se obtiene según la tabla 5.3;. La solución implementada en este bloque es distinta a la del entrelazador de bytes. Aquí, el retardo total se obtiene incrementando los tamaños de las colas de manera que el retardo total sea de dos símbolos OFDM para cada capa. Por lo tanto para tener el retardo deseado los tamaños de las colas deben ser:

$$Q_i = q_i + \frac{2 \times N_{OFDM} - 120 \times m_L}{m_L} = q_i + N_L \times 96 \times 2^{modo-1} - 120 \tag{5.6}$$

Una vez que son empujados los bits en cada cola, desde el otro extremo se extrae un bit por cola y se mapean en un símbolo complejo de acuerdo a lo establecido en la norma. En la figura 5.10 se muestra cómo se realiza este mapeo para el caso de la modulación 64QAM.

Para resolver el mapeo, se implementaron arrays de números complejos. Una vez obtenidos los bits a modular, se ordenan y se convierten a un entero. Ese entero es la posición en el array del numero complejo que se mapea en la constelación, que se copia a la salida.

Por último los complejos deben ser normalizados según un factor de normalización que se presenta en la tabla 5.4.

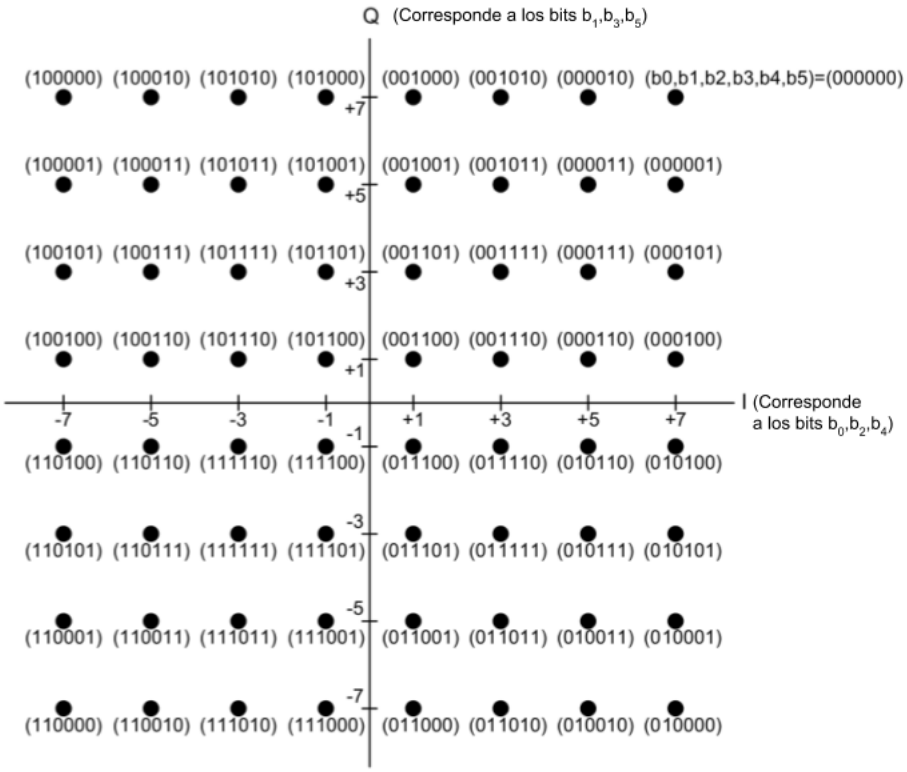

Figura 5.10: Mapeo de la constelación 64QAM.

Modulación	**Factor de normalización**
QPSK	$1/\sqrt{2}$
16QAM	$1/\sqrt{10}$
64QAM	$1/\sqrt{42}$

Tabla 5.4: Factores de normalización para los distintos esquemas de modulación.

5.8. Formación de cuadros OFDM

El cuadro OFDM es una estructura de datos que agrupa los datos de carga útil, portadoras piloto y señales de control. Tiene todo lo necesario para que un receptor compatible con ISDB-T sea capaz de decodificarlo en flujos MPEG y reproducir la información.

Se trata de un conjunto de 204 símbolos OFDM, cada uno de ellos conformado por $13 \times 108 \times 2^{modo-1}$ símbolos complejos. Una de las particularidades del cuadro, es que su estructura tiene posiciones fijas, donde siempre viaja información con las características de la transmisión, y posiciones móviles, en las que viajan portadoras que estiman el efecto que tiene el canal sobre los datos, para poder, con esa información, robustecer al sistema.

Para cada segmento, la estructura del cuadro tiene su forma particular. En la

figura 5.11 presentamos un ejemplo para una modulación coherente, basada en un modo de transmisión 1.

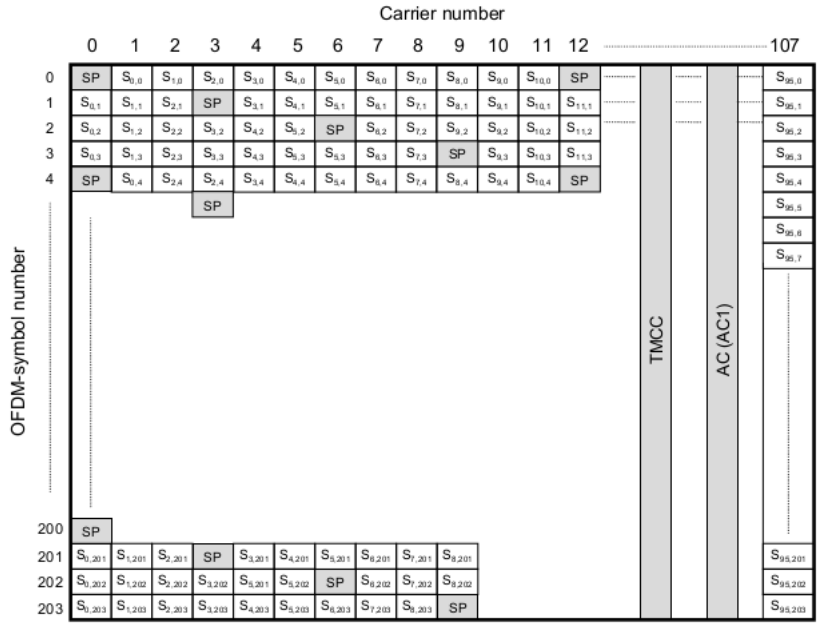

Figura 5.11: Estructura de cuadro OFDM para un segmento, con modo de transmisión 1.

A nivel de código, la solución que se implementó en este proyecto para el bloque, es bastante larga. Separamos el bloque en dos momentos, el transitorio y el régimen. Durante el transitorio, esto es, cuando se llama al bloque por primera vez y se crea en la memoria una instancia del mismo, se inicializan todas las variables. Entre ellas están, la palabra TMCC para todo el cuadro, los valores iniciales de la secuencia PRBS que rige el movimiento de las portadoras SP y algunas variables generales que mantienen conteos de frames para seguimiento.

La implementación de la TMCC se puede desarrollar en dos etapas. En primera instancia se toman los valores seteados en el bloque `OFDM Frame Structure` y a partir de ello se codifican los primeros 122 bits de la TMCC como se indica en la tabla 5.5. Luego, en una segunda etapa, se lleva a cabo el proceso de generación de los bits de paridad. Este proceso es tal cual se describió en la sección de Códigos Cíclicos; en el caso de la TMCC los coeficientes del polinomio $m(x)$ están dados por los bits B_{20} a B_{121}, que son los que estarán protegidos por el código. Ambas partes, transmisor y receptor, conocen el polinomio generador $g(x)$ definido por el estándar ISDB-T como:

$$g(x) = x^{82} + x^{77} + x^{76} + x^{71} + x^{67} + x^{66} + x^{56} + x^{48} + x^{40} + x^{36} + x^{34} + x^{24} + x^{22} + x^{18} + x^{10} + x^4 + 1 \tag{5.7}$$

Con esto es posible calcular el polinomio de paridad que será

$$r(x) = resto\left\{\frac{x^{(n-k)}m(x)}{g(x)}\right\} \tag{5.8}$$

La implementación de la división de polinomios utilizada se basa en la *división sintética* de polinomios. Este método permite realizar la división utilizando operaciones de XOR, reduciendo enormemente el tiempo de ejecución del algoritmo.

Bit	**Descripción**	
B_0	Referencia para la modulación diferencial	
$B_1 - B_{16}$	Señal de sincronismo (w_0, w_1)	
$B_{17} - B_{19}$	Id. de tipo de segmento (diferencial o síncrono)	
$B_{20} - B_{21}$	Identificación del sistema (00 este estándar)	
$B_{22} - B_{25}$	Indicador de cambio de parámetro en la transmisión	
B_{26}	Flag para difundir alarma de emergencia	
B_{27}	Información actual	Flag de recepción parcial
$B_{28} - B_{40}$		Parámetros de transmisión capa A
$B_{41} - B_{53}$		Parámetros de transmisión capa B
$B_{54} - B_{66}$		Parámetros de transmisión capa C
B_{67}	Información próxima	Flag de recepción parcial
$B_{68} - B_{80}$		Parámetros de transmisión capa A
$B_{81} - B_{93}$		Parámetros de transmisión capa B
$B_{94} - B_{106}$		Parámetros de transmisión capa C
$B_{106} - B_{109}$	Corrección de corrimiento de fase (para radiodifusión)	
$B_{110} - B_{121}$	Reservados	
$B_{121} - B_{203}$	Paridad	

Tabla 5.5: Asignación de bits de la TMCC information.

La información que transporta la TMCC es crítica, y debe ser transmitida con una mayor confiabilidad que el resto de las señales. Al momento de ser insertada la TMCC el resto de las señales ya han sido codificadas en cuanto a protección contra errores. Esto es así porque codificar la TMCC con códigos cíclicos acortados reduce el tiempo de decodificación y la complejidad en el receptor, con lo cual se opta por transmitir la TMCC en varias portadoras a lo largo del cuadro OFDM. La recepción de la TMCC requiere una relación *Carrier-to-Noise* (C/N) menor que el resto de las señales, con lo cual es normal que se pueda recibir la TMCC en escenarios menos favorables.

En el caso general de un llamado a operar del bloque, la escritura se diseñó organizada por símbolos OFDM. En cada llamado al bloque, se completará un símbolo genérico. Algunas estructuras de control mantendrán el control de los cambios de palabras de control y otras variables. Los parámetros genéricos de transmisión que dependen de parámetros como el modo y las constelaciones de cada capa, se resuelven en el transitorio.

La cadena de sucesos de un llamado genérico es la siguiente. Primero se inicializa un vector de complejos en salida con el tamaño suficiente para todas las portadoras y los intervalos de guarda. Luego, se entra en una estructura de control que se encargara de repetir el proceso de escritura de símbolo OFDM cuantas veces el core de GNU Radio considere necesarios, a través del valor de *noutputitems*. Dependiendo del modo en el que se trabaje, se llama para cada segmento una función que se encarga de rellenar el segmento, según sea el modo de trabajo.

