Universidad de la República
Facultad de Ingeniería

Desarrollo de Transceptor ISDB-T en Radio Definida por Software e Implementación en Hardware Alternativo de Bajo Costo

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

Gastón Morales, Lucas Inglés, David Artenstein

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

Tutor

Pablo Flores ..... Universidad de la República
Federico La Rocca ..... Universidad de la República

Tribunal

Pablo Belzarena ..... Universidad de la República
Marcelo Coggan ..... Canal 10 y TCC
Gabriel Gómez ..... Universidad de la República
Mauricio González ..... Universidad de la República

Montevideo
jueves 20 agosto, 2020

Desarrollo de Transceptor ISDB-T en Radio Definida por Software e Implementación en Hardware Alternativo de Bajo Costo, Gastón Morales, Lucas Inglés, David Artenstein.

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.1).
Contiene un total de 132 páginas.
Compilada el jueves 20 agosto, 2020.
http://iie.fing.edu.uy/

Si comienza uno con certezas, terminará con dudas; mas si se conforma en comenzar con dudas, llegará a terminar con certezas.

Francis Bacon

Esta página ha sido intencionalmente dejada en blanco.

Agradecimientos

A nuestras familias, por el apoyo incondicional a lo largo de este camino. A nuestros amigos, que tanto en los momentos buenos como en los malos estuvieron siempre ahí. A nuestros profesores, que dedicaron su tiempo y su paciencia para mostrarnos el camino. A todos los compañeros que nos supieron acompañar a lo largo de toda la carrera. A nuestros tutores por su inmensa disposición para guiarnos durante el proceso. Al Instituto de Ingeniería Eléctrica de la Facultad de Ingeniería y todo su personal, por brindarnos el apoyo siempre que fue necesario.

Esta página ha sido intencionalmente dejada en blanco.

Resumen

En este proyecto se presenta un nuevo aporte en el desarrollo del transceptor ISDB-T (norma de transmisión de TV digital terrestre en Uruguay) en SDR. Como primer planteo del proyecto, se encuentra la adaptación del *software* del transmisor ISDB-T a *hardware* SDR alternativo de bajo costo. Los equipos de hardware alternativo a utilizar son un adaptador USB-VGA (cuyo costo es de 20 U$S en plaza) y una placa Raspberry Pi (costo de 40 U$S). Para poder llevar esto a cabo, se realizó una implementación one-seg del transmisor ISDB-T en radio definida por *software*. Esta implementación posibilita trabajar con un menor ancho de banda, optimizando la limitada potencia de transmisión que presenta el *hardware* SDR alternativo. Fue posible comprobar el correcto funcionamiento del adaptador USB-VGA como transmisor ISDB-T con variadas pruebas, realizando comparativas contra un equipo SDR convencional y logrando también transmitir hacia un sintonizador ISDB-Tb comercial. Respecto a la placa Raspberry Pi, esta fue estudiada en su funcionamiento como transmisor SDR, y se realizaron pruebas de transmisión de distinto tipo.

Por otro lado, se realizaron modificaciones al *software* transmisor ISDB-T full-seg ya existente, permitiendo que acepte una mayor variedad de *transport streams* de entrada y verificando su correcto funcionamiento para toda la variedad de parámetros de transmisión. Esto fue verificado mediante distintas pruebas, tanto frente a sintonizadores ISDB-Tb comerciales, como frente a receptores SDR. A su vez, en base a los requerimientos de codificación y multiplexación de la norma ISDB-Tb, se logró generar *transport streams* que pueden ser inyectados al transmisor. Estos puede ser generados a partir de videos almacenados, o en tiempo real desde una cámara web por ejemplo.

Esta página ha sido intencionalmente dejada en blanco.

Tabla de contenidos

Agradecimientos III

Resumen V

1. Introducción 1

2. Fundamento Teórico 5
2.1. Radio Definida por Software 5
2.2. GNU Radio 8
2.3. Modelo de un canal inalámbrico 9
2.4. Sistemas OFDM 10
2.4.1. Modelo OFDM para tiempo continuo 11
2.4.2. Modelo OFDM para tiempo discreto 14

3. ISDB-T 17
3.1. Introducción 17
3.2. Consideraciones generales 17
3.3. Transmisor ISDB-T 19
3.3.1. Broadcast Transport Stream 19
3.3.2. Etapas de procesamiento de un modulador ISDB-T 20
3.4. Codificación de video y audio en ISDB-Tb 23
3.4.1. ABNT NBR 15602 23
3.4.2. Bitrates en ISDB-T 26

4. Desarrollo y pruebas de software transmisor ISDB-T 29
4.1. Desarrollo de transmisor ISDB-T full-seg 29
4.2. Transmisor one-seg 31
4.2.1. Introducción 31
4.2.2. Implementación de transmisor one-seg en GNU Radio . . . 33
4.3. Transmisión de *transport streams* bajo la norma ISDB-T con sus distintas variantes de parámetros 36
4.3.1. Descripción de las pruebas 36
4.3.2. Resultados 37

Tabla de contenidos

5. Generación de *transport streams* para transmisión **41**
5.1. Uso de *ffmpeg* para codificación de videos almacenados 41
5.2. Uso de *ffmpeg* para codificación de videos generados en tiempo real 43

6. Adaptador USB-VGA como transmisor de bajo costo **45**
6.1. Introducción 45
6.2. Funcionamiento 47
6.3. Transmisión de *transport streams* bajo la norma ISDB-T 49

7. Transmisión de *transport streams* bajo la norma ISDB-T a través de adaptador USB-VGA **55**
7.1. Hardware utilizado 55
7.2. Frecuencia de trabajo y configuración de antenas 59
7.3. Pruebas de transmisión 59
7.3.1. Modulation Error Rate (MER) 60
7.3.2. Transmisión one-seg sin amplificador 60
7.3.3. Transmisión full-seg sin amplificador 64
7.3.4. Transmisión full-seg con amplificador 67
7.3.5. Transmisión full-seg con amplificador y filtro de radio frecuencia 70

8. Experimentos del transmisor ISDB-T variando el hardware **75**
8.1. Comparación entre el hardware USB-VGA y equipos SDR comerciales 75
8.1.1. Introducción 75
8.1.2. Transmisión one-seg 75
8.1.3. Transmisión full-seg 78
8.2. Funcionamiento contra sintonizador ISDB-Tb comercial 80

9. Placa Raspberry Pi como transmisor SDR **85**
9.1. Introducción 85
9.1.1. Raspberry Pi 85
9.1.2. Proyectos RpiTX y RpiDATV 86
9.2. Modulación en RpiTX y RpiDATV 88
9.2.1. RpiDATV 88
9.2.2. RpiTX 89
9.3. Transmisiones en RpiTX a través de datos IQ 92

10. Pruebas de Raspberry Pi como transmisor SDR **95**
10.1. Transmisión FM 95
10.2. Transmisión AM 99
10.3. Transmisión ISDB-T 100

11. Conclusiones **105**

Referencias **109**

Índice de tablas **113**

Índice de figuras **114**

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 1

Introducción

Las comunicaciones sufrieron una evolución vertiginosa desde la aparición de la radio. Conjuntamente con un importante avance de la electrónica, se lograron objetivos sin precedentes como la aparición de servicios de redes móviles, comunicación satelital, y altas tasas de transferencia de información.

La teledifusión como parte de esta constante evolución, ha recorrido un largo camino a lo largo de los años. Desde la primera transmisión de televisión por aire en 1929, hasta la incorporación de resoluciones 4K en las distintas normas de transmisión de televisión digital, todos los adelantos llegaron en un contexto dado enfrentando una necesidad.

La aparición de la televisión digital ha ido de la mano con la evolución de los sistemas de computación, principalmente con el desarrollo de los distintos algoritmos de codificación de la información audiovisual. Tanto la posibilidad de codificar las señales a tasas de bit razonables para su transmisión, como la creciente calidad audiovisual que los nuevos *codecs* permiten lograr, han generado que la televisión digital haya desplazado progresivamente a la televisión analógica.

Dentro de la televisión digital, es posible agrupar las normas en distintas categorías:

- Televisión digital por cable (transmisión de las señales a través de redes de cables hacia los hogares)
- Televisión digital satelital (transmisión de las señales vía satélite)
- Televisión digital terrestre (transmisión de las señales a través de transmisores terrestres)

En Uruguay, en el año 2010, se decidió adoptar la norma ISDB-Tb [1] como norma de transmisión de televisión digital terrestre. ISDB-Tb es la variante brasileña de la norma japonesa ISDB-T [2] (*Integrated Services Digital Broadcasting - Terrestrial*), siendo ambas normas muy similares entre sí en buena parte de su implementación. Las únicas diferencias entre estas son el estándar de interactividad que se utiliza en cada una, y algunas diferencias respecto a los *codecs* compatibles.

Capítulo 1. Introducción

Son numerosas las ventajas que la transmisión de televisión digital terrestre tiene frente a la tradicional transmisión analógica. Entre las más destacadas ventajas se encuentra la liberación del espectro, permitiendo mayor eficiencia en un recurso que es escaso. Además, la posibilidad de multiplexar distintos *streams* en un mismo canal, como ser distintas señales de video, o distintas señales de audio (a distintos lenguajes por ejemplo), dan un valor agregado a esta tecnología. Entre otras posibilidades de la TV digital, se encuentra la de adicionar servicios agregados, como puede ser integrar guías de programación. También la calidad de imagen y sonido a nivel de resolución, cuadros por segundo, y otros aspectos técnicos, inclinan la balanza hacia una tecnología con una evolución cada vez mas avanzada.

Debido a estos beneficios, el mundo hoy está enfrentando un cambio trascendente con vista al futuro. El cese de las emisiones analógicas de los operadoras de televisión, suceso conocido como Apagón Analógico. Esta transición es llevada a cabo por los distintos países de manera independiente. En Uruguay actualmente no existe una restricción puntual sobre las emisiones de televisión analógica. La emisión de televisión digital queda a criterio de cada estación, sin embargo se promueve a estas para que acompañen el cambio. Esto implica que los receptores de televisión digital terrestre sean cada vez más comunes en los hogares del país. A su vez, esto genera impulso para la investigación y el desarrollo de las tecnologías relativas a la televisión digital.

El paradigma de transmisión de señales de televisión en el mercado actual involucra costosos equipos de *hardware* de ecosistema cerrado, y de alto costo económico. Esto hace inalcanzable la transmisión ya sea para transmisoras de bajo presupuesto, o transmisiones con fines didácticos en la mayoría de los escenarios. A los grandes costos de estos equipos, se le agrega la dificultad de poder hacerle modificaciones si se quiere transmitir otra tecnología. Como posible solución a este problema, se encuentra la "Radio Definida por *Software*" (SDR por sus siglas del inglés, *Software Defined Radio*) [3]. La Radio Definida por *Software* permite implementar en *software* en una computadora personal, componentes de radiocomunicación que tradicionalmente eran implementados en *hardware* (filtros, amplificadores, moduladores, demoduladores, como algunos ejemplos). Los sistemas SDR implementados en computadoras personales, son usualmente acompañados de *hardware* transceptor que permite realizar la función de transmisión o recepción de señales. Estos son comúnmente denominados como SDR. Este paradigma abre las puertas a la utilización de sistemas completos de SDR (computadora personal y transceptor SDR por ejemplo), como sistemas de comunicación multipropósito, pudiendo abarcar una gran variedad de tecnologías de radiocomunicación con un mismo equipamiento.

Utilizando esta tecnología, surgen los proyectos "Implementación de un transceptor de ISDB-T abierto y para metrología bajo el paradigma de Radio Definida por Software" [4] [5] e "Implementación de un Transmisor de ISDB-T Abierto Bajo el Paradigma de Radio Definida por Software" [6], proyectos realizados dentro del Instituto de Ingeniería Eléctrica, que abren la puerta a la utilización de computadoras personales y transmisores de radio definida por *software* como interesantes

alternativas. A pesar de este gran avance dentro de la transmisión de televisión digital, el *hardware* SDR sigue siendo costoso, en el entorno de cientos de dólares. Con el fin de realizar una implementación del transmisor de televisión digital en un sistema lo más económico posible, este proyecto se propuso como objetivo encontrar alternativas de *hardware* transmisor, de manera de sustituir los equipos antes mencionados.

El proyecto Osmo-fl2k [7] propone la utilización de un adaptador USB-VGA comercial para la transmisión de distintas tecnologías a partir de emisiones electromagnéticas, lo cuál abrió paso a la implementación del transmisor ISDB-T con dicho *hardware*. A su vez, los proyectos RpiTX [8] y RpiDATV [9], utilizan una placa Raspberry Pi para la transmisión de diferentes tecnologías, brindando otra alternativa de *hardware* a utilizar. El precio del adaptador USB-VGA es de unos 20 dólares y el de la Raspberry Pi 3B+ de unos 40 dólares. Esto hace que se reduzcan los costos del transmisor considerablemente, siendo accesible no sólo por el precio, sino por la facilidad que se tiene para encontrar a la venta los equipos mencionados. La implementación del transmisor en un *hardware* alternativo conlleva a diferentes pruebas, para poder determinar sus capacidades de transmisión. A su vez, se debe comparar qué tanto se gana o se pierde con respecto a los transmisores SDR desde el punto de vista técnico.

La potencia de transmisión de los equipos alternativos es más limitada comparándola con los transmisores SDR, por lo que la reducción del ancho de banda de la señal a transmitir permitirá que los equipos alternativos optimicen el uso de la potencia. Por este motivo, la implementación de un transmisor ISDB-T one-seg llevada a cabo en este proyecto es fundamental. Este desarrollo causaría que se reduzca el ancho de banda utilizado considerablemente, ya que no se transmiten las 13 subdivisiones o segmentos de ancho de banda estipulados en una transmisión ISDB-T completa. En cambio, se transmite un solo segmento.