En caso de que se este trabajando en modo One-Seg, agregamos en este bloque un booleano que, en caso de estar en true, limita el rellenado de segmentos al segmento 0.

En el caso general, las funciones de llenado de segmento se llaman para todos los segmentos, en el orden lineal que tendrán a la salida, el mismo que ya vimos en el capítulo 3.

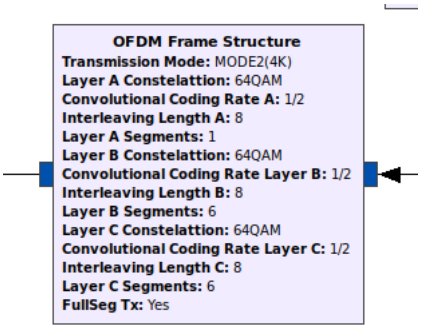

Figura 5.12: Bloque de conformacion de cuadro OFDM.

Las funciones de llenado de segmento, toman como parámetro de entrada el número de segmento que se va a escribir. Con este numero, encuentran en función de su posición en el cuadro, los índices en donde van las portadoras auxiliares y los datos. Luego iteran punto a punto por todas las portadoras del segmento, asignando según corresponda. En el caso de los datos, se copian simplemente de la entrada. Para el caso de las portadoras auxiliares, cada una tiene una funcion que almacena el estado de las mismas, y con ayuda de los índices de posición y de símbolo dentro del cuadro OFDM, evolucionan los datos dinámicos de las mismas y escriben en el segmento el dato correspondiente.

5.9. La transformada de Fourier

Una vez formado el cuadro OFDM, de forma correcta con todas las portadoras de datos y auxiliares completas, es el momento de pasar la señal al dominio del tiempo, para agregarle el prefijo cíclico y ponerla en el aire.

Utilizamos para esto uno de los bloques de GNU Radio que implementan el algoritmo *Fast Fourier Transform*. Este algoritmo, que ya fué mencionado en el documento, permite realizar de forma rápida y suficientemente aproximada de

la transformada de Fourier de una señal digital, siempre y cuando se maneje un tamaño de muestras que sea múltiplo de una potencia de 2.

Figura 5.13: Bloque que implementa el algoritmo FFT.

El parámetro *Reverse* del bloque, indica la dirección en la que se realiza la transformación. Dado que venimos trabajando en el dominio de la frecuencia y vamos a pasar al dominio temporal, debemos ejecutar el algoritmo en el sentido inverso.

5.10. El prefijo cíclico

Para sincronizar los relojes de transmisor y receptor, y eliminar de forma suficiente la interferencia intersimbólica, es que en ISDBT se utiliza un prefijo cíclico. Esto es, las primeras N muestras de la señal, se copian tal cual están hacia el final de la señal. Esto hace que, al calcular la autocorrelación de la señal en recepción, sea posible detectar de forma perfecta el comienzo de la señal, así como los parámetros de transmisión en caso de que sean desconocidos.

Se repasó el fundamento teórico detrás de esta práctica en el capítulo 3, en caso de buscar profundizar en la misma, una clara explicación de estos conceptos está en [7]. No entraremos en más detalles de este proceso para no ser repetitivos.

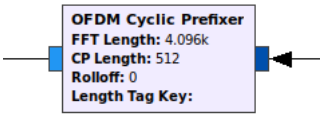

Figura 5.14: Bloque que implementa prefijo cíclico.

Este bloque tampoco es desarrollo propio del proyecto gr-isdbt-tx. Si bien escribimos un bloque que realiza esta funcionalidad (a nivel de programación se resuelve con un memcpy) preferimos dejar la implementación existente, que funciona muy bien y está validada por la comunidad de usuarios de GNU Radio.

De todos modos, la implementación de este bloque es un problema que los tutores nos plantearon al inicio del proyecto, y es interesante como primer acercamiento al entorno de desarrollo de GNU Radio. La lógica detrás del funcionamiento del bloque es simple, por lo que el kid de la cuestión está en manejar debidamente el movimiento de muestras entre el bloque y el core del software.

5.11. La transmisión desde USRP

Como ya se discutió en secciones anteriores, para realizar la transmisión de la señal a través del equipo USRP, es necesario utilizar el bloque sink del complemento *gr-uhd*. Siempre que se mantengan las condiciones para la buena transmisión que se enumeraron en el capítulo 4, es esperable tener a la señal en el aire sin mayores problemas. Algo que hay que tener en cuenta es que la tasa de muestras que recibe el USRP, por su arquitectura, debe ser un divisor de 64MSamples/s. Por este motivo es necesario utilizar un bloque `Rational Resampler` que realice el ajuste. Con un bloque `Rational Resampler` de parámetro 63/64, se tiene que $8{,}127MSps \times 63/64 \approx 8MHz$.

Como consideraciones generales, recomendamos verificar que el espectro a utilizar esté disponible, de forma de no solaparse con la señal de alguno de los canales comerciales, que transmiten a alta potencia y podrían causar problemas para decodificar nuestra señal.

Durante el desarrollo del transmisor, antes de probar la transmisión contra un televisor comercial, una prueba bastante sencilla es la de transmitir hacia un archivo. Luego, podemos usar ese archivo como entrada de datos al receptor *gr-isdbt* para corroborar que la transmisión tenga todos los parámetros correctos. Disponer de todos los bloques del receptor para tomar medidas, es un excelente método de debugging.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 6

Evaluación del sistema

Para la evaluación del transmisor, nos pusimos como objetivo final lograr la transmisión exitosa hacia un televisor comercial homologado por el LATU. En ese camino, nos propusimos una serie de objetivos de más corto alcance.

Como primer paso, nos habíamos propuesto lograr la decodificación con *gr-isdbt* dentro del mismo flowgraph (eso es, una prueba del sistema punta a punta trasmitiendo en condiciones ideales). Luego, nos propusimos superar una situación en la que una PC ejecuta el transmisor, enviando la señal a través del USRP, y otra computadora hace lo propio con el receptor *gr-isdbt*. Esto le agrega a la prueba anterior, el desafío de la puesta en el aire y decodificación de la señal ruidosa. Y por último, nos planteamos el objetivo final, la decodificación contra un televisor comercial. En este capítulo desarrollamos estas pruebas, y el nivel de éxito que alcanzó cada una.

En virtud de poder asegurar de forma cuantitativa si se cumplen o no esos objetivos, es necesario contar con herramientas que permitan medir de alguna manera la calidad de una señal digital. Existen varios indicadores de calidad en una transmisión digital, en este caso nos basaremos en dos de ellos que son ampliamente utilizados y proporcionan una medida concisa de la calidad de la señal. Es por eso que a continuación definiremos el *Bit Error Rate* (BER), y el *Modulation Error Rate* (MER).

6.1. Indicadores de calidad de una señal

6.1.1. Bit Error Rate (BER)

El estándar ISDB-T en su etapa de codificación de canal hace uso de una combinación de códigos de Reed-Solomon junto con códigos convolucionales, ambos están orientados a la corrección de errores de bits. En recepción se hace lo propio con los decodificadores Viterbi y Reed-Solomon, y ademas contamos con *gr-isdbt* que funciona como sistema de medición. Resulta importante cuantificar esos errores que se producen así como conocer la proporción de ellos que se han podido recuperar. Es de interés conocer tanto la cantidad de errores que hay antes y despues del Viterbi, y antes y despues del Reed-Solomon.

Se define la tasa de error de bits (BER) como el cociente entre la cantidad de bits erróneos que se recibieron y la cantidad de bits totales.

Para realizar una medición del BER hay que elegir un intervalo de tiempo determinado sobre el cual se va a promediar. En pruebas de aceptación de equipos comerciales se suelen realizar tests que duran varias horas, mientras que para pruebas de monitoreo los tests suelen ser de algunos minutos.

6.1.2. Modulation Error Rate (MER)

La tasa de error de modulación (MER) es una medida de la suma de todas las interferencias que afectan a una señal de TV digital. Para el cálculo del MER se registran todos los símbolos recibidos y sus desviaciones $(\delta I_j, \delta I_j)$ respecto a su posición teórica, donde el subíndice j corresponde al símbolo j-ésimo. Finalmente el MER corresponde al cociente entre la suma de los módulos de los símbolos teóricos sobre el módulo de la suma de los vectores de error formado por los $(\delta I_j, \delta I_j)$. Es común encontrarlo expresado en escala logarítmica.

$$MER = 10 \log_{10} \left(\frac{\sum_{j=0}^{N-1} I_j^2 + Q_j^2}{\sum_{j=0}^{N-1} \delta I_j^2 + \delta Q_j^2} \right) \tag{6.1}$$

Tener un MER alto indica una buena calidad en la señal, para los equipos comerciales se considera un MER de 35dB como muy bueno. Las señales recibidas en un hogar a través de una antena instalada en el techo tienen un MER que se ubica entre los 20dB y 30dB aproximadamente.

El MER se puede medir de varias maneras según lo que interese considerar. Es común muy calcular el MER para cada una de las capas por separado, pero también es posible considerar todas las señales incluyendo las portadoras piloto. Tener en cuenta las portadoras piloto para el cálculo del MER puede provocar que el promedio esté un poco por encima del MER de las tres capas por separado.

6.2. Pruebas contra gr-isdbt

Considerando que realizamos la mayoría del desarrollo contrastando los bloques del transmisor contra los de gr-isdbt, no parecería en principio un desafío mayor lograr la decodificacion punta a punta. Pero en sí, lo es, y eso es parte de la potencia del desarrollo con código abierto, constantemente podemos estar evaluando nuestro desempeño desde el otro lado.

Como receptor, *gr-isdbt* ha sido sometido a una diversidad de pruebas con resultados satisfactorios, por lo tanto, podemos considerarlo como una simulación de un televisor comercial válida. Es en ese punto en el que comenzamos las pruebas. Dentro de un entorno controlado, el transmisor *gr-isdbt-tx* conectado a la entrada del receptor *gr-isdbt*.

Pruebas sobre el flowgraph

En este caso ideal, donde todo funciona dentro de un mismo flowgraph y no hay variabilidad de las muestras transmitidas, el transmisor se comporta de forma perfecta. Detectamos un consumo excesivo de CPU en algunos bloques, en parte debido a problemas de optimización del código. Resolver estos problemas es un desafío que nos planteamos como parte del trabajo a futuro.

Para la prueba sobre flowgraph se utilizó un BTS constituído por tres capas jerárquicas donde se destina un segmento para la capa A y seis segmentos para las capas B y C respectivamente, todas moduladas en 64QAM. Tanto la tasa de código convolucional como la profundidad del entrelazamiento temporal son las mismas para las tres capas y son de 1/2 y 8 respectivamente.

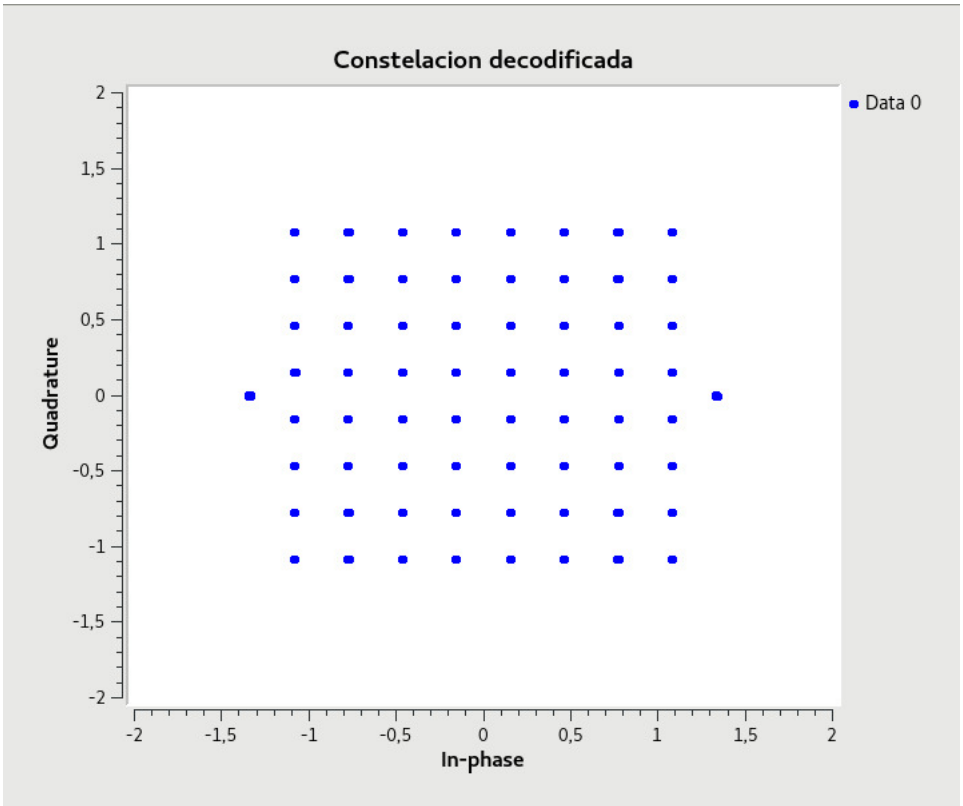

Figura 6.1: Constelación en recepción. Caso ideal.

Como se puede ver en la figura 6.1, el receptor *gr-isdbt* se sincroniza perfectamente con la señal transmitida, y en la constelación de recepción, vemos la misma constelación que en transmisión. Esto no debe sorprender, ya que en esta prueba, el canal está representado por un cable ideal. En la misma figura se puede apreciar las modulaciones 64QAM utilizada para las portadoras de datos y BPSK para las señales piloto.

Recibir la constelación perfecta en *gr-isdbt* significa varias cosas. Por un lado el cuadro OFDM está siendo conformado correctamente del lado del transmisor, por

otro lado, se está decodificando correctamente las portadoras piloto (SP, TMCC y piloto continuo). Además el receptor entiende perfectamente la información contenida en la TMCC, es decir que su contenido es el correcto y pasa satisfactoriamente los chequeos de paridad.

Podemos asegurar que esta prueba constituye un éxito dado que en el otro extremo de *gr-isdbt* se logró obtener cada una de las tres capas jerárquicas transmitidas sin ningún tipo de error.

Luego de esta primera prueba ideal, realizamos una segunda, aumentando la complejidad del canal. Se agregó un bloque `Channel Model` que simula un canal que agrega ruido blanco gaussiano y cuyos parámetros potencia de ruido, offset de frecuencia y respuesta al impulso son configurables. Se observó que el decodificador Reed-Solomon a partir de un cierto nivel de ruido deja de funcionar, sin embargo la TMCC continúa decodificándose correctamente incluso para niveles de ruido más altos. La tabla 6.1 resume los resultados obtenidos para distintos valores de la potencia de ruido N.

	N = 9	**N = 10**
BER Reed-Solomon	0.0065	0.49
BER Viterbi	0.051	0.064

Tabla 6.1: Valores de BER en los bloques correctores de errores.

Para valores $N < 9$ el BER es prácticamente el mismo, para cualquier N. Se observa que a partir de $N = 10$ el BER en el bloque Reed Solomon se dispara enormemente, alcanzando una tasa de la mitad de los bits erróneos. Esto pone de manifiesto la gran capacidad que tienen los códigos de Reed-Solomon pero si se supera el umbral de capacidad, su desempeño cae drásticamente. Para dar una idea de cómo resulta esto en términos de calidad de la imagen recibida, la figura 6.3 muestra los resultados para los casos en que $N = 9$ y $N = 10$.

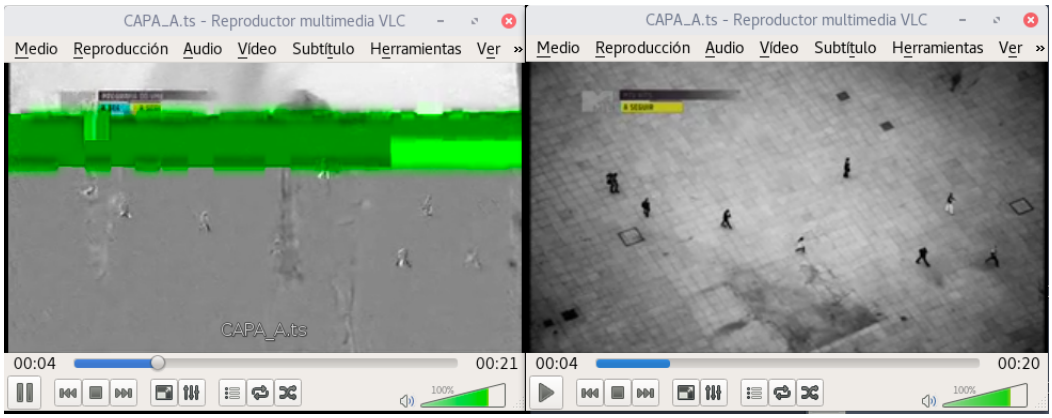

Figura 6.2: Calidad de la imagen recibida para $N = 10$ (izquierda) y $N = 9$ (derecha).

Pruebas en el aire

Realizamos la misma prueba con *gr-isdbt* como receptor, pero en lugar de simular un canal en GNU Radio, utilizamos el aire como medio, transmitiendo en una línea vista, a un metro de distancia.

En una primera instancia no se obtenía video a la salida de *gr-isdbt*, se veía la constelación decodificada de manera intermitente pero con demasiado ruido. Para esta prueba se estaba utilizando el USRP B100 como transmisor y el HackRF como receptor. Luego de realizar varias pruebas pudimos constatar que el HackRF introduce un offset de continua considerable lo cual no lo hace particularmete bueno para esta prueba. Una vez identificado este problema junto con otros bugs en el transmisor, fué posible la recepción exitosa con *gr-isdbt*.

6.3. Pruebas contra televisores comerciales

Las pruebas de validación con *gr-isdbt*, tanto dentro del framework de GNU Radio como en el aire, permitieron dar el siguiente paso para probar el transmisor contra televisores comerciales homologados para el estándar.

Esta prueba consistió en utilizar un laptop corriendo *gr-isdbt-tx* junto con un USRP B100 y una antena para la banda de TV digital. Desde el televisor se realizó un escaneo y la detección de nuestro canal fué inmediata. El televisor logró reproducir perfectamente el contenido que le estábamos transmitiendo, mostrando una excelente calidad de audio y video para cada una de las capas jerárquicas. Se probó transmitir sin línea de vista, moviendo la antena transmisora, y a una distancia de hasta 5 metros con excelentes resultados. Sin embargo era posible percibir un pequeño problema de fluidez de la imágen que no ocurrió en ninguna de las pruebas anteriores. Luego de varias pruebas y debugging, pudimos determinar que la causa provenía de un error en el modo de transmisión en que se estaba trabajando. Como vimos en capítulos anteriores, cada configuración jerárquica determina una cierta cantidad de paquetes que se transmiten por cuadro OFDM y por lo tanto un determinado bitrate. Una vez resuelto este problema la recepción en el TV comercial fué totalmente satisfactoria.

Esta prueba se repitió en el stand asignado para *gr-isdbt-tx* en Ingeniería De-Muestra 2018, en el que funcionó continuamente y sin perder la continuidad durante la totalidad del tiempo de exposición.

Como nota al pie, fuera del alcance de este documento, nos gustaría señalar la gran recepción por parte de los visitantes al stand. Nos encontramos con un público muy interesado y receptivo, y al final de la presentación, el proyecto fue galardonado como el mejor proyecto de Ingeniería Eléctrica en la categoría Señales y Telecomunicaciones.

6.4. Pruebas contra equipos analizadores de TV

Durante el desarrollo de las pruebas anteriores aparecieron una serie de bugs que se presentaron como un verdadero desafío. Los problemas de recepción inalám-

brica con *gr-isdbt* y las imperfecciones al momento de recibir en un TV comercial, motivaron el uso de sofisticadas herramientas comerciales para tener una idea de las posibles causas. Se tuvo la posibilidad de acceder a un equipo ETH de Rohde & Schwarz [24], propiedad de la Unidad Reguladora de Servicios de Comunicaciones, que es capaz de analizar señales de TV compatible con ISDB-T.

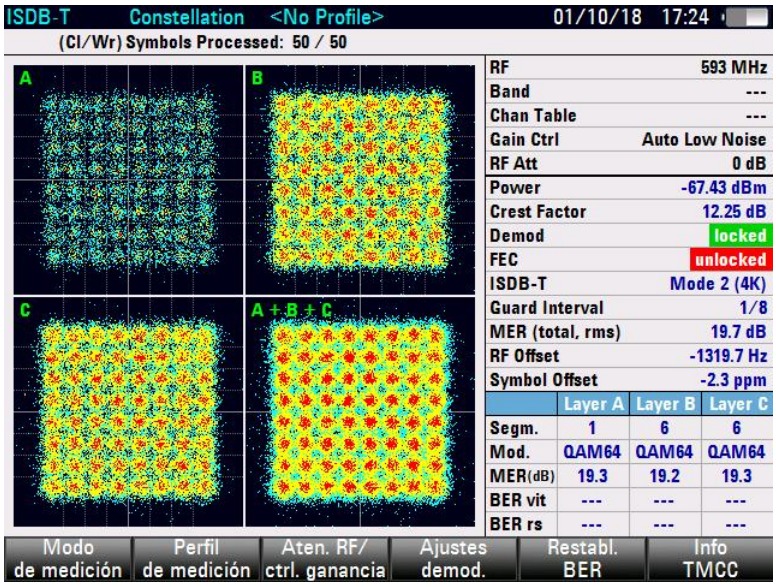

Figura 6.3: Constelación de las tres capas jerárquicas tomadas con el analizador ETH de Rohde & Schwarz.

Como vemos en la figura 6.3, con este instrumento se pudo observar claramente las constelaciones de las tres capas jerárquicas pero con demasiado ruido. También se observó que la señal presenta una variabilidad bastante grande del offset en frecuencia, que va desde los $400Hz$ a los $1300Hz$. Este parámetro para las señales comerciales de TVD presenta una estabilidad considerable, variando en el entorno de los $5Hz$. Los cristales de los equipos SDR utilizados tienen una precisión del orden de las 20ppm, lo cual hace que sea razonable esta variación encontrada.