Por otra parte, para la transmisión de ISDB-T se necesita que los *transport streams*[1] a transmitir tengan el formato definido en la norma ISDB-T. La correcta codificación y multiplexación permitiría procesar y transmitir cualquier tipo de video almacenado, como pueden ser videos descargados de internet. Incluso permite transmitir el video generado a través de una cámara web para lograr transmisiones en vivo.

Este documento expone el estudio e implementación del transmisor ISDB-T utilizando *hardware* alternativo de bajo costo, realizando las pruebas de funcionamiento pertinentes y comparándolas con el *hardware* SDR comúnmente utilizado. Se realizan pruebas en recepción utilizando tanto otros equipos SDR, como sintonizadores ISDB-T comerciales. A su vez se muestra la implementación del transmisor one-seg siendo este luego utilizado para las distintas pruebas con el *hardware* alternativo y transmisores SDR. Continuando con el estudio del transmisor ISDB-T, se realizan pruebas con los distintos parámetros configurables de transmisión con el fin de continuar corroborando su correcta implementación. También se explica la codificación de distintos formatos de video según las especificaciones requeridas

[1]Formato definido en la norma ISO/IEC 13818-1 [10], utilizado para la transmisión y almacenamiento de flujos de video, audio y otros datos.

para la norma ISDB-T. Tanto videos previamente codificados para una posterior transmisión, como codificando y transmitiendo videos en tiempo real.

El documento se organiza de la siguiente forma. En el próximo capítulo se brindan algunos conceptos teóricos de temáticas que competen al proyecto, como ser GNU Radio, Radio Definida por Software, entre otros. En el capítulo 3 se cubre la norma ISDB-T explicando tanto conceptos generales de esta, como algunos aspectos particulares que son necesarios para el resto del proyecto. En el capítulo 4 se presenta el trabajo realizado sobre el *software* transmisor ISDB-T, así como algunas pruebas realizadas para validar su funcionamiento. Luego, en el capítulo 5 se explica el proceso de generar videos para que estos puedan ser utilizados para una transmisión bajo la norma ISDB-T. Los capítulos 6 y 7 involucran la teoría, implementación y pruebas de un adaptador USB-VGA como *hardware* transmisor de bajo costo. Posteriormente en el capítulo 8 se muestran pruebas comparativas entre *hardware* SDR convencional y el adaptador USB-VGA en transmisiones ISDB-T, tanto frente a receptores SDR, como frente a sintonizadores ISDB-Tb comerciales. Luego, en los capítulos 9 y 10 se presenta la teoría e implementación de una placa Raspberry Pi como transmisor SDR de bajo costo. Finalmente, en el capítulo 11 se muestran las conclusiones finales del proyecto.

Capítulo 2

Fundamento Teórico

El proyecto que se expondrá presenta una importante carga de trabajo con sistemas de Radio Definida por Software. Tanto el trabajo a nivel de *software*, como los distintos equipos transceptores SDR utilizados, son parte de la estructura necesaria para poder llevar a cabo los distintos objetivos del proyecto. En particular, el *software* GNU Radio es una de las principales herramientas para implementar sistemas SDR y es pertinente brindar una introducción sobre este. Al mismo tiempo, al estar trabajando con un sistema de comunicación digital regido por la norma ISDB-T, es necesario explicar las bases de la modulación ODFM, tanto como la norma en sí.

2.1. Radio Definida por Software

Las radios forman parte de nuestra vida cotidiana. Estas se encuentran integradas en computadoras, televisiones, teléfonos celulares, satélites, entre otros. Desde sus inicios las radios fueron implementadas completamente con electrónica. Esta implementación conlleva las limitaciones inherentes al diseño de radios por *hardware*: poca versatilidad, funcionalidades específicas, rangos de funcionamiento en frecuencia muy acotados. Estas restricciones son resueltas con la aparición del paradigma de Radio Definida por Software (SDR, por sus siglas en inglés *Software Defined Radio*). Esta tecnología permite a partir de un único equipo, diseñar un incontable número de proyectos de radio. Este paradigma se basa fuertemente en el diseño por *software*. La señal recibida es primero bajada a banda base y muestreada por ciertos equipos creados con este fin (comúnmente denominados como SDRs). Las muestras son luego procesadas arbitrariamente por una computadora (o similar). Este procesamiento puede incluir etapas de demodulación, de corrección de errores, entre otros, logrando el objetivo final de reconstruir la señal originalmente enviada (audio, video, texto, etc). El proceso inverso de enviar muestras procesadas de una señal también es posible por medio de este paradigma [11].

Con el mismo equipo se puede lograr un sinnúmero de aplicaciones, desde implementar una radio FM (como ilustra la figura 2.1), hasta aplicaciones más complejas como un receptor/transmisor completo de televisión digital. Al pres-

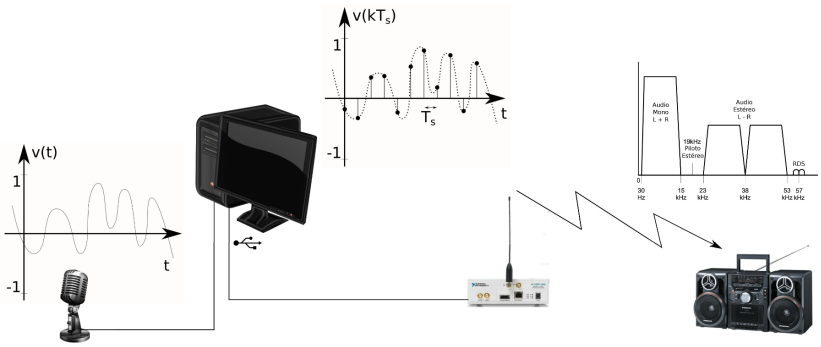


Figura 2.1: Estación de FM [11].

cindir de la cantidad de dispositivos físicos que anteriormente eran necesarios, los proyectos diseñados en este paradigma son escalables, y evitan la necesidad de reemplazar *hardware* en caso de una falla o modificación puntual: las modificaciones necesarias son llevadas a cabo a nivel de *software*. La tabla 2.1 ilustra una pequeña lista de los dispositivos SDR más conocidos en el mercado junto a sus precios en origen.

Nombre	Min. Freq.(MHz)	Max. Freq.(Mhz)	BW(MHz)	Tx	Precio(USD)
RTL-SDR	24	1766	3,2	No	25
SDRplay RSP1A	0,001	2000	10	No	109
Airspy Mini	24	1700	6	No	99
ADALM-PLUTO	325	3800	20	Si	249
HackRF	30	30	20	Si	300
BladeRF	300	3800	40	Si	400-650
USRP B200	70	6000	56	Si	885
MatchStiq	300	3800	28	Si	4500

Tabla 2.1: Dispositivos SDR más conocidos en el mercado.

Dentro de los más conocidos, el RTL-SDR (el cual se puede observar en la figura 2.2) se destaca debido a su accesible precio. Es comúnmente utilizado para ensayos en donde no se requieren grandes anchos de banda. Sin embargo los 3,2 MHz que ofrece alcanza para muchos casos de uso. El bajo precio de este dispositivo es motivo por el cuál los aficionados a las comunicaciones inalámbricas suelen iniciarse con él. Cabe destacar que este dispone de un conversor analógico digital de tan solo 8 bits de resolución. Este equipo será utilizado a lo largo del proyecto solo para hacer recepciones de señales ISDB-T one-seg, ya que no es capaz de soportar todo el ancho de banda de una señal full-seg, ni de hacer transmisiones de ningún tipo.

Para escenarios donde se necesite analizar señales más complejas, de un mayor ancho de banda, se puede optar por otras soluciones. Ettus ofrece una linea de

Figura 2.2: RTL-SDR [12].

Figura 2.3: USRP B200 [13].

dispositivos de altas prestaciones. Entre ellos se encuentra el USRP B200 (el cual se puede observar en la figura 2.3), dispositivo que soporta un ancho de banda de 56 MHz, y un rango de frecuencias de trabajo de entre 70 MHz y 6000 MHz. También es posible de utilizar como transmisor SDR de señales, y cuenta con un conversor analógico digital de 12 bits de resolución. A lo largo del proyecto este equipo será utilizado tanto para recibir señales ISDB-T full-seg y one-seg, como para realizar transmisiones también en esas dos modalidades.

Si bien los dispositivos físicos de transmisión y recepción son esenciales, el paradigma de la Radio Definida por Software se centra en la posibilidad de acondicionar la señal mediante el uso de *software*. Para ello, hoy en día el *software* GNU Radio es ampliamente el más utilizado para desarrollo de SDRs.

2.2. GNU Radio

GNU Radio [14] es un entorno de desarrollo libre y abierto, el cuál provee herramientas de procesamiento de señales permitiendo implementar radios definidas por *software*. El procesamiento se da a través de la concatenación de elementos llamados bloques. Los mismos pueden ser incorporados de una amplia gama de bloques funcionales que trae la aplicación por defecto o pueden ser desarrollados por el usuario. Los bloques disponibles implementan elementos estándar, como lo son filtros, codificadores de canal, ecualizadores, demoduladores, etc.

GNU Radio trabaja con flujos de datos constantes que transitan a través de varios bloques de procesamiento, desde un origen hasta un final. Los tipos de datos pueden variar, siendo los disponibles: bytes, números enteros o complejos. Los bloques tienen diferentes funcionalidades, que pueden variar en realizar operaciones sobre las muestras, exportar datos hacia fuera del programa o del entorno, o puede tener toda una estructura de bloques dentro. En el diseño de bloques, GNU Radio facilita su desarrollo ofreciendo esquemas armados de código a los que se los modifica para lograr procesar los datos de la manera deseada. Existen cuatro tipos de bloques predefinidos, y se diferencian por la relación entre ítems de entrada y salida: Bloques Sincrónicos, Bloques Decimadores, Bloques Interpoladores y Bloques Generales.

- Bloques Sincrónicos: Son utilizados cuando la cantidad de elementos salientes es igual al número de elementos entrantes.
- Bloques Decimadores: El número de ítems de entrada es un múltiplo de la cantidad de elementos salientes.
- Bloques Interpoladores: El número de ítems de salida es múltiplo de la cantidad de ítems de entrada.
- Bloques Generales: No especifica una relación entre entradas y salidas. Los tipos de bloques son simplificaciones de este tipo de bloque general.

Los bloques pueden ser desarrollados en código Python o en C++, aunque el más utilizado es este último. Esto permite implementar sistemas que funcionan en tiempo real, por más que requieran del procesamiento de grandes tasas de datos. Cada bloque provee una funcionalidad específica. Al ser agrupado con otros se conforma una estructura compleja con la cuál se pueden consumar distintos objetivos, como pueden ser distintas modulaciones, transmisores de radio, televisión, etc. GNU Radio Companion es una aplicación que permite construir mediante una interfaz gráfica la interconexión entre los bloques, así como configurar cada uno de estos. En particular el trabajo expuesto en este documento se realiza en parte en este entorno. En la figura 2.4 se observa el diagrama de bloques o *flowgraph* de un demodulador AM.

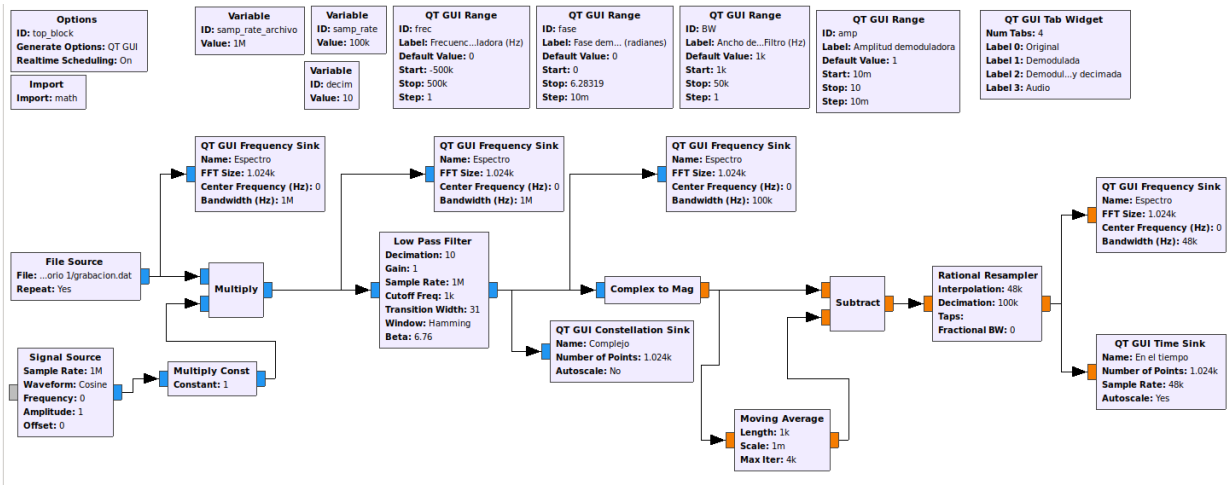


Figura 2.4: Diagrama de bloques de un demodulador AM implementado en GNU Radio Companion [15].

2.3. Modelo de un canal inalámbrico

El canal es parte esencial de un sistema de comunicación inalámbrico. La señal transmitida puede sufrir distorsiones generadas por variados fenómenos físicos en el camino entre el transmisor y el receptor. Estos tienen que ser tomados en cuenta a la hora de diseñar el receptor, ya que estos efectos tendrían que intentar ser contrarrestados en recepción. El modelado de un canal inalámbrico permite predecir en cierta medida algunos de estos fenómenos que van a ocurrir en una comunicación de este tipo.