Capítulo 7

Conclusiones y trabajo a futuro

Este proyecto tenia como objetivos, dos grandes ítems. Por un lado buscábamos implementar un transmisor de televisión digital basado en SDR, cuya validez estaría determinada por la capacidad de transmitir hacia un televisor comercial homologado por LATU. Por otro lado, también buscábamos echar luz sobre el conocimiento disponible en el país sobre la norma ISDB-T.

En cuanto al primer objetivo, entendemos que ha sido cumplido de forma satisfactoria. Hemos probado el transmisor contra varios televisores comerciales, de diferentes proveedores, y en todos los casos el equipo encuentra el canal transmitido de forma exitosa, y decodifica las tres capas procesadas de forma correcta.

Durante las pruebas encontramos diversos problemas que, mediante distintas estrategias, hemos podido solucionar. El problema de fluidez de reproducción del video fué quizás el más difícil de detectar, pero también su causa era tan sencilla como la incorrecta utilización de los parámetros. Cada prueba realizada aportó su cuota en el proceso de validación del transmisor. Como se mencionó anteriormente, dentro del flowgraph las cosas parecían funcionar perfectamente pero al pasar a testear sobre *gr-isdbt* aparecieron otra clase de problemas, al igual que en los televisores comerciales. Esto deja en evidencia la necesidad de tener un profundo dominio de cada aspecto de la norma, de otra manera, por ejemplo, un simple error al realizar los ajustes de atraso hace que el sistema deje de funcionar por completo.

Por otro lado, sobre el objetivo de aumentar la cantidad y la calidad de la información sobre la norma disponible para los técnicos nacionales, consideramos que este objetivo ha sido cumplido, si bien el alcance del mismo no estará determinado hasta que este documento se haga público. El repositorio con el proyecto está disponible de forma gratuita para cualquier interesado en aprender más de la norma ISDB-T, y en conjunto con gr-isdbt, se tiene una herramienta de análisis del sistema, que cuenta con control absoluto de todas las variables.

Esperamos que este proyecto pueda colaborar con el mejor desempeño de la red nacional de transmisoras de televisión abierta, y que se haga uso de las herramientas que la norma incluye. En esa línea, esperamos aumentar la popularidad del repositorio publicando a la brevedad un video demostrativo en YouTube sobre nuestro transmisor, que esperamos que logre atraer interesados al código que

puedan realizar más pruebas y nuevos aportes para mejorar la calidad del mismo.

Queda pendiente, para este grupo de trabajo, el mejor desarrollo de algunos de los bloques del sistema. Algunos ítems, como la portabilidad del divisor jerárquico, será necesario resolver cuanto antes, para robustecer el funcionamiento del transmisor. También las mejoras por el lado de la optimización de la programación y el manejo de recursos de la PC. En la primer ejecución de *gr-isdbt* para transmitir con el USRP, el bloque OFDM Frame Structure acaparaba todo el procesador. Para seguir adelante fué necesario replantearse la programación del bloque, de otra manera era imposible tener un transmisor funcional. Esto se logró y mejoró notablemente el rendimiento, pero aún queda trabajo en algunos otros bloques demandantes.

Finalmente, nos gustaría plantear un objetivo a mediano o largo plazo, que es un pequeño proyecto que hubiese sido implementado de disponer con el tiempo suficiente. Este transmisor, necesita como fuente de datos un archivo de BTS. Sería muy interesante crear un bloque TS Remux que tome tres flujos MPEG-2 y devuelva el BTS con su correspondiente patrón de ordenamiento. De todas maneras, utilizando los bloques existentes en GNU Radio, es posible alimentar al *gr-isdbt-tx* con tres flujos MPEG-2 y lograr un funcionamiento satisfactorio.

Referencias

[1] Javier Hernandez and Santiago Castro. gr-isdbt-tx. https://github.com/jhernandezbaraibar/gr-isdbt-Tx, 2018. [Web; accedido el 04-12-2018].

[2] ARIB. Transmission System for Digital Terrestrial Television Broadcasting. https://www.arib.or.jp/english/html/overview/doc/6-STD-B31v1_6-E2.pdf, 2001.

[3] Presidencia de la República. Decreto de Presidencia N° 153/12. https://www.impo.com.uy/bases/decretos/153-2012, 2012. [Web; accedido el 24-11-2018].

[4] MIEM. Decreto N° 585/12. http://archivo.presidencia.gub.uy/sci/decretos/2012/05/miem_585.pdf, 2015. [Web; accedido el 24-11-2018].

[5] Instituto Nacional de Estadística. *Encuesta Continua de Hogares*, 2017.

[6] Federico Larroca, Pablo Flores-Guridi, Gabriel Gómez, Víctor González Barbone, and Pablo Belzarena. An open and free ISDB-T full seg receiver implemented in GNU Radio. https://iie.fing.edu.uy/publicaciones/2016/LFGGB16/LFGGB16.pdf, 2016.

[7] Pablo Flores-Guridi. La norma ISDBT y un receptor implementado en SDR. https://iie.fing.edu.uy/investigacion/grupos/artes/wp-content/uploads/sites/13/2017/04/tesis_MSc_pflores-corregida2-acta.pdf, 2016.

[8] Robert W Chang. Synthesis of band-limited orthogonal signals for multichannel data transmission. *Bell System Technical Journal*, 45(10):1775–1796, 1966.

[9] S Weinstein and Paul Ebert. Data transmission by frequency-division multiplexing using the discrete Fourier transform. *IEEE transactions on Communication Technology*, 19(5):628–634, 1971.

[10] Todd K Moon. Error correction coding. *Mathematical Methods and Algorithms. Jhon Wiley and Son*, 2005.

[11] MPEG. Moving Picture Experts Group. https://mpeg.chiariglione.org/, 1995. [Web; accedido el 06-11-2018].

[12] MPEG. MPEG 4 part 10. https://www.iso.org/standard/52974.html, 2009. [Web; accedido el 5-10-2018].

[13] MPEG. MPEG 2 part 1. https://www.iso.org/standard/55688.html, 2010. [Web; accedido el 5-10-2018].

[14] Pablo Flores-Guridi and Federico Larroca. The ISDB-T multiplex frame pattern explained. In *URUCON, 2017 IEEE*, pages 1–4. IEEE, 2017.

[15] G David Forney. Concatenated codes. 1965.

[16] Jan-Jaap Van de Beek, Magnus Sandell, and Per Ola Borjesson. ML estimation of time and frequency offset in OFDM systems. *IEEE transactions on signal processing*, 45(7):1800–1805, 1997.

[17] Allan S Margulies and Joseph Mitola. Software defined radios: a technical challenge and a migration strategy. In *Spread Spectrum Techniques and Applications, 1998. Proceedings., 1998 IEEE 5th International Symposium on*, volume 2, pages 551–556. IEEE, 1998.

[18] GNU Radio. GNU Radio Software. https://www.gnuradio.org/, 2016. [Web; accedido el 01-10-2018].

[19] Ettus Research. https://www.ettus.com/, 2010. [Web; accedido el 01-10-2018].

[20] Great Scott Gadgets, 2018. [Web; accedido el 01-10-2018].

[21] Ettus Research. https://kb.ettus.com/B100, 2010. [Web; accedido el 01-10-2018].

[22] Ettus Research. The USRP™ Hardware Driver Repository. https://github.com/EttusResearch/uhd, 2018. [Web; accedido el 24-11-2018].

[23] Git User BogdanDIA. DVB-T implementation in GNU Radio. https://github.com/BogdanDIA/gr-dvbt, 2015. [Web; accedido el 06-11-2018].

[24] Rohde & Schwarz. R & S ETH Handheld TV Analyzer. https://www.rohde-schwarz.com/fi/product/eth-productstartpage_63493-10186.html, 2018. [Web; accedido el 8-10-2018].

Glosario

ADC Analog-to-Digital Converter. 31

ARIB Association of Radio Industries and Businesses. 3

ATSC Advanced Television Systems Comittee. 1

BER Bit Error Rate. 51

BPSK Binary Phase Shift Keying. 18

BTS Broadcast Transport Stream. 15

CD Compact Disc. 13

CP Cyclic Prefix. 9

DAC Digital-to-Analog Converter. 21

DBPSK Differential Binary Phase Shift Keying. 25

DINATEL Dirección Nacional de Telecomunicaciones. 2

DTMB Digital Terrestrial Multimedia Broadcast. 1

DVB-T Digital Video Broadcasting - Terrestrial. 1

ES Elementary Stream. 13

FEC Forward Error Correction. 13

FPGA Field Programmable Gate Array. 30

HD High Definition. 18

IFFT Inverse Fast Fourier Transform. 18

IIP ISDB-T Information Packet. 34

ISDB-T Integrated Services Digital Broadcasting-Terrestrial. 1

Glosario

ISI Intersymbol Interference. 9

LATU Laboratorio Tecnológico del Uruguay. 3

LTE Long Term Evolution. 27

MER Modulation Error Rate. 51

MIEM Ministerio de Industria, Energía y Minería. 1

MPEG Moving Picture Experts Group. 5

OFDM Orthogonal Frequency Division Multiplexing. 5

PAT Program Association Table. 14

PES Packetized Elementary Stream. 13

PID Process Identifier. 14

PMT Program Map Table. 14

PRBS Pseudorandom binary sequence. 24

RF Radio Frequency. 27

SD Standard Definition. 18

SDR Software Defined Radio. 3

SP Scattered Pilot. 24

TMCC Transmission and Multiplexing Configuration Control. 25

TS Transport Stream. 13

USRP Universal Software Radio Peripheral. 30

Índice de tablas

Esta página ha sido intencionalmente dejada en blanco.

Índice de figuras

Esta es la última página.
Compilado el lunes 10 diciembre, 2018.
http://iie.fing.edu.uy/

Implementación de un Transmisor de ISDB-T Abierto Bajo el Paradigma de Radio Definida por Software

Javier Hernández (javier.hernandez@fing.edu.uy), Santiago Castro (santiago.castro@fing.edu.uy)

***Resumen*—ISDB-T es el principal estándar adoptado en América Latina para la difusión de televisión digital terrestre. Hoy en día el estándar está consolidado en la región, los países de América Latina han realizado importantes inversiones para desplegar el nuevo sistema de televisión digital terrestre basado en este estándar. Esto nos lleva a investigar y desarrollar nuevas posibilidades en lo que respecta al sistema de televisión digital. En este artículo presentamos gr-isdbt-tx, un transmisor ISDB-T abierto, libre y gratuito totalmente basado en software, implementado en GNU Radio. Esta implementación es capaz de generar una señal de televisión digital compatible con cualquier televisor digital homologado para ISDB-T. También permite que estudiantes, profesionales e investigadores puedan conocer al detalle cómo es que funciona un transmisor de televisión digital, sin necesidad de costear equipos de gran valor económico. En conjunción con gr-isdbt: Un receptor ISDB-T fullseg abierto y libre implementado en GNU Radio, el usuario tiene una poderosa herramienta para correr un sistema de televisión digital de punta a punta, y así explorar y probar el comportamiento de un sistema de comunicación complejo sin ningún tipo de hardware. En este artículo presentamos el transmisor desarrollado, así como sus fundamentos y los problemas más relevantes que han tenido que ser resueltos para lograr una implementación exitosa. También presentamos algunas pruebas y sus resultados.**

***Términos*—Integrated Services Digital Broadcasting, GNU Radio, Software Defined Radio.**

I. Introducción

Los antiguos sistemas de televisión analógica están siendo reemplazados por los nuevos sistemas digitales. Existe una innumerable cantidad de mejoras y ventajas respecto a los antiguos, como por ejemplo una mayor eficiencia espectral, mejor calidad de audio y video, la posibilidad de transmitir múltiples programas al mismo tiempo en el mismo canal, y la utilización de canales de datos. El enorme crecimiento que han tenido las comunicaciones inalámbricas en los últimos tiempos, ha hecho que el espectro radioeléctrico se convierta en un recurso natural finito extremadamente valioso. Varios países ya han transitado el camino apagón analógico, reemplazando definitivamente los sistemas de televisión analógica. Sin embargo hay muchos otros en los que aún conviven ambos sistemas de teledifusión. El despliegue de un nuevo sistema de televisión digital es una tarea mayúscula, tal como en otros campos de las comunicaciones inalámbricas surgen distintas soluciones para el mismo problema. Respecto a los sistemas de televisión digital, se pueden encontrar los siguientes estándares: ISDB (Integrated Services Digital Broadcasting) es el estándar japonés, DVB (Digital Video Broadcasting) es el estándar europeo, ATSC (Advanced Television Systems Committee) es el estándar norteamericano y DTMB (Digital Terrestrial Multimedia Broadcast) es el estándar chino.

Este trabajo se enfoca en el estándar ISDB-T el cual ha sido ampliamente adoptado por los países de América Latina. Muchos países aún se encuentran desplegando sus sistemas de televisión digital, para lograr un despliegue exitoso es imprescindible contar con una comprensión profunda de la norma. Este grado de dominio no sería posible alcanzarlo si no se conoce en detalle todo el sistema, desde la etapa de transmisión hasta la etapa de recepción. Las compañías que fabrican esta clase de equipamiento son algunas pocas en el mundo y sus soluciones no suelen estar al alcance de todos.

Las Radios Definidas por Software (SDR por sus siglas en inglés) son una gran alternativa para desarrollar sistemas de comunicación. Se trata de un nuevo paradigma donde los sistemas de radiocomunicaciones, tradicionalmente implementados en hardware, son en cambio implementados por software. Esto típicamente se lleva a cabo a través de una computadora personal encargada de ejecutar el software, y un hardware genérico que se encarga de sintonizar, filtrar y muestrear la señal de radio a cierta tasa. De esta manera la señal puede ser procesada por la computadora. Hay una gran variedad de estos equipos que van desde unos pocos dólares hasta algunos cientos o incluso miles de dólares. El paradigma de la radio definida por software permite lograr una implementación completa de sistemas de radiocomunicaciones de manera económica y sencilla en cierto sentido.

En este trabajo en particular, utilizamos una PC de propósito general junto con equipos basados en SDR como lo son el USRP B100 (Universal Software Radio Peripheral de Ettus) [1] o el HackRF One (de Great Scott Gadgets) [2] que cumplen con la tasa de muestreo requerida por el estándar ISDB-T. Para el desarrollo utilizamos el toolkit de GNU Radio, se trata de un software ampliamente utilizado para el desarrollo de radios definidas por software. Provee de bloques de procesamiento de señal de propósito general como filtros, re-samplers, moduladores y demoduladores analógicos y digitales. Estos bloques pueden ser combinados para lograr diferentes sistemas de comunicación. También es posible la implementación de nuevos bloques personalizados por parte de los usuarios, de manera de contribuir con el código fuente de GNU Radio. Esto hace que la comunidad de GNU Radio sea muy activa y resulte en una contribución continua de los usuarios tanto en el desarrollo de nuevas radios definidas por software o el testeo de otras implementaciones.

En este trabajo presentamos gr-isdbt-tx

(https://github.com/jhernandezbaraibar/gr-isdbt-Tx), un transmisor de ISDB-T abierto bajo el paradigma de radio definida por software, que continúa y complementa la línea de trabajo comenzada con el receptor gr-isdbt.

Este transmisor ha sido implementado completamente en GNU Radio y puede tomar hasta tres flujos de entrada MPEG-2 tal como lo establece el estándar. Tiene la posibilidad de trabajar tanto en modo one-seg como en full-seg, y el usuario cuenta con la libertad de establecer los parámetros de configuración del sistema. El desarrollo sobre GNU Radio hace que, con solamente un PC, el usuario pueda correr un sistema de televisión digital de punta a punta integrando gr-isdbt-tx con gr-isdbt. Alternativamente con un hardware SDR, como el USRP B100, es posible transmitir la señal por el aire hacia cualquier televisor de uso doméstico compatible con ISDB-T.

En lo que sigue se presenta una descripción del estándar ISDB-T, haciendo hincapié en los puntos fundamentales de esta norma y los parámetros más relevantes. Los detalles en cuanto a la programación y los algoritmos utilizados pueden ser consultados en el repositorio del proyecto [9].

II. El estándar ISDB-T

El estándar de television digital terrestre ISDB-T puede ser caracterizado en cuatro grandes etapas: la conformación del Broadcast Transport Stream (BTS), la codificación de canal, la formación del cuadro OFDM, y la puesta en el aire de la señal. En la figura 1 se presenta el esquema completo del sistema transmisor. El sistema admite como fuente de datos hasta tres flujos de transporte MPEG-2 [3]. Cada flujo está conformado por una secuencia de paquetes denominados *Transport Stream Packet* (TSP) de 188 bytes. Los flujos MPEG-2 son multiplexados por el bloque TS Remux en un único flujo llamado *Broadcast Transport Stream* (BTS); además de la multiplexación de los TSP el TS Remux tiene otras tareas como el agregado de cierta información jerárquica en los paquetes.

Una de las características destacables de ISDB-T es la posibilidad de la transmisión jerárquica de la información, esto es, que cada flujo MPEG-2 puede ser transmitido con una configuración propia con distintos grados de robustez. A la transmisión de un flujo MPEG-2 con su configuración jerárquica se lo denomina *capa jerárquica* y el estándar permite transmitir hasta tres capas en simultáneo. Cada capa tiene asignada un conjunto de portadoras en el espectro que más adelante se analizará su conformación.

Una vez conformado el BTS se comienza a robustecer la señal agregando redundancia y utilizando distintos tipos de entrelazamientos. En los sistemas de comunicaciones es común encontrar que se utilicen códigos correctores de errrores concatenados, esto aumenta significativamente el desempeño de los mismos. En ISDB-T se utiliza como código exterior un Reed-Solomon (204,188) y un código convolucional (FEC) como código interior.

Como cada flujo debe ser procesado por separado es necesario que a cada uno se le aplique un proceso según la configuración jerárquica elegida. Es por esto que hay un bloque **divisor jerárquico** encargado de separar el BTS en tres flujos y encaminarlos según lo que corresponda.

El robustecimiento de la señal se realiza con distintos objetivos, por ejemplo para evitar el fading en frecuencia y las ráfagas de errores. El bloque dispersor de energía tiene como objetivo eliminar posibles largas secuencias de unos o ceros que puedan haber en el flujo de transporte. De otra manera podrían haber errores de sincronismo y también habrían porciones del espectro donde se concentraría la energía, quedando así subutilizado este recurso. El dispersor de energía utiliza un circuito para generar una secuencia pseudo-aleatoria la cual es invertible aplicando el mismo circuito a la secuencia. Como resultado se obtiene una secuencia aleatoria sin largas secuencias de ceros y unos.

A continuación del dispersor de energía se encuentra el bloque de **entrelazamiento de bytes**. El entrelazamiento consiste en aplicar una serie de retardos a los distintos bytes que van arribando al bloque. Como el receptor conoce la manera en que se realizaron estos retardos es capaz de deshacerlos y volver a la secuencia original. Esta estrategia de entrelazamiento es ampliamente utilizada luego de un código de bloque como el Reed-Solomon, este último es capaz de corregir errores de hasta ocho bytes. Si durante la transmisión se corrompiera una cantidad mayor de bytes consecutivos el código se volvería inútil, sin embargo al estar los bytes entrelazados el receptor vé estos errores como puntuales y el código es capaz de corregirlos.

Como *inner code* se utiliza un código convolucional con *puncturing* cuya tasa madre es 1/2. La técnica del puncturing permite lograr **tasas de código** de 1/2, 2/3, 3/4, 5/6, 7/8.

La siguiente etapa consiste en el **entrelazamiento de bits** y **mapeo**. Este proceso es similar al entrelazamiento de bytes; en *gr-isdbt-tx* los bits que ingresan a este bloque son empujados en distintas colas de distintos tamaños, luego se extrae el bit que se encuentra al otro extremo de cada cola. Con esto se logra tener a la salida un flujo de bits entrelazados. Es importante tener en cuenta que, tanto el bloque de entrelazamiento de byte como el de bit, generan atrasos que deben ser compensados. En *gr-isdbt-tx* el mapeo de bits en símbolos complejos se realiza conjuntamente con el entrelazamiento de bits. ISDB-T admite las modulaciones DQPSK, QPSK, 16QAM y 64QAM. Este proceso simplemente consiste en agrupar los bits en símbolos complejos según tablas que pueden consultarse en [4]. Hasta este punto los flujos de datos de las distintas capas vienen siendo procesados de acuerdo a los parámetros de cada una de ellas, ahora resta volver a combinar los tres flujos para lograr un único flujo de transporte. El bloque que se encarga de esta tarea es el **combinador jerárquico**. Luego de armado el flujo único con las tres capas jerárquicas, se realiza un **entrelazamiento temporal** de los símbols complejos. Consiste en distribuír los símbolos complejos en el dominio del tiempo. Esta técnica actúa como mecanismo de protección frente a ruidos impulsivos que típicamente se caracterizan por una corta duración en el tiempo. La **profundidad** o largo del entrelazamiento temporal es el parámetro que rige este proceso de entrelazado y puede ser seteado de manera independiente para cada capa. Está íntimamente ligado a la dispersión temporal que se realiza en los símbolos; a mayor profundidad,

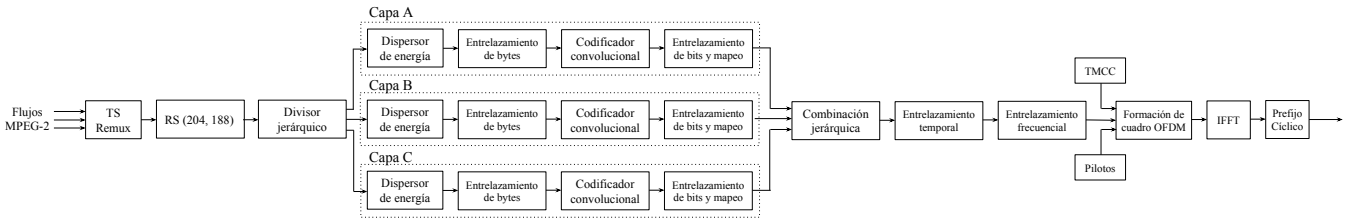

Fig. 1: Esquema del sistema transmisor ISDB-T.

los símbolos de una misma portadora son retardados un tiempo mayor.