Para el siguiente análisis se asumirá una configuración de dos antenas, transmisor y receptor respectivamente, ambas mantenidas en posiciones fijas, trabajando en campo lejano y en régimen sinusoidal. La forma de onda recibida en bornes de una antena receptora, ante el campo generado por una antena transmisora trabajando a frecuencia f, y separadas una distancia r entre ellas, se puede expresar de la siguiente forma:

$$g(\theta, \phi, f)\frac{e^{j2\pi f(t-r/c)}}{r}. \tag{2.1}$$

La función $g(\theta, \phi, f)$ describe los patrones de radiación del transmisor y receptor, así como las pérdidas, entre otros. Estos factores son dependientes de la frecuencia, y de la posición relativa de las antenas (θ, ϕ). Se define la siguiente función:

$$H(f) = g(\theta, \phi, f)\frac{e^{-j2\pi fr/c}}{r}. \tag{2.2}$$

Siendo $e^{j2\pi ft}$ la forma de la señal emitida desde bornes de la antena transmisora, se puede afirmar según la ecuación 2.1 que en bornes de la antena receptora se tendrá la siguiente respuesta:

$$H(f)e^{j2\pi ft}. \tag{2.3}$$

Al ser la radiación electromagnética un sistema lineal, la respuesta a una combinación lineal de sinusoidales de entrada de la forma: $x(t) = \int X(f)e^{j2\pi ft}$ será también una combinación lineal de sinusoidales según la siguiente expresión:

$$y(t) = \int H(f)X(f)e^{j2\pi ft}df. \tag{2.4}$$

Aplicando la transformada de Fourier a la respuesta, se obtiene que $Y(f) = H(f)X(f)$, por lo cual la señal en el tiempo será:

$$y(t) = \int x(\tau)h(t-\tau)d\tau. \tag{2.5}$$

Finalmente, se puede afirmar que ante dos antenas transmisor y receptor fijas, el canal se comporta como un sistema lineal e invariante en el tiempo, y la transferencia se expresa:

$$H(f) = g(\theta, \phi, f)\frac{e^{-j2\pi fr/c}}{r}. \tag{2.6}$$

Este es un modelo que si bien es sencillo, las expresiones 2.4 y 2.5 se pueden aplicar a casi cualquier situación que nos encontramos en la práctica, con la salvedad de que muchas veces $H(f)$ también dependerá del tiempo (y también incluye ruido). Más detalles sobre el modelado de canal se pueden encontrar en la referencia bibliográfica de esta sección [16]. De todas formas, este será el modelo que se usará a continuación para analizar OFDM.

2.4. Sistemas OFDM

En esta sección se explicará el esquema de modulación OFDM, utilizado por diversas aplicaciones como la radio digital, telefonía 4G, estándar 802.11n/ac y los estándares de televisión digital terrestre DVB-T, DVB-T2 e ISDB-T entre otros. OFDM es un esquema de modulación multiportadora el cual se basa en dividir el flujo de bits que ingresan al sistema en N-subflujos, de forma que los bits se envíen en paralelo en el tiempo y no secuencialmente. La principal ventaja de este esquema de modulación es su capacidad para enfrentar condiciones severas del canal, como por ejemplo el desvanecimiento selectivo en frecuencia.

Se puede observar en la figura 2.5 un esquema de transmisión OFDM. La señal OFDM es susceptible a interferencia intersimbólica (ISI) producto de la propagación por múltiples caminos. Para resolver este problema se introduce el prefijo

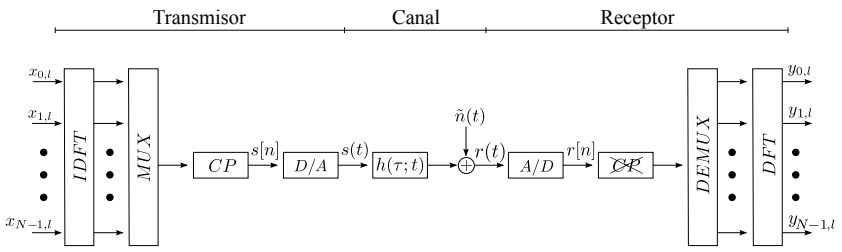


Figura 2.5: Esquema básico de un sistema OFDM [17].

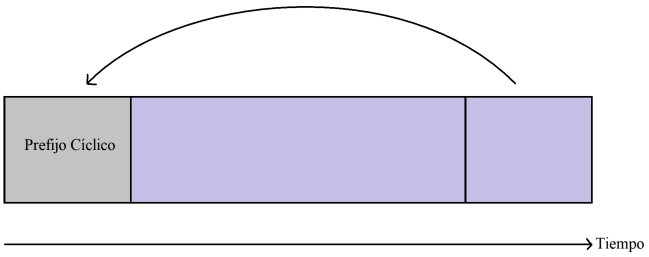


Figura 2.6: El prefijo cíclico es una copia de la ultima parte del símbolo OFDM [17].

cíclico. Esto es, anexar una copia del tramo final de cada símbolo en su comienzo como se observa en la figura 2.6. Con ello, se resuelve el ISI provocado por el multicamino además de agregar ortogonalidad perfecta entre subportadoras, necesarias en sistemas OFDM. El prefijo cíclico es introducido y removido en los extremos de la comunicación, como se observa en la figura 2.5.

De esta manera, aunque llegue parte del símbolo anterior, estará afectando una parte del símbolo actual que no es importante. Se consigue que la señal transmitida se comporte respecto al canal en frecuencia como una convolución circular. Esto significa que si introducimos en recepción la Transformada de Fourier discreta, podremos obtener una salida igual a la multiplicación de la respuesta en frecuencia del canal y los símbolos transmitidos.

2.4.1. Modelo OFDM para tiempo continuo

OFDM se basa principalmente en el concepto de transformada de Fourier de una ventana temporal. En un sistema OFDM, a cada símbolo complejo (*constellation point*) se lo multiplica por una ventana rectangular y una exponencial de la forma $e^{\frac{j2\pi nt}{\tau}}$, con n entre cero y N (siendo N la cantidad de subportadoras). Los N resultados de los productos son sumados y enviados por el canal. De esta manera se consigue que el conjunto de señales a enviar sea una señal que se comporte en frecuencia como un conjunto de funciones *sinc* ortogonales. El valor de continua de cada *sinc* no se verá afectado por los demás.

En la figura 2.7 se puede ver un sistema OFDM en tiempo continuo. Como se mencionó anteriormente, a cada símbolo de entrada se lo multiplica por una función. Sea esta función ϕ_K, la definimos de la siguiente manera:

$$\phi_K(t) = \begin{cases} \frac{1}{\sqrt{T-T_{CP}}}.e^{j2\pi\frac{W}{N}k(t-T_{CP})} & t \in [0,T] \\ 0 & t \notin [0,T] \end{cases} \tag{2.7}$$

En donde T es la duración total de un símbolo OFDM incluido el prefijo cíclico, T_{CP} la duración del prefijo cíclico, $W = \frac{N}{T-T_{CP}}$ es el ancho de banda de la señal y N la cantidad de portadoras.

Con la anterior definición, la señal obtenida para el símbolo OFDM l-ésimo,

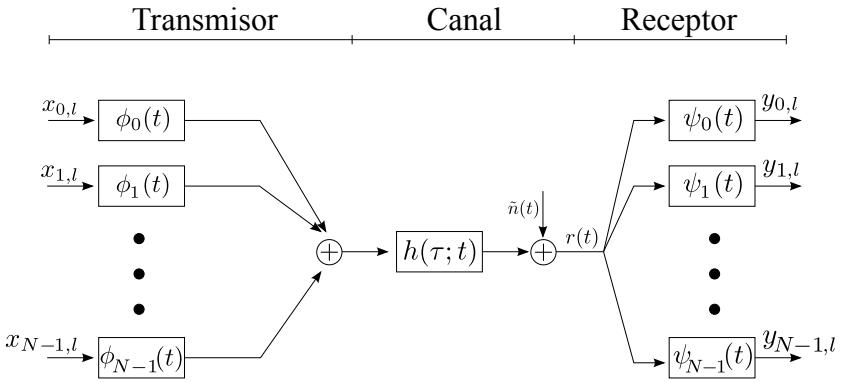


Figura 2.7: Sistema OFDM ideal continuo en banda base [17].

será de la forma:

$$s_l(t) = \sum_{n=0}^{N-1} x_{k,l}\phi_K(t - lT).$$

Al enviar una secuencia infinita de símbolos OFDM, podemos expresar la señal resultante como sigue.

$$s(t) = \sum_{l=-\infty}^{l=\infty} s_l(t) = \sum_{l=-\infty}^{l=\infty}\sum_{k=0}^{N-1} x_{k,l}\phi_K(t - lT).$$

En el receptor de un sistema $OFDM$, se cuenta con un banco de filtros sincronizando con la última parte de cada símbolo $OFDM$,

$$\psi_K(t) = \begin{cases} \phi_K^*(T - t) & t \in [0, T - T_{CP}] \\ 0 & t \notin [0, T - T_{CP}] \end{cases} \tag{2.8}$$

Esto significa que la parte del prefijo cíclico es removida. La respuesta al impulso del canal está restringida a la duración del prefijo cíclico. Al multiplicar por ϕ_K y luego por su conjugada, se logra que la señal sea multiplicada por una base ortonormal y su conjugada, no afectando su energía. Sea r la señal recibida luego de pasar por el canal, h la respuesta al impulso del canal, $\tilde{n}$ ruido blanco y Gaussiano, complejo y aditivo y s la señal transmitida. Se tiene que:

$$r(t) = (h * s)(t) + \tilde{n}(t) = \int_0^{T_{CP}} h(\tau;t)s(t - \tau)d\tau + \tilde{n}(t). \tag{2.9}$$

La señal recibida será entonces:

$$y_k = (r * \psi_k)(t)|_{t=T} = \int_{-\infty}^{\infty} r(t)\psi_k(T - t)dt$$

$$y_k = \int_{T_{CP}}^{T} \left(\int_0^{T_{CP}} h(\tau;t) \left[\sum_{k'=0}^{N-1} x_{k'}\phi_{k'}(t - \tau) \right] d\tau \right) \phi_k^*(t)dt + \int_{T_{CP}}^{T} \tilde{n}(T - t)\phi_k^*(t)dt.$$

Si se asume que la respuesta en frecuencia del canal es constante durante toda la duración del símbolo OFDM, se tiene que $h(\tau;t) = h(\tau)$, y por lo tanto:

$$y_k = \sum_{k'=0}^{N-1} x_{k'} \int_{T_{CP}}^{T} \left(\int_0^{T_{CP}} h(\tau) \times \phi_{k'}(t-\tau) d\tau \right) \phi_k^*(t) dt + \int_{T_{CP}}^{T} \tilde{n}(T-t)\phi_k^* dt.$$

Como los límites de integración son $T_{CP} < t < T$ y $0 < \tau < T_{CP}$, se puede concluir que $0 < t - \tau < T$. Las tres relaciones anteriores aseguran que la siguiente integral está contenida dentro del soporte de ϕ_k, por lo que su cálculo es directo,

$$\int_0^{T_{CP}} h(\tau)\phi_{k'}(t-\tau)d\tau = \int_0^{T_{CP}} h(\tau) \frac{e^{j2\pi \frac{W}{N} k'(t-\tau-T_{CP})}}{\sqrt{T-T_{CP}}} d\tau$$

$$= \frac{e^{j2\pi \frac{W}{N} k'(t-T_{CP})}}{\sqrt{T-T_{CP}}} \int_0^{T_{CP}} h(\tau) e^{-j2\pi \frac{W}{N} k'\tau} d\tau. \tag{2.10}$$

La última integral de la ecuación 2.10 es la respuesta en frecuencia del canal de radiofrecuencia, evaluada en $f = k'\frac{W}{N}$; o lo que es lo mismo, evaluada en la portadora k'-ésima. Sea $H(f)$ la Transformada de Fourier de $h(\tau)$,

$$H_{k'} = H\left(k'\frac{W}{N}\right) = \int_0^{T_{CP}} h(\tau) e^{-j2\pi \frac{W}{N} k'\tau} d\tau. \tag{2.11}$$

Utilizando las ecuaciones 2.10 y 2.11 es posible simplificar la salida del banco de filtros a la forma:

$$y_k = \sum_{k'=0}^{N-1} x_{k'} \int_{T_{CP}}^{T} \frac{e^{j2\pi \frac{W}{N} k'(t-T_{CP})}}{\sqrt{T-T_{CP}}} H_{k'} \phi_k^* dt + \int_{T_{CP}}^{T} \tilde{n}(T-t)\phi_k^* dt$$

$$= \sum_{k'=0}^{N-1} x_{k'} H_{k'} \int_{T_{CP}}^{T} \phi_{k'}(t)\phi_k^*(t) dt + n_k.$$

Como los filtros $\phi_k(t)$ son ortogonales entre sí,

$$\int_{T_{CP}}^{T} \phi_{k'}(t)\phi_k^*(t) dt = \delta[k-k'],$$

donde $\delta[n]$ es la función Delta de Kronecker. Operando con la ecuación anterior, se obtiene que la salida y_k puede ser escrita como:

$$y_k = H_k x_k + n_k, \tag{2.12}$$

siendo x_k el símbolo k-esimo transmitido y n_k el ruido que el canal agregó a ese k-esimo símbolo.