Debido a la característica de ISDB-T de utilizar un gran número de portadoras en el espectro, ocurre que frente a canales selectivos en frecuencia se pueden ver afectadas una cantidad importante de portadoras consecutivas. Una estrategia para mitigar esto es el **entrelazamiento frecuencial**. El entrelazamiento que realiza este estándar consiste en dos etapas; el entrelazamiento inter-segmentos y el entrelazamiento intra-segmentos. En la primer etapa los símbolos que comparten la misma modulación son intercalados, mientras que en la segunda etapa los símbolos de un mismo segmento son rotados de acuerdo a cierta expresión que establece el estándar. Posteriormente son aleatorizados según ciertas *Look Up Tables* [4].

Para colocar la señal en el aire, ISDB-T utiliza un esquema de **modulación OFDM** [5] (Orthogonal Frecuency-Division Multiplexing). Este esquema consiste en el uso de portadoras ortogonales, lo que ofrece una mejor eficiencia espectral respecto a los clásicos esquemas de multiplexación en frecuencia. La idea de esta tecnología consiste en modular pulsos rectangulares con los símbolos complejos provenientes del sistema, que como ya se mencionó, pueden ser DQPSK, QPSK, 16QAM y 64QAM. Luego de esta modulación se agrega un **prefijo cíclico** que consiste en una copia del símbolo OFDM y es una fracción del símbolo activo. Puede tomar los valores de 1/4, 1/8, 1/16, 1/32. Esta copia consiste en tomar la última porción del símbolo activo OFDM y copiarlo al comienzo. Es posible ver que si la señal tiene un ancho de banda W, y N es la cantidad total de portadoras, entonces un símbolo activo OFDM tiene una duración $T_S = N/W$ [5].

En cuanto al número de portadoras utilizadas, ISDB-T admite la utilización de 2^{11}, 2^{12} o 2^{13} portadoras según si el **modo de transmisión** es 1, 2 o 3 respectivamente. Sin embargo no todas estas portadoras son utilizadas ya que algunas son reservadas para mantener un intervalo de guarda. Las portadoras son divididas en 14 conjuntos denominados **segmentos**, cada uno de ellos con igual número de portadoras. De esos 14, sólo 13 son utilizados para transmitir datos y el restante es utilizado como guarda a ambos lados del espectro.

Es usual encontrar que en los sistemas de comunicación digitales se delimiten los datos transmitidos dándoles forma de cuadros con una duración fija. Esto vuelve más sencillas las tareas de sincronización dado que en recepción se sabe en qué momento comienza un cuadro. Particularmente en los sistemas OFDM estos cuadros se denominan precisamente

Parámetro	Valor
Ancho de banda del canal	6 MHz
Cantidad de segmentos	13
Ancho de banda de cada segmento	$6000/14 \approx 428.57kHz$
Cantidad de portadoras activas por segmento	96 de datos y 12 pilotos (Modo 1) 192 de datos y 24 pilotos (Modo 2) 384 de datos y 48 pilotos (Modo 3)
Duración de símbolo activo	$252\mu s$ (Modo 1) $504\mu s$ (Modo 2) $1008\mu s$ (Modo 3)
Duracion del prefijo cíclico	1/4, 1/8, 1/16, 1/32 (fracción del símbolo activo)
Tasa de código convolucional	1/2, 2/3, 3/4, 5/6, 7/8
Tasa de código Reed-Solomon	(188, 204)
Profundidad del entrelazamiento temporal	0, 1, 2, 4 (Modo 1) 0, 2, 4, 8 (Modo 2) 0, 4, 8, 16 (Modo 3)
Esquemas de modulación	DQPSK, QPSK, 16QAM, 64QAM
Frecuencia de muestreo (f_{IFFT})	$512/63 \approx 8.127$ MHz

Tabla I: Parámetros relevantes en el estándar ISDB-T.

cuadros OFDM. El cuadro OFDM es una estructura de datos que agrupa los datos de carga útil, portadoras piloto y señales de control. Tiene todo lo necesario para que un receptor compatible con ISDB-T sea capaz de decodificarlo en flujos MPEG-2 y reproducir la información.

Se trata de un conjunto de 204 símbolos OFDM, cada uno de ellos formados por $13 \times 108 \times 2^{modo-1}$ símbolos complejos. El bloque que se encarga de conformar el cuadro OFDM también tiene a cargo las tareas de insertar ciertas señales piloto como son la **Transmission and Multiplexing Configuration Control** (TMCC), las **Scattered Pilot** (SP) y las **Auxiliary Channel** (AC).

La TMCC contiene información de sincronismo y control para que el receptor pueda saber qué cantidad de capas jerárquicas se está utilizando, qué segmentos están asignados para cada una de ellas, la modulación utilizada, el código convolucional y entrelazamiento temporal entre otra información útil.

Las SP o portadoras dispersas son señales BPSK que se insertan a lo largo de los símbolos OFDM y van rotando símbolo a símbolo. Estas portadoras se generan a partir de un circuito que evoluciona según el índice de la SP dentro del símbolo OFDM. El objetivo de estas portadoras es la estimación del canal y por lo tanto son rotadas para evitar que caigan siempre en un lugar que presente fade en frecuencia.

También existe la posiblidad de transmitir información adicional en las portadoras AC, consiste en portadoras que utilizan modulación DBPSK pensadas para oficiar de canal

auxiliar. En caso de no ser utilizadas se rellenan con unos.

Con el propósito de facilitar el sincronismo en recepción, se utiliza un **piloto continuo** modulado en BPSK a la derecha del espectro. En la tabla I se presenta en detalle los parámetros más relevantes en el estándar.

Una vez armado el cuadro OFDM lo que sigue es la modulación de los símbolos en el esquema OFDM. Para ello se aplica el algoritmo de la **Inverse Fast Fourier Transform** (IFFT) y a continuación se agrega el prefijo cíclico.

III. GNU Radio y las Radios Definidas por Software

GNU Radio es un entorno de desarrollo orientado a procesamiento de señales, gratuito y de código abierto. Mediante la interconexión de bloques de procesamiento, puede usarse en conjunto con antenas de RF para desarrollar SDRs, o en una versión local en modo de simulación de sistemas. La aplicación por defecto trae una amplia gama de bloques funcionales para trabajar, desde herramientas simples como filtros y ecualizadores, hasta estructuras complicadas, como lo es un transmisor TVD completo. Existe además una gran comunidad, muy activa, que está permanentemente publicando nuevos contenidos, para ampliar la gama de herramientas existentes en la actualidad. Esto se da porque al ser de código abierto, es relativamente sencillo crear bloques nuevos. El entorno soporta desarrollos de código en Python y en C++, siendo este ultimo el lenguaje con el que hemos implementado nuestro transmisor *gr-isdbt-tx*, en la figura 4 se puede ver el diagrama de bloques.

IV. El Broadcast Transport Stream

El estándar MPEG no está pensado para la transmisión en capas jerárquicas. Para lograr la característica de la transmisión jerárquica y la recepción parcial, ISDB-T cuenta con una solución propia denominada Broadcast Transport Stream. Esta solución consiste en el uso de un flujo de transporte propio, formado a partir de tres flujos MPEG multiplexados, lo que supone un procesamiento no tan sencillo. El nuevo flujo deberá incluir cierta información jerárquica, que no provee MPEG, que permita identificar los flujos con sus correspondientes capas jerárquicas. Es importante que el sistema funcione a una velocidad de reloj constante y determinada, por este motivo el flujo BTS deberá tener una tasa perfectamente definida y constante independientemente de los parámetros de cada capa.

La conformación del BTS se lleva a cabo en el bloque TS Remux. Es el encargado de agregar 16 bytes al final de cada paquete TS con la ISDB-T information y la paridad; posicionar los TSP dentro del BTS de acuerdo a cierto patrón de ordenamiento que se detallará, para posibilitar la transmisión jerárquica; insertar la cantidad necesaria de paquetes nulos para asegurar una tasa constante

$$r_{BTS} = 4 \times f_{IFFT} \approx 32.508 Mbps \tag{1}$$

La obtención de la fórmula anterior puede encontrarse en [6]. Durante la división jerárquica la ISDB-T information de los TSP es removida por lo cual en las siguentes fases de procesamiento ya no se cuenta con la información jerárquica en los flujos de transporte. Es por eso, que resulta necesario definir un orden dentro del BTS de manera que el receptor

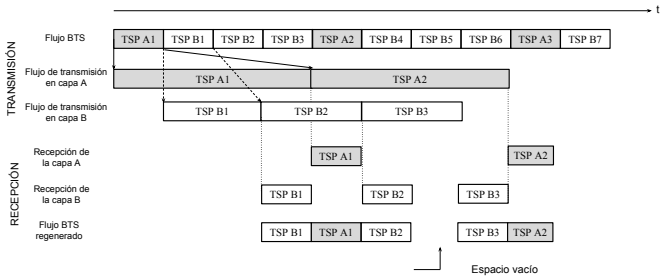

Fig. 2: Patrón de ordenamiento incorrecto del BTS.

pueda reconstruirlo perfectamente sin necesidad de la ISDB-T information. Supongamos que tenemos un BTS como el de la figura 2.4 en el cual hay dos capas jerárquicas A y B, cada una con su respectiva tasa tal que $r_{BTS} > r_B > r_A$. Las capas con tasas altas implican un tiempo de transmisión de paquetes mucho más corto que las capas con tasas bajas, y viceversa. Observando el diagrama de la figura 2 puede verse que en recepción el paquete B1 es procesado antes de que termine de ser transmitido el paquete A1, cuando en realidad el orden original era A1 y luego B1. Además se generan espacios de tiempo en los cuales no se entregan paquetes al sistema porque aún se están procesando paquetes que van arribando. Esos intervalos de tiempo deben ser rellenados con paquetes nulos a fin de mantener la tasa constante. Esto resulta en un desordenamiento del patrón original del BTS y de la pérdida total de la capacidad de reordenar los datos en recepción.

Para formar los patrones de ordenamiento en transmisión de manera tal que el receptor sea capaz de recuperarlos perfectamente, se agregan los denominados ajustes de atraso. En el ejemplo de la figura 3 se agrega un ajuste de atraso de 2 TSP y se introduce un paquete nulo en el BTS original. Como resultado, el flujo BTS reconstruído por el receptor resulta ser exactamente igual al BTS original. Dada una configuración jerárquica del sistema, existe un único patrón de ordenamiento del BTS.

El transmisor implementado en este trabajo toma como flujo de entrada un BTS ya conformado, y con la información jerárquica de los TSP es que logra procesar cada capa por separado. De ahí en adelante las capas son entrelazadas y moduladas cada una de acuerdo a su propia configuración.

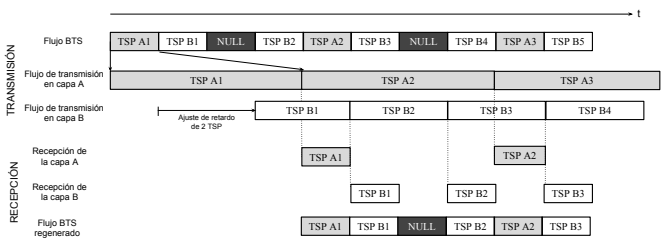

Fig. 3: Patrón de ordenamiento correcto del BTS.

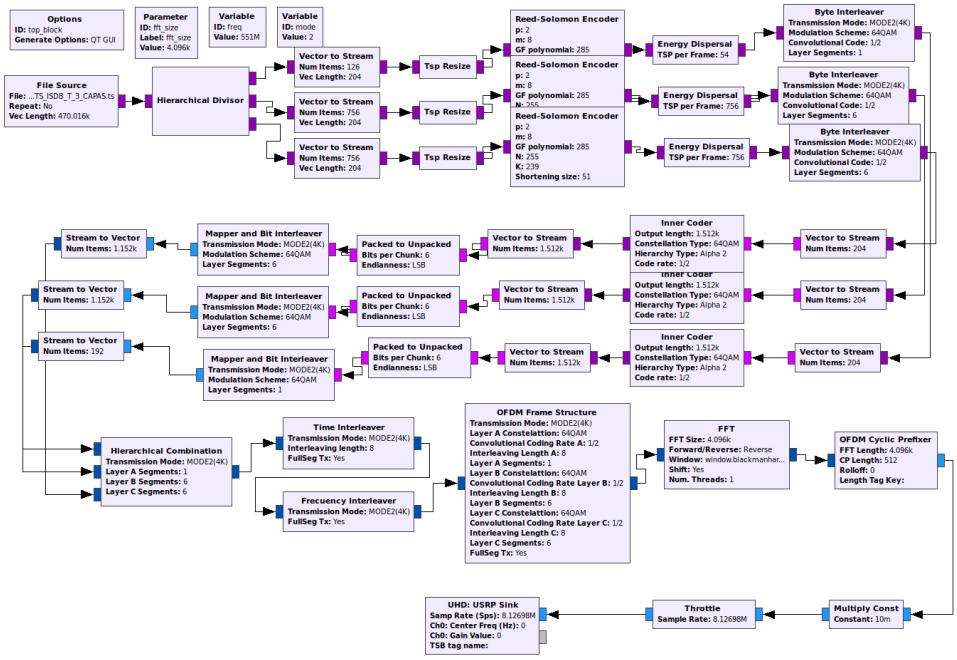

Fig. 4: Flowgraph del sistema gr-isdbt-tx.

V. Los ajustes de atraso y el sincronismo

Los bloques de entrelazamiento introducen atrasos. Estos atrasos son producto de los retardos variables a los que se someten los bytes, bits o símbolos complejos. Para que el sistema funcione correctamente el atraso total introducido por cada bloque debe ser un múltiplo de cuadro OFDM. Esto es un detalle importante que se tuvo en cuenta en el desarrollo de *gr-isdbt-tx*, por lo que entendemos necesario desarrollar el concepto del ajuste de atraso.

A. Ajuste de atraso en el entrelazamiento de byte

El entrelazamiento de bytes se implementa empujando secuencialmente los bytes que arriban al bloque hacia búfers o colas. Estas colas son 12, y tienen un tamaño definido de $17 \times i$ bytes, con $i \in [0, .., 11]$. Este proceso introduce un retardo equivalente al tamaño de la cola más larga multiplicado por el número de colas, es decir que el retardo es:

$$d_{bytes} = (11 \times 17 \times 12) = 2244 \quad bytes \tag{2}$$

Expresado en TSP, los 2244 bytes equivalen a 11 TSPs (cada TSP tiene 188 bytes más 16 bytes insertados por el codificador Reed-Solomon). Para conocer el ajuste que es necesario realizar lo que debemos hacer es calcular cúantos TSP hay en un cuadro OFDM y a este número restarle los 11 TSP que agrega el proceso de por sí. En particular para una capa jerárquica con modulación m_L, código convolucional FEC_L, el ajuste de atraso necesario es el siguiente:

$$D_{TSP} = \frac{204 \times m_L \times FEC_L \times 96 \times 2^{modo-1}}{8 \times 204} - 11 \tag{3}$$

Por lo tanto expresado en bytes, el ajuste que hay que agregar es $204 \times D_{TSP}$ bytes.

B. Ajuste de atraso en el entrelazamiento de bits

En este entrelazamiento se utilizan 2, 4 ó 6 colas dependiendo si la modulación utilizada es QPSK, 16QAM, o 64QAM respectivamente. El tamaño de las colas, q_i, viene definido por el estándar. Por ejemplo para 16QAM los tamaños son $q_0 = 0$, $q_1 = 40$, $q_2 = 80$, $q_3 = 120$. Cualquiera sea la modulación el retardo máximo siempre es de 120 bits. Sin embargo vamos a dejarlo paramétrico en q_i a fin de calcular el ajuste de atraso que es necesario introducir a cada cola.

El retardo agregado por el proceso corresponde al camino más largo que deben atravesar los bits, es decir un retardo de 120 símbolos. Según la modulación que utilice la capa, estos 120 símbolos se traducen en un mayor o menor retardo en el tiempo; para QPSK corresponde a esperar que ingresen 120×2 bits en la entrada, mientras que para 64QAM deben pasar 120×6 bits.

Como los retardos deben ser siempre múltiplos de símbolos OFDM, el tamaño de un símbolo OFDM corresponde a:

$$N_{OFDM} = m_L \times N_L \times 96 \times 2^{modo-1} \quad bits \tag{4}$$

N_{OFDM} depende de la cantidad de segmentos utilizados N_L, del modo de transmisión y de la modulación de la capa $m_L \in \{2, 4, 6\}$.

La solución implementada en este bloque es distinta a la del entrelazador de bytes. Aquí, el retardo total se obtiene incrementando los tamaños de las colas de manera que el retardo total sea de dos símbolos OFDM para cada capa. Por lo tanto para tener el retardo deseado los tamaños de las colas deben ser:

$$Q_i = q_i + \frac{2 \times N_{OFDM} - 120 \times m_L}{m_L} \tag{5}$$

$$Q_i = q_i + N_L \times 96 \times 2^{modo-1} - 120 \quad (6)$$

Una vez que son empujados los bits en cada cola, desde el otro extremo se extrae un bit por cola y se mapean en un símbolo complejo de acuerdo a lo establecido en la norma.

C. Ajuste de atraso en el entrelazamiento temporal

En el caso del entrelazamiento temporal, los símbolos complejos que arriban al bloque son encolados de manera secuencial en búfers de tamaño definido por la siguiente expresión:

$$q_L(i) = I_L \times ((i \times 5)\ mod\ 96) \quad (7)$$

donde $i \in [0, 96 \times 2^{modo-1}]$ representa el número de portadora e I_L es la profundidad del entrelazamiento y puede tomar los valores de la tabla I.

Este proceso genera un atraso de $95 \times I_L\ mod\ 204$ símbolos, entonces para completar un cuadro OFDM hacen falta:

$$d_I = 204 - (95 \times I_L\ mod\ 204) \quad (8)$$

símbolos OFDM. Por lo tanto el retardo total en cuadros OFDM introducido por el proceso será:

$$N_L = (95 \times I_L + d_I)\ mod\ 204 \quad (9)$$

La implementación en el transmisor de este ajuste de atraso consiste en empujar los símbolos que arriban al bloque a cada una de las colas de tamaño $I_L \times (i \times 5)\ mod\ 96 + d_I$. Del otro extremo de las colas se toman los símbolos secuencialmente y se tiene la secuencia entrelazada. Este mecanismo, al igual que otros entrelazamientos, hace que en el arranque del sistema se tengan datos espurios en los registros.

VI. Pruebas

A. Pruebas contra gr-isdbt

Considerando que realizamos la mayoría del desarrollo contrastando los bloques del transmisor contra los de *gr-isdbt*, no parecería en principio un desafío mayor lograr la decodificacion punta a punta. Pero en sí, lo es, y eso es parte de la potencia del desarrollo con código abierto, constantemente podemos estar evaluando nuestro desempeño desde el otro lado.

Como receptor, *gr-isdbt* ha sido sometido a una diversidad de pruebas con resultados satisfactorios, por lo tanto, podemos considerarlo como una simulación de un televisor comercial válida. Es en ese punto en el que comenzamos las pruebas. Dentro de un entorno controlado, el transmisor *gr-isdbt-tx* conectado a la entrada del receptor *gr-isdbt*.

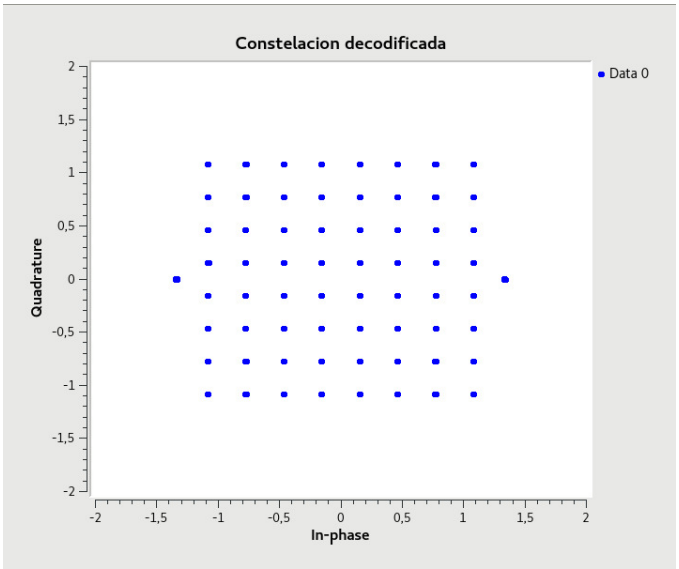

Fig. 5: Constelación recibida con gr-isdbt. Prueba realizada dentro de GNU Radio.

1) Pruebas sobre el flowgraph: En este caso ideal, donde todo funciona dentro de un mismo flowgraph y no hay variabilidad de las muestras transmitidas, el transmisor se comporta de forma perfecta. Detectamos un consumo excesivo de CPU en algunos bloques, en parte debido a problemas de optimización del código.

Para la prueba sobre flowgraph se utilizó un BTS constituído por tres capas jerárquicas donde se destina un segmento para la capa A y seis segmentos para las capas B y C respectivamente, todas moduladas en 64QAM. Tanto la tasa de código convolucional como la profundidad del entrelazamiento temporal son las mismas para las tres capas y son de 1/2 y 8 respectivamente.

Como se puede ver en la figura 5, el receptor *gr-isdbt* se sincroniza perfectamente con la señal transmitida, y en la constelación de recepción, vemos la misma constelación que en transmisión. Esto no debe sorprender, ya que en esta prueba, el canal está representado por un cable ideal. En la misma figura se puede apreciar las modulaciones 64QAM utilizada para las portadoras de datos y BPSK para las señales piloto.