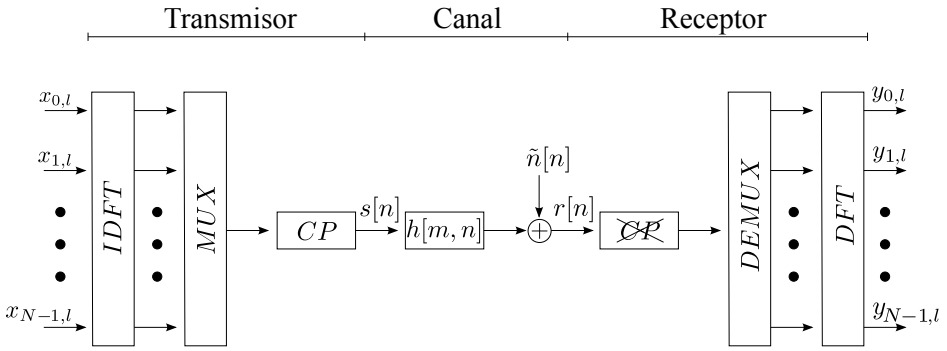


Figura 2.8: Modelo de sistema OFDM discreto [17].

2.4.2. Modelo OFDM para tiempo discreto

En la figura 2.8 se puede observar el modelo OFDM para tiempo discreto. La modulación y demodulación de las portadoras ortogonales enventanadas temportalmente, $\phi_k(t)$, puede reemplazarse por el cómputo de la antitransformada discreta de Fourier en transmisión (IDFT) y de la transformada discreta de Fourier en recepción (DFT). El efecto del canal se ve representado por una convolución discreta.

Desde el punto de vista del receptor, utilizando un prefijo cíclico más largo que la respuesta al impulso del canal, transforma a la convolución lineal entre el canal y la señal transmitida en una convolución cíclica ($\circledast$). Es posible escribir entonces el sistema OFDM según:

$$y_l = DFT(IDFT(x_l) \circledast h_l + \tilde{n}_l) = DFT(IDFT(x_l) \circledast h_l) + n_l, \tag{2.13}$$

donde y_l contiene los N puntos recibidos, x_l los N puntos de constelación transmitidos, h_l la respuesta al impulso del canal y $\tilde{n}_l$ el ruido introducido por el canal. $n_l = \mathrm{DFT}(\tilde{n}_l)$ representa el no correlacionado ruido gaussiano.

Para las convoluciones cíclicas la DFT de la convolución de dos señales es igual al producto de las DFT individuales, logrando el siguiente resultado.

$$y_l = x_l \times DFT(h_l) + n_l = H_l \times x_l + n_l, \tag{2.14}$$

donde H_l es la respuesta en frecuencia del canal durante el símbolo l-ésimo.

Sea N_{CP} la cantidad de muestras al final del símbolo OFDM replicadas al comienzo para lograr el prefijo cíclico. Si se supone que el largo de la respuesta al impulso del canal es exactamente igual a N_{CP} y se entiende que en un caso real se tienen símbolos concatenados uno detrás del otro, se concluye que las primeras N_{CP} muestras de $IDFT(x_l) \circledast h_l$ están corrompidas por ISI, asociada a las últimas N_{CP} muestras del símbolo anterior. De esta manera, la utilización del prefijo cíclico sirve para eliminar la ISI ocasionada por el multicamino. Como contraparte de agregar un prefijo cíclico aparece una reducción del *bitrate* útil de $\frac{N}{N+N_{CP}}$. Por lo cuál resulta importante elegir el largo adecuado para combatir la ISI ocasionada por el multicamino, pero que no reduzca el *bitrate* útil más de lo necesario. Por

más detalles sobre OFDM se recomienda la lectura de la referencia bibliográfica de esta sección [17].

Con los conceptos sobre OFDM expresados en este capítulo, se procederá a explicar en el siguiente capítulo la norma ISDB-T, norma que utiliza OFDM como esquema de modulación.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 3

ISDB-T

3.1. Introducción

ISDB es un conjunto de normas de transmisión japonesa de televisión digital y radio digital. Está compuesta por una familia de componentes, entre ellos ISDB-T para transmisión terrestre, ISDB-S para transmisión satelital, ISDB-C para transmisión por cable. De aquí en adelante nos centraremos en la norma ISDB-T (*terrestrial*).

En este capítulo se introducen los fundamentos de ISDB-T, etapas de procesamiento de un transmisor y consideraciones en la codificación de video y audio. El entendimiento teórico de la norma es base para lograr los objetivos de este proyecto.

3.2. Consideraciones generales

Para la transmisión, ISDB-T utiliza un esquema de modulación OFDM (introducido en la sección 2.4). En este contexto, los valores de prefijo cíclico que admite son: $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$. El estándar define tres modos de operación: modo 1, 2 y 3. La cantidad de portadoras utilizadas por símbolo OFDM varía en función de este según $1024 \times 2^{\text{modo}}$. Para obtener anchos de banda y *bitrate* constantes independientemente del modo de transmisión, el tiempo de símbolo aumenta de forma proporcional al número de portadoras, obteniendo:

- Modo 1 (2k) , 2048 portadoras, tiempo de símbolo 252μs
- Modo 2 (4k) , 4096 portadoras, tiempo de símbolo 504μs
- Modo 3 (8k) , 8192 portadoras, tiempo de símbolo 1008μs

La norma ISDB-T divide el espectro en 14 porciones idénticas llamadas segmentos, de los cuales 13 son utilizados para mandar datos, mientras que la porción restante se deja de guarda a los lados del espectro. La figura 3.1 ilustra como se disponen los segmentos en el canal de 6 MHz. A su vez, la segmentación permi-

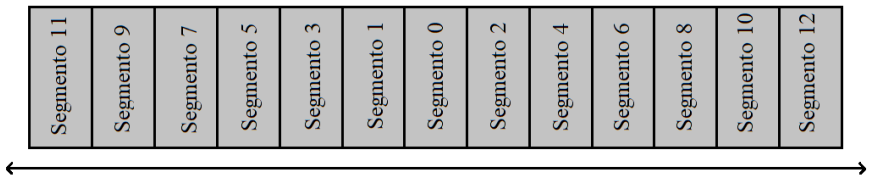


Figura 3.1: Canal de 6MHz.

te asignar varios segmentos a un servicio determinado, y ajustar los parámetros de transmisión individualmente de acuerdo a las necesidades. De esta manera se definen las capas jerárquicas A, B y C, las que permiten enviar simultáneamente diferentes flujos de transporte con distintos parámetros de transmisión. Teniendo en cuenta lo anterior, es común en ISDB-T que el segmento central constituya el segmento conocido como one-seg. Al segmento one-seg se lo configura con características robustas con el fin asegurar su correcta recepción sin sufrir alteraciones. Como consecuencia este segmento tiene menor *bitrate* respecto a los otros. Se detallará sobre este tema más adelante.

De ahora en adelante para ejemplificar se trabajará en modo 3, no obstante el razonamiento es análogo para los demás modos de trabajos. Teniendo en cuenta la cantidad de portadoras por símbolo OFDM, y la duración del símbolo, podemos obtener la frecuencia de muestreo de ISDB-T como:[1]

$$\frac{8192}{1008\mu s} = 8{,}127 \text{ MHz}. \tag{3.1}$$

Sin embargo, para canales terrestres se define el uso de solo 6 MHz de ancho de banda. Esto significa que no es posible transmitir información con la totalidad de las portadoras de un símbolo OFDM. Sino con solo una fracción de ella, de tal manera de que el ancho de banda de la señal no sea mayor al anteriormente definido. Para obtener la cantidad de portadoras que llevan información, se calcula según 3.2.

$$6 \text{ MHz} \times 1008\mu s = 6048 \text{ portadoras}, \tag{3.2}$$

y utilizando 13 de los 14 segmentos disponibles:

$$13 \times \frac{6048}{14} = 5616 \text{ portadoras}, \tag{3.3}$$

resultando en 432 portadoras por segmento, y 5617 portadoras utilizadas por símbolo OFDM (se suma un piloto continuo a la derecha). El resto de las portadoras del símbolo estarán vacías y quedarán dispuestas al comienzo y final de este, de manera de no alterar el espectro. A este proceso se lo conoce como *zero-padding*.

Por otro lado, para facilitar la sincronización del receptor, el estándar define los cuadros OFDM implementados como forma de delimitar los datos transmitidos a efectos de poder procesarlos. Un cuadro OFDM consta de 204 símbolos OFDM.

[1] Aunque en la ecuación se utilizan la cantidad de portadoras y tiempo de símbolo de modo 3, el resultado es el mismo para cualquiera de los modos.

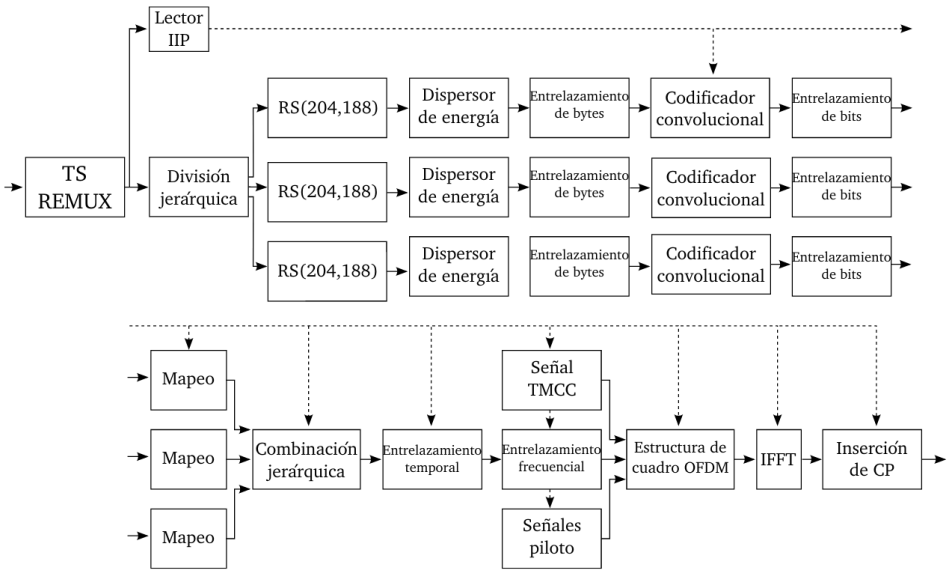


Figura 3.2: Sistema transmisor ISDB-T [18].

3.3. Transmisor ISDB-T

La figura 3.2 presenta un diagrama del transmisor ISDB-T. El diagrama expone una serie de etapas por las que el contenido multimedia es adaptado a la norma. Estas etapas involucran manejo de flujo de datos, robustecimiento de la señal, conversión en una estructura de cuadro OFDM y la etapa de transmisión por el medio físico.

3.3.1. Broadcast Transport Stream

El esquema de transmisión parte de un flujo de transporte llamado BTS (*Broadcast Transport Stream*). En una trama BTS se combinan diversos flujos TS (*Transport Stream*) por medio del bloque multiplexador TS REMUX que se visualiza en la figura 3.2. Cada TS está conformado por paquetes denominados TSP (*Transport Steam Packet*), cada uno conteniendo 188 bytes de información. Los TSP al ser multiplexados en una BTS pasan por un proceso de codificación agregando 16 bytes denominados *dummy bytes*. Los primeros 8 bytes del mismo son para identificar la información de control llamada *ISDB-T Information*, el cuál informa por ejemplo a qué capa jerárquica pertenece el TSP (entre otras cosas). Los restantes 8 bytes se configuran como código corrector de errores Reed Solomon (204, 196), con una capacidad de corrección de errores de hasta 4 bytes en un TSP. En la figura 3.3 se puede observar la estructura de un paquete TSP. Las tramas de una BTS se transmiten a una tasa constante de $32,5079$ Mbps.

A los flujos de transporte multiplexados en la BTS, se le agrega un TSP llamado *ISDB-T Information Packet* (IIP). Dentro del IIP se encuentra el campo

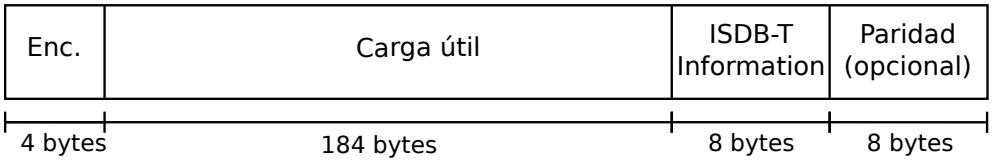


Figura 3.3: Formato de un TSP [18]

Modulation Control Configuration Information (MCCI). Este tiene un largo de 160 bits y contiene la configuración del sistema así como el modo de transmisión y prefijo cíclico utilizado. De él se extrae la *TMCC Information*, información que es tomada por el transmisor para su configuración y también enviada al aire a través de determinadas portadoras. La información aportada por estas portadoras será detallada más adelante en este texto. De esta manera el receptor lee la información del TMCC y puede decodificar la señal sin necesidad de configuraciones manuales. Por cada *multiplex frame* corresponde multiplexar un paquete IIP, siendo un *multiplex frame* la cantidad de TSPs que entrega el TS REMUX en la duración de un cuadro OFDM.

3.3.2. Etapas de procesamiento de un modulador ISDB-T

En esta sección se explicará brevemente cada una de las etapas del transmisor de la figura 3.2 en forma secuencial. La BTS es separada al comienzo del transmisor en la "División Jerárquica" en tres flujos independientes correspondientes a cada capa jerárquica A, B o C. Se toman 204 bytes de cada flujo y se encaminan hacia un codificador Reed Solomon (204,188). Este reemplaza los últimos 16 bytes por la palabra formada a partir del código Reed Solomon, con la cual el receptor es capaz de corregir hasta 8 bytes por paquete.

Cambios de flancos son utilizados comúnmente en recepción para obtener información de sincronismo a nivel de bits. Es importante evitar ráfagas de ceros y unos consecutivos para no perder esta información. Por ello, el flujo resultante de datos pasa por un "Dispersor de Energía", obteniendo una secuencia pseudoaleatoria de bits. Con este proceso además se logran símbolos equiprobables y se elimina el nivel de continua de la señal. El dispersor de energía se implementa a través de un XOR bit a bit entre la señal de entrada (sin contar al byte de sincronismo) y una secuencia de bits pseudoaleatoria conocida PRBS (*Pseudorandom binary sequence*). Tanto el transmisor como el receptor deben estar sincronizados en la evolución de la palabra, debido a lo cual se reinicia el circuito generador en cada inicio de cuadro.

Luego de la dispersión de energía, se hace un entrelazado de bytes con el fin de que al producirse una falla en ráfaga en el canal, afecte a distintas partes del flujo de información. Con esto se permite que potenciales pérdidas en ráfagas sean puntualmente corregidas a partir del código Reed Solomon. Es necesario tener en cuenta que procesamientos como este generan distintos atrasos para las capas jerárquicas que pueden perjudicar la sincronización. Por esto se realiza un ajuste de atraso en esta etapa para lograr que el retardo total sea igual al período de un

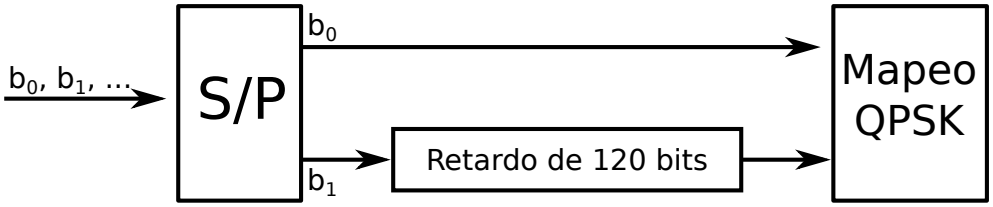


Figura 3.4: Diagrama del sistema para el entrelazamiento de bit y la modulación QPSK [18].

cuadro OFDM.

Posteriormente se utiliza un código Viterbi, con el cual los datos digitales son protegidos agregando redundancia. Además, este permite obtener distintas tasas de código a través del puncturing[2] con un código madre de tasa $\frac{1}{2}$ y constante $k = 7$. Los polinomios generadores valen: $G1 = 171_{OCT}$ y $G2 = 133_{OCT}$. A través del puncturing podemos obtener tasas de código convolucional: $\frac{1}{2}$, $\frac{2}{3}$, $\frac{3}{4}$, $\frac{5}{6}$ y $\frac{7}{8}$.

Al código convolucional le sigue un mapeo y entrelazado a nivel de bits. Para el mapeo se agrupa el flujo de datos de a 2, 4 o 6 bits según el esquema de modulación utilizado, QPSK, 16QAM o 64QAM respectivamente. A su vez, se normaliza la constelación obtenida de manera de obtener potencia media de los símbolos OFDM igual a uno. El factor de normalización es conocido y depende del esquema de modulación utilizado. Para el entrelazado se retienen los distintos bits de un mismo grupo con diferentes retardos. Por ejemplo, en QPSK se agrupa el flujo de a 2 bits, de los cuales el primero es enviado y el segundo se atrasa 120 bits respecto al primero como se puede observar en la figura 3.4. Durante el entrelazado a nivel de bits se debe realizar nuevamente un ajuste de atraso. El retardo propio del proceso de entrelazamiento sumado al ajuste de atraso debe totalizar dos símbolos OFDM.

Hasta este punto el procesamiento se aplicaba a cada capa jerárquica de manera independiente, sin embargo, estos flujos paralelos convergen a la etapa de combinación jerárquica. Esta multiplexa los flujos de entrada en uno solo, el cual contiene sus segmentos ordenados de forma creciente (del uno al trece). A este flujo resultante se le aplica primero un entrelazado temporal y luego uno frecuencial.

El entrelazado temporal resulta de retardar cada portadora de un mismo segmento de manera independiente. Este retardo es logrado almacenando de forma secuencial distintas portadoras en un *buffer* previo a enviarlo. El proceso se realiza de forma independiente para cada capa jerárquica. Teniendo en cuenta necesidades para la sincronización, es importante introducir un ajuste de atraso tal que el retardo total en transmisión y recepción sea múltiplo de un cuadro OFDM, para todos los niveles jerárquicos.

El entrelazado frecuencial tiene como objetivo robustecer el flujo de información frente a un canal selectivo en frecuencia. En esta etapa existe entrelazamiento inter-segmentos e intra-segmentos. El entrelazado inter-segmento se basa en intercalar las portadoras entre los segmentos que comparten la modulación. Durante este

[2]El puncturing es una técnica utilizada para lograr códigos de cualquier tasa $\frac{m}{n}$ en transmisión, sin aumentar la complejidad de la decodificación en recepción.

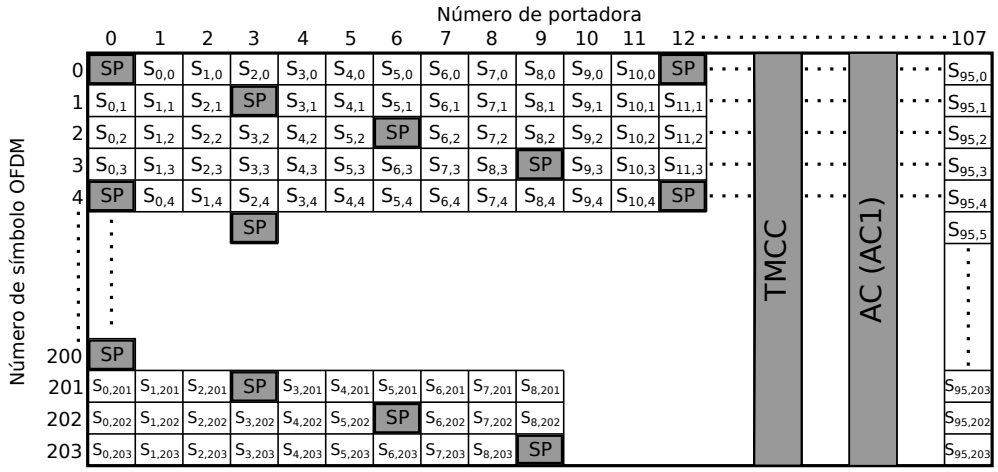


Figura 3.5: Estructura de cuadro ISDB-T en modo de transmisión 1 [18].

entrelazado no interviene el segmento correspondiente a one-seg. El entrelazado intra-segmento integra dos etapas:

- Rotación: Se basa en rotar a las portadoras según la siguiente ecuación,

$$S_{i,k} = S_{(k+i)modC,k}, \tag{3.4}$$

  donde i representa el número de portadora (dentro del segmento), k el número de segmento correspondiente y C $= 96 \times 2^{modo-1}$ la cantidad de portadoras por segmento. El parámetro i toma valores entre [0, C-1] y k toma valores entre cero y el número de segmentos que utiliza cierta modulación, menos uno.

- Aleatorización de símbolos: Se logra al disponer a las portadoras en un orden pseudo-aleatorio, conocido para cada modo de transmisión.

La estructura de cuadro OFDM no solo cuenta con portadoras relacionadas al *transport stream*, sino que ciertas portadoras llevan otro tipo de información. Estas se encuentran en posiciones conocidas, y tienen su propio esquema de modulación. Luego del entrelazado frecuencial, se adiciona al flujo de transporte las siguientes portadoras:

- Portadoras TMCC (*Transmission Multiplexing Configuration Control*): Transportan la información relativa a las capas jerárquicas (modulación, cantidad de segmentos, profundidad del entrelazado temporal, tasa de código convolucional, etc.) así como información de control y sincronismo. Dada la importancia de las portadoras TMCC, el transmisor las repite a lo largo de un mismo símbolo OFDM en posiciones conocidas, moduladas en DB-PSK. En recepción, para lograr reconstruir la información de la TMCC se esperarán los 204 símbolos OFDM y en función de las coincidencias de cada portadora recibida, interpretará la información de la TMCC.

- Scattered Pilots: Moduladas en BPSK, se generan a partir de una secuencia pseudoaleatoria PRBS. Estos pilotos tienen como objetivo la estimación del canal.

- Portadoras auxiliares (AC pilots): Son moduladas en DBPSK. Su uso es opcional, contienen información auxiliar de control.

- Piloto continuo: Se agrega una portadora modulada en BPSK a la derecha de la totalidad de portadoras activas con fines únicamente de sincronismo.

Llamaremos **portadoras activas** a todas las portadoras no nulas, y **portadoras de datos** al subconjunto de las portadoras activas que solamente contienen datos del *transport stream*. Para el modo de transmisión 1, se pueden observar las posiciones de las portadoras activas en la figura 3.5. Luego de la etapa de asignación de portadoras TMCC y pilotos, se agregan los ceros necesarios para construir la señal enviada a la IFFT. Este proceso constituye el *zero-padding* introducido en la sección 3.2. La cantidad de ceros agregados, dada en la tabla 3.1, dependerá del modo en el que trabaje el modulador.

Modo	Nz	A la izquierda	A la derecha
Modo 1	643	322	321
Modo 2	1287	644	643
Modo 3	2575	1288	1287

Tabla 3.1: Ceros necesarios para el *zero-padding* de la IFFT [18].

3.4. Codificación de video y audio en ISDB-Tb

3.4.1. ABNT NBR 15602

Se encuentra necesario explicar algunas directivas para la codificación de video y audio en ISDB-Tb, ya que en el capítulo 5 se ahondará en el proceso de generación de *transport streams* para que estos sean transmitidos según la norma. Se decidió profundizar en ISDB-Tb en vez de la variante japonesa, debido a que ISDB-Tb es la norma que se usa en nuestro país. Tanto los operadores que emiten localmente, como los sintonizadores que se venden en plaza, se rigen según esta. La norma "ABNT NBR 15602" consiste en tres documentos que definen la codificación de audio, codificación de video y multiplexación a utilizar en transmisiones ISDB-Tb. "ABNT NBR 15602-1" [19] define la codificación de video, "ABNT NBR 15602-2" [20] la codificación de audio y "ABNT NBR 15602-3" [21] el sistema de multiplexación.

	Resolución de luminancia	Razón de aspecto
SD	720 x 480	4:3
		16:9
	720 x 576	4:3
		16:9
HD	1280 x 720	16:9
	1920 x 1080	16:9

Tabla 3.2: Resoluciones y razón de aspecto para *codec* H.264 en full-seg.

Codecs de video compatibles

En ISDB-Tb se define la compatibilidad únicamente con el *codec MPEG-4 AVC Standard* (Norma ITU-T H.264 [22]). Para este *codec* se definen los posibles parámetros a utilizar para la codificación. Estos contemplan la resolución, tasa de cuadros por segundo, sistema de escaneo, relación de aspecto, así como otros parámetros específicos. Además, se diferencian los parámetros según si el servicio es full-seg o one-seg. H.264 define dos parámetros adicionales para la codificación. Estos son el perfil y el nivel. El perfil de un *codec* refiere al conjunto de capacidades que serán utilizadas a la hora de codificar. El perfil *Baseline* en H.264, utiliza las herramientas de codificación más básicas que H.264 otorga. Esto permite un menor costo computacional para los procesos de codificación y decodificación, siendo ideal su uso en dispositivos móviles o de baja capacidad de procesamiento. Los perfiles *Main* y *High* agregan progresivamente mayores niveles de complejidad en sus algoritmos logrando mejores niveles de calidad y eficiencia en la codificación y decodificación, a costa de la necesidad de mayor capacidad de procesamiento. El nivel permite realizar un ajuste más fino a los perfiles seleccionados. Por ejemplo, utilizar el nivel 1, el nivel más bajo en el perfil *Baseline*, codificará a menor *bitrate* y menor resolución.

Full-seg

En la tabla 3.2 se definen las resoluciones compatibles, junto con la razón de aspecto en cada una. Luego en la tabla 3.3 se presentan las tasas de cuadros compatibles, así como el sistema de escaneo. Para un servicio full-seg se permite el uso de los perfiles *Main* o *High*. Respecto a los niveles, se puede utilizar como máximo el nivel 4 para estos perfiles.

One-seg

Para transmisiones one-seg, se puede observar en la tabla 3.4 las resoluciones permitidas junto a la razón de aspecto correspondiente. Se permiten los siguientes valores de tasa de cuadros: 5/1001 Hz, 10/1001 Hz, 12/1001 Hz, 15/1001 Hz, 24/1001 Hz y 30/1001 Hz. Para un servicio one-seg se permite utilizar solamente

Tasa de cuadros (Hz)	Entrelazado o Progresivo
25	P
25	I
50	P
30000/1001	P
30000/1001	I
60000/1001	P

Tabla 3.3: Tasa de cuadros y sistema de escaneo para *codec* H.264 en full-seg.

Resolución de luminancia	Razón de aspecto
160 x 120	4:3
160 x 90	16:9
320 x 240	4:3
320 x 180	16:9
352 x 288	4:3

Tabla 3.4: Resoluciones y razón de aspecto para *codec* H.264 en one-seg.

el perfil *Baseline* con un nivel máximo de 1.3.