Recibir la constelación perfecta en *gr-isdbt* significa varias cosas. Por un lado el cuadro OFDM está siendo conformado correctamente del lado del transmisor, por otro lado, se está decodificando correctamente las portadoras piloto (SP, TMCC y piloto continuo). Además el receptor entiende perfectamente la información contenida en la TMCC, es decir que su contenido es el correcto y pasa satisfactoriamente los chequeos de paridad.

Podemos asegurar que esta prueba constituye un éxito dado que en el otro extremo de *gr-isdbt* se logró obtener cada una de las tres capas jerárquicas transmitidas sin ningún tipo de error.

2) Pruebas en el aire: Realizamos la misma prueba con *gr-isdbt* como receptor, pero en lugar de simular un canal en GNU Radio, utilizamos el aire como medio, transmitiendo en una línea vista, a un metro de distancia.

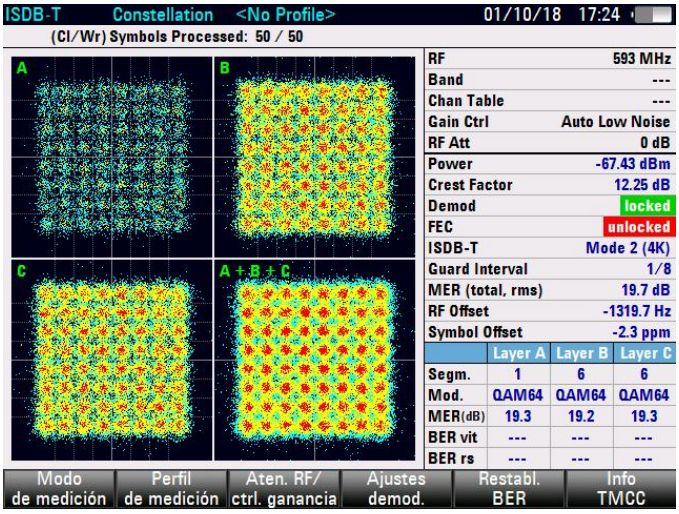

Fig. 6: Constelación de las tres capas jerárquicas tomadas con el analizador de TV ETH.

En una primera instancia no se obtenía video a la salida de *gr-isdbt*, se veía la constelación decodificada de manera intermitente pero con demasiado ruido. Para esta prueba se estaba utilizando el USRP B100 como transmisor y el HackRF como receptor. Luego de realizar varias pruebas pudimos constatar que el HackRF introduce un offset de continua considerable lo cual no lo hace particularmete bueno para esta prueba. Una vez identificado este problema junto con otros bugs en el transmisor, fué posible la recepción exitosa con *gr-isdbt*.

B. Pruebas contra televisores comerciales

Las pruebas de validación con *gr-isdbt*, tanto dentro del framework de GNU Radio como en el aire, permitieron dar el siguiente paso para probar el transmisor contra televisores comerciales homologados para el estándar.

Esta prueba consistió en utilizar un laptop con procesador Intel Core i7, 16GB de RAM, disco SSD, corriendo *gr-isdbt-tx* junto con un USRP B100 y una antena para la banda de TV digital. Desde el televisor se realizó un escaneo y la detección de nuestro canal fué inmediata. El televisor logró reproducir perfectamente el contenido que le estábamos transmitiendo, mostrando una excelente calidad de audio y video para cada una de las capas jerárquicas. Se probó transmitir sin línea de vista, moviendo la antena transmisora, y a una distancia de hasta 5 metros con excelentes resultados.

C. Pruebas contra equipos analizadores de TV

Durante el desarrollo de las pruebas anteriores aparecieron una serie de bugs que se presentaron como un verdadero desafío. Los problemas de recepción inalámbrica con *gr-isdbt* y las imperfecciones al momento de recibir en un TV comercial, motivaron el uso de sofisticadas herramientas comerciales para tener una idea de las posibles causas. Se tuvo la posibilidad de acceder a un equipo ETH de Rohde & Schwarz [7], propiedad de la Unidad Reguladora de Servicios de Comunicaciones [8], que es capaz de analizar señales de TV compatible con ISDB-T.

Como vemos en la figura 6, con este instrumento se pudo observar claramente las constelaciones de las tres capas jerárquicas pero con demasiado ruido. También se observó que la señal presenta una variabilidad bastante grande del offset en frecuencia, que va desde los $400Hz$ a los $1300Hz$. Este parámetro para las señales comerciales de TVD presenta una estabilidad considerable, variando en el entorno de los $5Hz$. Los cristales de los equipos SDR utilizados tienen una precisión del orden de las 20ppm, lo cual hace que sea razonable esta variación encontrada.

VII. Conclusiones y trabajo a futuro

Este proyecto tenia como objetivos, dos grandes ítems. Por un lado buscábamos implementar un transmisor de televisión digital basado en SDR, cuya validez estaría determinada por la capacidad de transmitir hacia un televisor comercial homologado para ISDB-T. Por otro lado, también buscábamos echar luz sobre el conocimiento disponible sobre la norma ISDB-T.

En cuanto al primer objetivo, entendemos que ha sido cumplido de forma satisfactoria. Hemos probado el transmisor contra varios televisores comerciales, de diferentes proveedores, y en todos los casos el equipo encuentra el canal transmitido de forma exitosa, y decodifica las tres capas procesadas de forma correcta.

Durante las pruebas encontramos diversos problemas que, mediante distintas estrategias, hemos podido solucionar. Cada prueba realizada aportó su cuota en el proceso de validación del transmisor. Como se mencionó anteriormente, dentro del flowgraph las cosas parecían funcionar perfectamente pero al pasar a testear sobre *gr-isdbt* aparecieron otra clase de problemas, al igual que en los televisores comerciales. Esto deja en evidencia la necesidad de tener un profundo dominio de cada aspecto de la norma, de otra manera, por ejemplo, un simple error al realizar los ajustes de atraso hace que el sistema deje de funcionar por completo.

El repositorio con el proyecto está disponible de forma gratuita para cualquier interesado en aprender mas de la norma ISDB-T, y en conjunto con *gr-isdbt*, se tiene una herramienta de análisis del sistema, que cuenta con control absoluto de todas las variables.

Queda pendiente el mejor desarrollo de algunos de los bloques del sistema. Algunos ítems, como la portabilidad del divisor jerárquico, será necesario resolver cuanto antes para robustecer el funcionamiento del transmisor. También las mejoras por el lado de la optimización de la programación y el manejo de recursos de la PC. En la primer ejecución de *gr-isdbt* para transmitir con el USRP, el bloque OFDM Frame Structure acaparaba todo el procesador. Para seguir adelante fué necesario replantearse la programación del bloque, de otra manera era imposible tener un transmisor funcional. Esto se logró y mejoró notablemente el rendimiento, pero aún queda trabajo en algunos otros bloques demandantes.

Por otra parte entendemos que con este trabajo se ha podido dejar en evidencia una vez más las grandes capacidades

que tienen GNU Radio y las SDR, y el amplio abanico de posibilidades que ofrecen a la hora de desarrollar sistemas de comunicaciones.

REFERENCIAS

[1] E. Research, https://kb.ettus.com/B100, 2010, [Web; accedido el 01-10-2018].

[2] G. S. Gadgets, https://greatscottgadgets.com, 2018, [Web; accedido el 01-10-2018].

[3] J. Hernandez and S. Castro, "gr-isdbt-tx," https://github.com/jhernandezbaraibar/gr-isdbt-Tx, 2018, [Web; accedido el 04-12-2018].

[4] MPEG, "Moving Picture Experts Group," https://mpeg.chiariglione.org/, 1995, [Web; accedido el 06-11-2018].

[5] ARIB, "Transmission System for Digital Terrestrial Television Broadcasting," https://www.arib.or.jp/english/html/overview/doc/6-STD-B31v1_6-E2.pdf, 2001.

[6] R. W. Chang, "Synthesis of band-limited orthogonal signals for multichannel data transmission," *Bell System Technical Journal*, vol. 45, no. 10, pp. 1775–1796, 1966.

[7] P. Flores-Guridi, "La norma ISDBT y un receptor implementado en SDR," https://iie.fing.edu.uy/investigacion/grupos/artes/wp-content/uploads/sites/13/2017/04/tesis_MSc_pflores-corregida2-acta.pdf, 2016.

[8] R. . Schwarz, "R & S ETH Handheld TV Analyzer," https://www.rohde-schwarz.com/fi/product/eth-productstartpage_63493-10186.html, 2018, [Web; accedido el 8-10-2018].

[9] URSEC, "Unidad reguladora de servicios de comunicaciones," https://www.ursec.gub.uy, 2018, [Web; accedido el 04-12-2018].

[10] F. Larroca, P. Flores-Guridi, G. Gmez, V. Gonzlez Barbone, and P. Belzarena, "An open and free ISDB-T full seg receiver implemented in GNU Radio," https://iie.fing.edu.uy/publicaciones/2016/LFGGB16/LFGGB16.pdf, 2016.

[11] S. Weinstein and P. Ebert, "Data transmission by frequency-division multiplexing using the discrete Fourier transform," *IEEE transactions on Communication Technology*, vol. 19, no. 5, pp. 628–634, 1971.

[12] T. K. Moon, "Error correction coding," *Mathematical Methods and Algorithms. Jhon Wiley and Son*, 2005.

[13] MPEG, "MPEG 4 part 10," https://www.iso.org/standard/52974.html, 2009, [Web; accedido el 5-10-2018].

[14] ——, "MPEG 2 part 1," https://www.iso.org/standard/55688.html, 2010, [Web; accedido el 5-10-2018].

[15] P. Flores-Guridi and F. Larroca, "The ISDB-T multiplex frame pattern explained," in *URUCON, 2017 IEEE*. IEEE, 2017, pp. 1–4.

[16] G. D. Forney, "Concatenated codes." 1965.

[17] J.-J. Van de Beek, M. Sandell, and P. O. Borjesson, "ML estimation of time and frequency offset in OFDM systems," *IEEE transactions on signal processing*, vol. 45, no. 7, pp. 1800–1805, 1997.

[18] A. S. Margulies and J. Mitola, "Software defined radios: a technical challenge and a migration strategy," in *Spread Spectrum Techniques and Applications, 1998. Proceedings., 1998 IEEE 5th International Symposium on*, vol. 2. IEEE, 1998, pp. 551–556.

[19] G. Radio, "GNU Radio Software," https://www.gnuradio.org/, 2016, [Web; accedido el 01-10-2018].

[20] E. Research, https://www.ettus.com/, 2010, [Web; accedido el 01-10-2018].

[21] ——, "The USRP? Hardware Driver Repository," https://github.com/EttusResearch/uhd, 2018, [Web; accedido el 24-11-2018].

[22] G. U. BogdanDIA, "DVB-T implementation in GNU Radio," https://github.com/BogdanDIA/gr-dvbt, 2015, [Web; accedido el 06-11-2018].