Codecs de audio compatibles

Se define la compatibilidad con el estándar de codificación de audio MPEG-4 AAC (norma ISO/IEC 14496-3 [23]). Similar a H.264, para AAC se definen los parámetros perfil y nivel que controlarán el conjunto de capacidades que serán utilizadas a la hora de codificar. Los perfiles y niveles compatibles con ISDB-Tb son:

- LC-AAC (*low complexity*), perfil básico del estándar AAC; niveles 2 y 4;
- HE-AAC (*high efficiency*), perfil avanzado de alta eficiencia; niveles 2 y 4;
- HE-AACv2 (versión 2), perfil HE combinado con la herramienta PS (*parametric stereo*)[3]; nivel 2.

Para una transmisión **full-seg** se recomiendan los siguientes números de canales de audio, y se permiten los siguientes perfiles y niveles:

- Números de canales recomendados: mono (1.0), dos canales (estéreo o 2.0), o multicanal (5.1).
- Perfiles y niveles permitidos: LC-AAC: nivel 2 para dos canales; LC-AAC: nivel 4 para multicanal; HE-AAC: nivel 2 para dos canales; HE-AAC: nivel 4 para multicanal.

En el caso de una transmisión **one-seg** se permiten los siguientes parámetros:

- Número máximo de canales codificados: 2 canales (estéreo o dos canales monoaurales)
- Perfiles y niveles permitidos: HE-AACv2: nivel 2.

3.4.2. Bitrates en ISDB-T

A la hora de generar *transport streams* para transmitir a través de ISDB-Tb, es importante respetar el *bitrate* útil que se permite según la norma. Desviaciones respecto a este generarán problemas a la hora de reproducir correctamente el contenido en recepción. Cabe destacar que tanto ISDB-T como ISDB-Tb se comportan exactamente igual en cuanto al cálculo del *bitrate* útil.

Para el cálculo del *bitrate* útil se deben tomar en cuenta los parámetros que se están utilizando en transmisión. Por cada capa jerárquica se pueden elegir sus respectivos parámetros, por lo cual el cálculo de *bitrate* útil es independiente para cada capa. También influye la cantidad de segmentos que se utilicen en cada capa.

El bitrate útil en un segmento para una capa se expresa con la ecuación 3.5:

$$r_{seg} = \frac{N_s.m_l.R_{RS}.R_{CC}}{T_s(1+CP)}. \tag{3.5}$$

Siendo N_s la cantidad de portadoras de datos útiles en el segmento, m_l[4] representa

[3]Herramienta que permite reconstruir una señal estéreo mediante el envío de una señal mono, en conjunto de una pequeña información extra. Esto posibilita reducir el *bitrate* respecto del envío de una señal estéreo completa.

[4]m_l vale 2 en QPSK, 4 en 16QAM y 6 en 64QAM.

la cantidad de bits por símbolo que dependerá del esquema de modulación, R_{RS} es la tasa de Reed Solomon, R_{CC} es la tasa de código convolucional, T_s es el tiempo de símbolo, y CP es el valor del prefijo cíclico expresado como fracción de T_s. Los valores N_s y T_s son dependientes del modo que se este usando en la transmisión (aunque su cociente $\frac{N_s}{T_s}$ es constante para todo modo). R_{RS} tiene un valor fijo de 188/204. CP es un parámetro que se define para el transmisor, y que toma el mismo valor para todas las capas de una misma transmisión. Por último, m_l y R_{CC} son parámetros que pueden variar según lo que se defina en cada capa. Cabe destacar que al utilizar más de un solo segmento en una capa, se debe multiplicar la expresión 3.5 por la cantidad de segmentos para obtener el *bitrate* correspondiente a cada capa. Con esto es posible por ejemplo, calcular el máximo *bitrate* posible que se puede lograr con esta norma. Para eso se seleccionan los siguientes parámetros: $m_l = 6$ (modulación 64QAM), $R_{CC} = \frac{7}{8}$, $CP = \frac{1}{32}$, y se utiliza modo 3 (aunque la selección de modo es invariante para el valor de *bitrate*, ya que como se dijo anteriormente, el cociente $\frac{N_s}{T_s}$ es el mismo para todo modo). Y se tiene que $R_{RS} = \frac{188}{204}$ y $\frac{N_s}{T_s} = \frac{384}{1008\ \mu s}$. Por lo que se obtiene que:

$$r_{seg} = \frac{N_s.m_l.R_{RS}.R_{CC}}{T_s(1+CP)} = \frac{384 \times 6 \times \frac{188}{204} \times \frac{7}{8}}{1008 \times 10^{-6} \times (1+\frac{1}{32})}\ \frac{b}{s} = 1,78728\ \frac{Mb}{s}. \tag{3.6}$$

Este resultado es para un solo segmento. Si se transmite utilizando los 13 segmentos con estos parámetros, se obtiene:

$$r = 13 \times 1,78728\ \frac{Mb}{s} = 23,2347\ \frac{Mb}{s}, \tag{3.7}$$

valor de *bitrate* máximo que se puede obtener en una transmisión ISDB-T. El documento ARIB STD-B31 [2] aporta una tabla, adjuntada como figura 3.6 en la presente documentación, en la que se da el resultado de aplicar la ecuación 3.5 a distintos conjuntos de parámetros. En la tabla, "*Carrier modulation*" define el parámetro m_l, "*Convolutional Code*" el parámetro R_{CC} y "*Guard interval ratio*" el parámetro CP.

El proyecto como tal, precisa del manejo de los conceptos de codificación para ISDB-Tb, tanto como del resto de las nociones que fueron explicadas en toda la extensión de este capítulo. La norma ISDB-T es la base de los distintos desarrollos que se propusieron e implementaron a lo largo del proyecto.

Carrier modulation	Convolutional code	Number of TSPs transmitted *1 (Mode 1/2/3)	Data rate (kbps) *2			
			Guard ratio: 1/4	Guard ratio: 1/8	Guard ratio: 1/16	Guard ratio: 1/32
DQPSK QPSK	1/2	12/24/48	280.85	312.06	330.42	340.43
	2/3	16/32/64	374.47	416.08	440.56	453.91
	3/4	18/36/72	421.28	468.09	495.63	510.65
	5/6	20/40/80	468.09	520.10	550.70	567.39
	7/8	21/42/84	491.50	546.11	578.23	595.76
16QAM	1/2	24/48/96	561.71	624.13	660.84	680.87
	2/3	32/64/128	748.95	832.17	881.12	907.82
	3/4	36/72/144	842.57	936.19	991.26	1021.30
	5/6	40/80/160	936.19	1040.21	1101.40	1134.78
	7/8	42/84/168	983.00	1092.22	1156.47	1191.52
64QAM	1/2	36/72/144	842.57	936.19	991.26	1021.30
	2/3	48/96/192	1123.43	1248.26	1321.68	1361.74
	3/4	54/108/216	1263.86	1404.29	1486.90	1531.95
	5/6	60/120/240	1404.29	1560.32	1652.11	1702.17
	7/8	63/126/252	1474.50	1638.34	1734.71	1787.28

*1: Represents the number of TSPs transmitted per frame

*2: Represents the data rate (bits) per segment for transmission parameters
Data rate (bits): TSPs transmitted × 188 (bytes/TSP) × 8 (bits/byte) × 1/frame length

Figura 3.6: Tabla de cálculo de *bitrates* [2].

Capítulo 4

Desarrollo y pruebas de software transmisor ISDB-T

La continuación en el desarrollo del *software* transmisor ISDB-T es uno de los objetivos del proyecto. Como tal, se procederá en este capítulo a describir las distintas implementaciones que se le hicieron al *software*, así como las pruebas que se realizaron para validar el correcto funcionamiento.

4.1. Desarrollo de transmisor ISDB-T full-seg

La figura 4.1 ilustra un transmisor ISDB-T implementado en GNU Radio. Este transmisor es el resultado de un conjunto de esfuerzos de desarrollo de la Facultad de Ingeniería de la UdelaR [5]. En él, se pueden diferenciar las etapas de procesamiento llevadas a cabo por un modulador de ISDB-T y expuestas en la sección 3.3.2. En este transmisor se pueden introducir hasta tres *transport streams* independientes para generar una transmisión de hasta tres capas jerárquicas. El repositorio original de este transmisor ISDB-T (y en el que se basó este proyecto) se puede encontrar en [24].

A este transmisor, se decidió como parte del proyecto añadirle la opción de poder inyectar un BTS en lugar de TSs independientes. Para esto, se procedió a extraer parte de un transmisor ISDB-T desarrollado en un proyecto de fin de carrera anterior [6]. Este transmisor trabaja con un BTS a la entrada del mismo, y contiene los bloques necesarios para poder dividir el BTS de entrada en varios TSs. El bloque "*Hierarchical Divisor*", explicado ya en la sección 3.3.2, subdivide los flujos de las distintas capas jerárquicas que se encuentran en el BTS hacia flujos independientes. El bloque "*TSP Resize*" toma cada TSP que conforma a la BTS y elimina los últimos 16 bytes de información. Esto permite trabajar con TSPs cuyo tamaño de 188 bytes es el esperado por el resto del flujo del transmisor. Se procedió a tomar estos bloques, y añadirlos al proyecto del transmisor de la figura 4.1, obteniendo lo que se observa en la figura 4.2. Posteriormente, se procedió a verificar el correcto funcionamiento de este transmisor, corregirlo en escenarios donde el conjunto de parámetros no funcionase y realizar pruebas de funcionamiento. Se

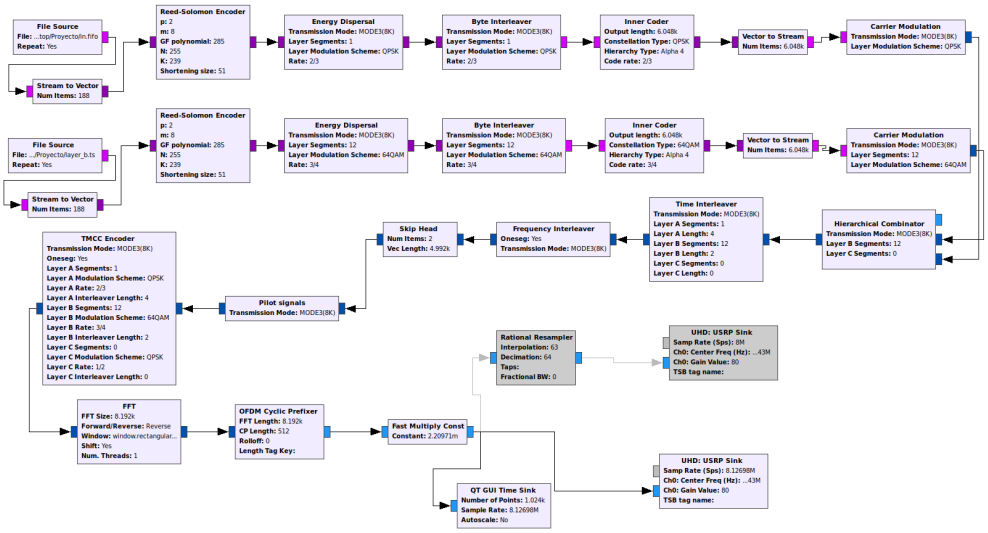


Figura 4.1: Transmisor de trece segmentos que parte de TSs independientes, implementado en GNU Radio.

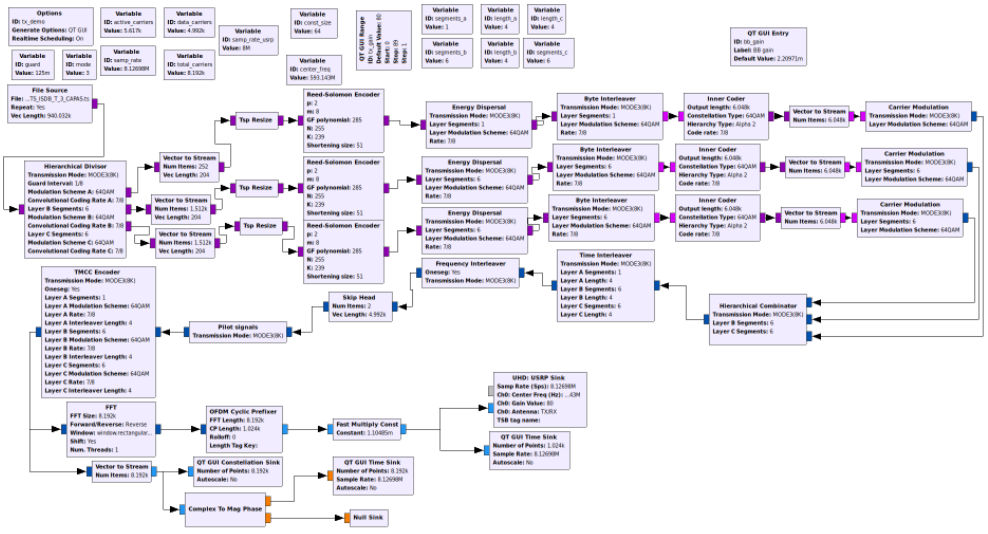


Figura 4.2: Transmisor de trece segmentos que parte de un BTS, implementado en GNU Radio.

detectaron principalmente dos problemas:

- La imposibilidad de transmitir BTSs que contengan menos de tres capas jerárquicas (el máximo de capas jerárquicas que se pueden utilizar en una transmisión ISDB-T). Es común en transmisiones de las operadoras de televisión por aire locales usar tan solo dos capas jerárquicas.
- La imposibilidad de transmitir para algunos valores de tasa de código convolucional.

Resolver estos problemas requirió múltiples ensayos, estudiando las secciones del transmisor de manera aislada. Se concluyó que los problemas se centraban en el ya mencionado bloque "*Hierarchical Divisor*" que se le integró al transmisor. Solucionar estos problemas consistió en modificar partes del código fuente del bloque. Se agregaron condiciones que permiten extraer menos de tres capas jerárquicas del bloque, posibilitando trabajar con una mayor variedad de BTSs. También se corrigieron algunos errores de cuentas en el código, que generaban que el transmisor no funcione correctamente para algunos valores de códigos convolucionales. Hecho esto, se pudo comprobar que el transmisor ahora funciona con todas las configuraciones disponibles.

4.2. Transmisor one-seg

4.2.1. Introducción

La norma ISDB-T contempla la posibilidad de realizar lo que se denomina como recepción parcial. Esto consiste en recibir y decodificar el segmento central de la transmisión, independientemente de los demás 12 segmentos. Para que esto sea posible, la codificación de canal y entrelazado en frecuencia están completamente autocontenidas dentro del segmento. La recepción parcial fue ideada para ser utilizada en dispositivos móviles, cuya baja capacidad de recepción y procesamiento se ve beneficiada por los bajos anchos de banda y *bitrates*. Además, la distancia entre el usuario y el móvil conjuntamente con el tamaño de la pantalla logran una experiencia razonable para este servicio de bajo *bitrate*. En lo que compete a este proyecto, se entiende que es razonable trabajar con una funcionalidad similar a esta, pero que se aplique a la transmisión. Es decir, desarrollar *software* transmisor ISDB-T que genere una transmisión parcial. Esto beneficiaría al *hardware* transmisor de bajo costo, cuya potencia de transmisión es limitada. La posibilidad de poder transmitir un menor ancho de banda que en una transmisión full-seg permitiría aprovechar en mejor manera esa potencia. Otro beneficio de la transmisión one-seg es la considerable disminución en el procesamiento del equipo transmisor, dado que este solo procesa los datos correspondientes a un segmento. Cabe destacar que a diferencia del concepto de recepción parcial, la transmisión parcial no está contemplada en la norma ISDB-T.

En una transmisión one-seg, solo es necesario enviar las portadoras correspondientes a un segmento. La tabla 4.1 indica las portadoras por segmento para cada

Modo	Portadoras Activas	Portadoras de datos
Modo 1	108	96
Modo 2	216	192
Modo 3	432	384

Tabla 4.1: Cantidad de portadoras por segmento para cada modo.

modo. De esto se desprende que las portadoras necesarias para la IFFT estarán dadas por:

- 128 portadoras para modo 1.
- 256 portadoras para modo 2.
- 512 portadoras para modo 3.

Como fue explicado en la sección 3.3.2, no todas las portadoras son utilizadas para enviar datos, sino que a una fracción se le asignan valores nulos debido al *zero-padding*. La tabla 4.2 indica la cantidad de portadoras a las que se le asignan valores nulos en transmisión one-seg.

Modo	Nz	A la izquierda	A la derecha
Modo 1	20	10	10
Modo 2	40	20	20
Modo 3	80	40	40

Tabla 4.2: Ceros necesarios para el zero-padding de la IFFT.

Teniendo en cuenta que el tiempo de símbolo en modo tres corresponde a 1008μs, se obtiene que la frecuencia de muestreo del transmisor one-seg corresponde a[1]

$$\frac{512}{1008\mu s} \simeq 507{,}94 \text{ kHz}, \tag{4.1}$$

lo cual implica bajar considerablemente la frecuencia a la que las portadoras se deben enviar de 8,127 MHz para full-seg a 507,94 kHz en one-seg. Por otro lado, también disminuye el uso espectral, donde el ancho de banda de la señal pasará a ocupar:

$$\frac{432}{1008\mu s} \simeq 428{,}6 \text{ kHz}. \tag{4.2}$$

Por lo tanto se pasa de ocupar 5,6 MHz a tan solo 428,6 kHz de ancho de banda, reduciéndose considerablemente el ancho de banda necesario para la transmisión.

El transmisor one-seg está basado en el transmisor full-seg que se introdujo en la sección 4.1. En lo que resta de este capítulo se expondrán las características más importantes de su implementación.

[1]Observar que los siguientes resultados son independientes del modo.

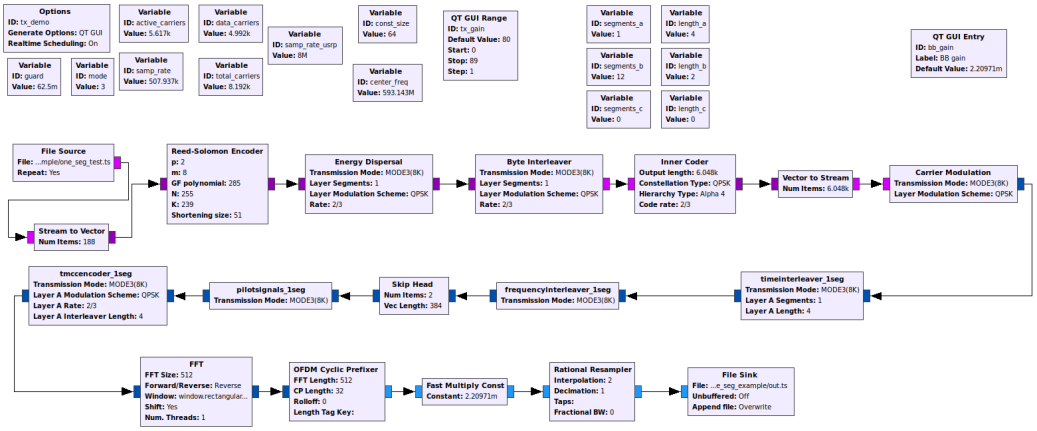


Figura 4.3: Transmisor one-seg implementado en GNU Radio.

4.2.2. Implementación de transmisor one-seg en GNU Radio

Se puede visualizar en la figura 4.3 el transmisor one-seg. En este transmisor se reemplazan los bloques:

- *Time Interleaver* por *timeinterleaver_1seg*
- *Frequency Interleaver* por *frequencyinterleaver_1seg*
- *Pilot Signals* por *pilotsignals_1seg*
- *TMCC Encoder* por *tmccencoder_1seg*

Estos nuevos bloques procesan únicamente las portadoras correspondientes a one-seg. Hasta la etapa de combinación jerárquica, la principal diferencia con las etapas de procesamiento introducidas en 3.3.2 es que se procesará únicamente lo correspondiente al TS asociado a one-seg. Se puede observar esta diferencia comparando los *flowgraph* 4.1 y 4.3. Al contar únicamente con un segmento de la capa A, el combinador jerárquico es innecesario en este transmisor. Ya que su propósito es unir los flujos de las tres capas en uno solo ordenado de manera ascendente (explicado en 3.3.2).

Timeinterleaver_1seg:

Durante la etapa de entrelazado temporal, se toma de entrada las portadoras de datos correspondientes a un segmento,

$$\text{portadoras} = 96 \times 2^{\text{modo}}.$$

El bloque distribuye los símbolos complejos entre distintos símbolos OFDM. Esta distribución dependerá de la profundidad del entrelazado y del número de la portadora i. Es implementado almacenando los complejos en colas en las que se empujan los símbolos que van arribando. La figura 4.4 ilustra el procesamiento

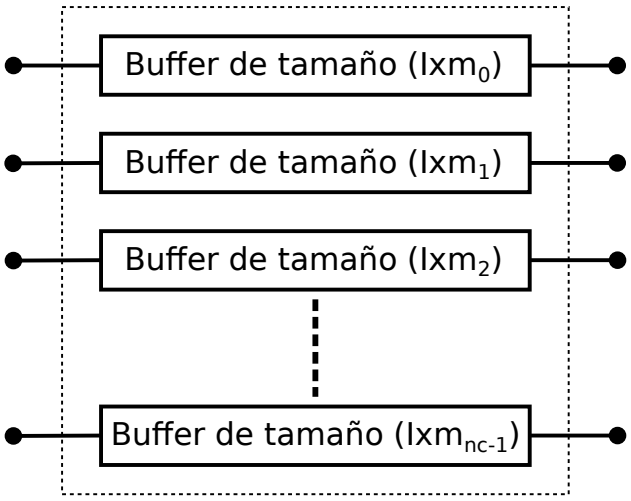


Figura 4.4: Entrelazado temporal en transmisor de un segmento [18].

empleado. El tamaño de estas colas es $I \times m_i$ con I la profundidad del entrelazado y $m_i = (i \times 5) \bmod 96$. Se obtiene como resultado las portadoras de datos entrelazadas temporalmente.

Frequencyinterleaver_1seg:

Frequencyinterleaver_1seg lleva a cabo el entrelazado frecuencial intrasegmento. La norma especifica que para el entrelazado de one-seg, hay que hacer ambas etapas de rotación y aleatorización introducidas en 3.3.2. Este transmisor implementa la recomendación extraída de [18], en donde se plantea de manera teórica la posibilidad de prescindir de la etapa de rotación (ver 3.4) para one-seg. Esta recomendación fue verificada en las posteriores pruebas de funcionamiento, en donde el transmisor funcionó correctamente. La aleatorización de símbolos es lograda reordenando las portadoras según look-up tables establecidos en la norma para cada modo de transmisión.

Pilotsignals_1seg:

En esta etapa, se reordenan las portadoras de datos en un flujo que ahora tiene posicionadas correctamente las portadoras dentro del segmento. Este bloque tiene como entrada las $96 \times 2^{(\text{modo}-1)}$ portadoras de datos y como salida $108 \times 2^{(\text{modo}-1)}$ portadoras en donde se encuentran ordenadas las posiciones de:

- Portadoras de datos
- *AC Pilots*
- *Scattered Pilots*
- *TMCC carriers*

todos asignados según lo especifica la norma para el segmento one-seg según el modo de transmisión.

En transmisiones full-seg la cantidad de portadoras utilizadas para cada modo de transmisión es $13 \times 108 \times 2^{(\text{modo}-1)}$ dando una totalidad de 1404, 2808 y 5616 portadoras para los modos 1, 2 y 3 respectivamente. Dentro de este escenario es necesario ubicar las posiciones de las portadoras para el segmento one-seg. La tabla 4.3 expone las posiciones de las portadoras TMCC y pilotos AC dentro del total de portadoras activas en full-seg.

Modo 3	
Posiciones de portadoras TMCC	2693, 2723, 2878, 2941
Posiciones de pilotos AC	2599, 2681, 2798, 2801, 2818, 2836, 2969, 2999
Modo 2	
Posiciones de portadoras TMCC	1319, 1474
Posiciones de pilotos AC	1394, 1397, 1414, 1432
Modo 1	
Posiciones de portadoras TMCC	697
Posiciones de pilotos AC	683, 727

Tabla 4.3: Posiciones de portadoras TMCC y pilotos AC del segmento central relativo a la cantidad de portadoras activas en full-seg para cada modo de transmisión.

Para ubicar las portadoras TMCC y pilotos AC, el bloque *pilotsignals_1seg* realiza lo siguiente: toma la posición de cada portadora desde la tabla 4.3, y le resta la posición de la primer portadora del segmento one-seg. Con esto se obtiene la posición relativa de cada portadora respecto al comienzo del segmento one-seg. Por ejemplo, en modo 2 se tiene 216 portadoras activas por segmento, esto significa que la primera portadora correspondiente al segmento one-seg relativa a full-seg se ubicaría en la posición $216 \times 6 = 1296$. De esta manera las portadoras TMCC estarán ubicadas en las posiciones 23 $(1319 - 1296)$ y 178 $(1474 - 1296)$, dentro de las 216 activas.

Las *Scattered Pilots* se posicionan cada 12 portadoras. La posición de cada una de las *Scattered Pilots* varía en cada símbolo OFDM. Estas rotan según la expresión $3\times(\text{n mod } 4)$, donde n es el número de símbolo OFDM dentro del cuadro. Se adaptaron las posiciones correspondientes para el segmento one-seg.

Las portadoras de datos son asignados a partir de las posiciones que quedaron libres luego de las etapas anteriores. Durante este bloque, las posiciones de portadoras TMCC y AC quedan asignadas pero vacías para luego ser llenadas en etapas posteriores. Este transmisor no incluye contenido dentro de las portadoras AC debido a su uso opcional. Esta característica queda como una posible implementación futura. También, al estar trabajando en un segmento, no se incorpora el piloto continuo introducido en la sección 3.3.2.

Tmccencoder_1seg:

En la implementación de *tmccencoder_1seg* se toman las $108 \times 2^{(\text{modo}-1)}$ portadoras de *pilotsignals_1seg* y se devuelve $128 \times 2^{(\text{modo}-1)}$ portadoras. En su salida se incluye la información de todas las portadoras entrantes, ahora incorporando la información relativa a la TMCC. Es también en este bloque que se implementa el *zero-padding*, el cuál, dependiendo del modo, adiciona según la tabla 4.2 la cantidad de ceros necesarios en portadoras laterales.

La información aportada por la TMCC se describe en la sección 3.3.2. *tmccencoder_1seg* traduce y asigna a las portadoras TMCC correspondientes la información relativa de transmisión. Cada parámetro de transmisión tiene su correspondiente palabra de bits, las cuales en conjunto constituyen a la portadora TMCC.

4.3. Transmisión de *transport streams* bajo la norma ISDB-T con sus distintas variantes de parámetros

4.3.1. Descripción de las pruebas

Con el fin de testear el transmisor ISDB-T full-seg explicado en la sección 4.1 (transmisor de la figura 4.1, al que se le inyectan TSs independientes), se realizaron distintas pruebas de transmisión con distintos equipos y parámetros, para poder constatar su correcto funcionamiento en todos los casos.

La frecuencia elegida para las pruebas fue 593,143 MHz correspondiente al canal 34 de televisión digital, el cual se encuentra libre. Las pruebas efectuadas no refieren al alcance que tiene el transmisor, dado que estas fueron realizadas por la tesis de grado "Implementación de un transmisor de ISDB-T abierto bajo el paradigma de radio definida por *software*" [6]. El objetivo de estas pruebas es verificar el impacto de los distintos parámetros de transmisión especificados en la norma de televisión digital y su correcta implementación en *software*.

Hardware utilizado para transmitir:

- USRP B200 [13].
- Antena Ikusi modelo Flash HD NANO [25].
- *Laptop* con procesador Intel Core i7-4712MQ y 16GB de memoria RAM.

Hardware utilizado para recibir:

- USRP B100 [26].
- Antena LP09650 de Ettus.
- Computadora de escritorio, con procesador Intel Core i7-8700 y 24GB de memoria RAM.

La elección de las antenas se debe a la frecuencia utilizada, ambas están diseñadas para trabajar por encima de los 400 MHz. Las pruebas se realizaron con todas

las combinaciones de parámetros posibles, aunque en esta sección solo se mostrarán los casos que creemos más representativos. Entre estos, se tomaron casos de transmisiones con el mayor *bitrate* útil posible en ISDB-T. Esto implica utilizar el mínimo intervalo de guarda permitido, el máximo código convolucional y modulación 64QAM. Esta transmisión es la transmisión menos robusta posible. Por otro lado se tomaron transmisiones con el menor *bitrate* útil posible en ISDB-T. Por lo que se eligieron los siguientes parámetros: máximo intervalo de guarda, menor código convolucional y modulación QPSK. Esta transmisión es la transmisión más robusta posible. Por último se eligieron parámetros intermedios, para representar los casos faltantes. Los cálculos de *bitrate* según los parámetros se pueden realizar tanto utilizando la ecuación 3.5 o la figura 3.6.

Se variaron los modos de transmisión de ISDB-T, lo que implica que tanto la cantidad de portadoras totales, como las portadoras activas, variaron en cada transmisión. A su vez, varía el tiempo de símbolo OFDM, de manera que el ancho de banda y el *bitrate* permanecen constantes, como fue explicado en el capítulo 3.

4.3.2. Resultados

Para la prueba N^o1 se transmitió capa A, sin capa B y sin capa C. Se eligieron los parámetros de la tabla 4.4. Los parámetros fueron seleccionados de esta manera ya que dan como resultado el mayor *bitrate* útil posible en una transmisión ISDB-T, cuyo valor es 23,235 Mbps.

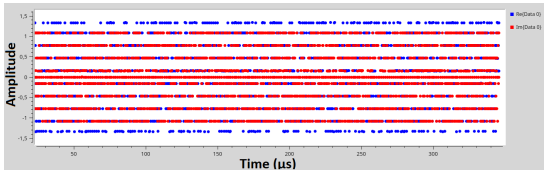
Para la prueba N^o2 se transmitió capa A y capa B, sin capa C. Se eligieron los parámetros indicados en la tabla 4.5. Los parámetros seleccionados dan como resultado el menor *bitrate* útil que se puede lograr en ISDB-T. En capa A el *bitrate* correspondiente es de 280,85 kbps, en la capa B es de 3,3702 Mbps, siendo la suma entre ellas (y el *bitrate* total de la transmisión) un total de 3,6510 Mbps.

Para la prueba N^o3 se transmitió capa A y capa B, sin capa C. Se eligieron los parámetros indicados en la tabla 4.6. Los mismos representan los parámetros que determinan un *bitrate* útil intermedio entre el máximo y el mínimo posibles.

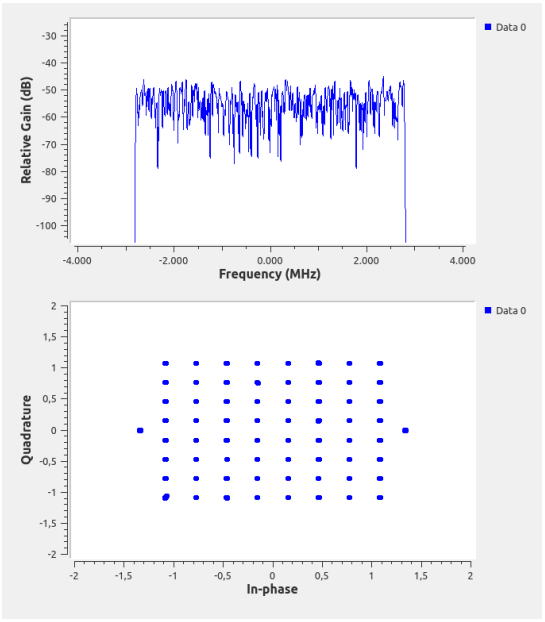
En las figuras 4.5, 4.6 y 4.7 se pueden ver las respectivas constelaciones y complejos recibidos. En la prueba número N^o1, se puede apreciar la modulación DBPSK de las portadoras piloto a la derecha e izquierda de la modulación 64QAM. En la prueba N^o2, se utiliza modulación QPSK para todas las capas y al igual que para la prueba anterior, aparecen las portadoras DBPSK a ambos lados. Por último en la prueba N^o3, se utiliza la modulación 64QAM para una capa y QPSK para la otra, por lo que aparecen superpuestas en la constelación. Además, a cada lado en la constelación vuelven a aparecer las portadoras piloto moduladas en

Capa	Modo	Intervalo de guarda	Código convolucional	Número de segmentos	Entrelazamiento temporal	Modulación
A	2	1/32	7/8	13	4	64QAM

Tabla 4.4: Parámetros utilizados para la prueba N^o1.



(a) Amplitud de los complejos recibidos.



(b) Amplitud de espectro recibido y constelación.

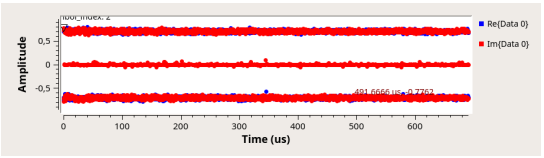
Figura 4.5: Prueba $N°1$ realizada en *software*.

Capa	Modo	Intervalo de guarda	Código convolucional	Número de segmentos	Entrelazamiento temporal	Modulación
A	1	1/4	1/2	1	4	QPSK
B	1	1/4	1/2	12	2	QPSK

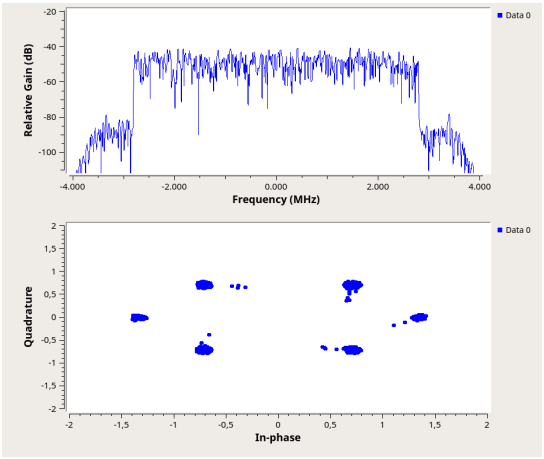
Tabla 4.5: Parámetros utilizados para la prueba $N°2$.

Capa	Modo	Intervalo de guarda	Código convolucional	Número de segmentos	Entrelazamiento temporal	Modulación
A	3	1/4	2/3	1	4	QPSK
B	3	1/4	3/4	12	2	64QAM

Tabla 4.6: Parámetros utilizados para la prueba $N°3$.

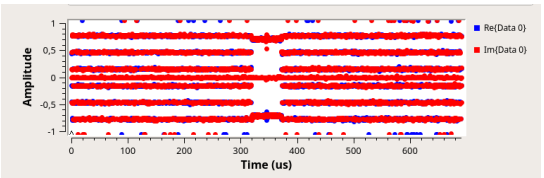


(a) Amplitud de los complejos recibidos.

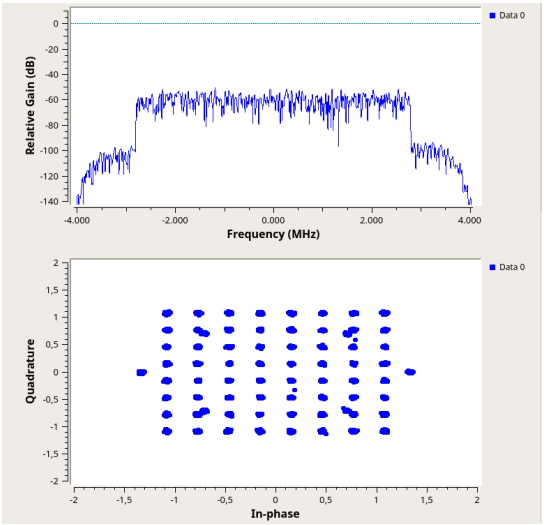


(b) Amplitud de espectro recibido y constelación.

Figura 4.6: Prueba $N°2$.



(a) Amplitud de los complejos recibidos.



(b) Amplitud de espectro recibido y constelación.

Figura 4.7: Prueba $N°3$.

DBPSK.

En las imágenes 4.5a, 4.6a y 4.7a se muestran las variaciones de amplitud de la parte real e imaginaria de cada complejo recibido. Es otra manera de visualizar la constelación recibida, pero nos da la información de la posición de los complejos en el símbolo OFDM. Por ejemplo, en el centro de la figura 4.7a se pueden observar los complejos recibidos modulados en QPSK correspondientes a la capa A. Se observan 2 niveles dado que la amplitud de la parte real e imaginaria varía entre -A y A, tomando como los casos posibles los complejos $A + jA$, $A - jA$, $-A + jA$ y $-A - jA$. En las puntas de la figura, se ven los complejos correspondientes a la capa B, modulada en 64QAM. Por lo que aparecen 8 niveles, dado que son los valores que varían la parte real e imaginaria de los complejos en dicha modulación.

Se notó que utilizando un intervalo de guarda pequeño, como lo es 1/32, se perdía el sincronismo entre el transmisor y el receptor luego de unos segundos de transmisión. Para verificar que este problema se daba al adicionar el canal en el sistema, y no fuera propio del transmisor y sus parámetros, se replicó la prueba en *software* (transmisión y recepción en *software*, sin participación del canal, ni del *hardware* de transmisión). En esta se relevó un funcionamiento correcto, sin problemas.

Es posible afirmar gracias a estas pruebas, que todos los parámetros disponibles en el transmisor ISDB-T full-seg funcionan de manera correcta. En este capítulo a su vez se expuso la explicación teórica e implementación del transmisor one-seg. El mismo es una parte fundamental del proyecto para la implementación del transmisor en *hardware* alternativo.

Capítulo 5

Generación de *transport streams* para transmisión

Tal como fue explicado en el capítulo introductorio, se propuso como objetivo del proyecto lograr generar nuestros propios *transport streams*, para que luego estos sean transmitidos bajo la norma ISDB-Tb. Estos serán introducidos directamente en el transmisor de la figura 4.1, sin pasar por el proceso de crear un BTS a partir de ellos. Es de interés generar *transport streams* tanto a partir de videos almacenados, como de videos generados en tiempo real (sea por una cámara web, o por la grabación de la pantalla de una computadora como posibles ejemplos). Es importante aclarar que para que estos sean compatibles con la norma es necesario cumplir con las directivas de codificación y *bitrate* expresadas en la sección 3.4. La herramienta *ffmpeg* [27] de código abierto, utilizada en una terminal de distribución Linux, permite mediante un único comando codificar correctamente tanto videos almacenados, como videos generados en tiempo real.

5.1. Uso de *ffmpeg* para codificación de videos almacenados

Se hará uso de un ejemplo de comando de *ffmpeg* para ilustrar su funcionamiento, y explicar cada una de las opciones del comando que se involucran en la generación de un *transport stream* compatible con ISDB-Tb. Cabe destacar que este comando en particular generará un *transport stream* compatible con una transmisión de 13 segmentos con modulación QPSK, intervalo de guarda $\frac{1}{16}$ y tasa de código convolucional $\frac{2}{3}$. Además, por la resolución utilizada el contenido será del tipo SD (*Standard Definition*).

Comando:

```
ffmpeg -i file.mp4 -c:v libx264 -b:v 2M -r ntsc -s 720x480 -pix_fmt
yuv420p -profile:v main -level:v 3 -c:a aac -b:a 64k -profile:a aac_low
-f mpegts -mpegts_service_id 0x70e0 -metadata service_name=‘‘ISDBT Channel’’
-muxrate 5727280 ISDBTVideo.ts
```

- `-i` : Opción con la que se define la ruta del archivo a convertir.
- `-c:v` : *Codec* de video a utilizar en la codificación. `libx264` refiere a la implementación libre del *encoder* de H.264.
- `-b:v` : *Bitrate* de video promedio al que se va a codificar el archivo. La suma de este con el valor de -b:a deben ser menores al valor de -muxrate.
- `-r`: Valor que define los cuadros por segundo del video de salida. En el ejemplo se utiliza el valor `ntsc` que equivale a 30/1001 Hz.
- `-s`: Resolución del video de salida.
- `-pix_fmt`: Formato de píxeles del archivo de salida.
- `-profile:v`: Perfil del *codec* de video que se va a utilizar en la codificación.
- `-level:v`: Nivel del *codec* de video que se va a utilizar en la codificación.
- `-c:a`: *Codec* de audio a utilizar en la codificación.
- `-b:a`: *Bitrate* de audio.
- `-profile:a`: Perfil de AAC a utilizar. En este caso se seleccionó el perfil LC-AAC.
- `-f`: Formato de salida del video, y por consecuente el multiplexador a utilizar. Para ISDB-T se utiliza el formato MPEG-2 TS (*MPEG-2 Transport Stream*, definida en la norma ISO/IEC 13818-1), implementada con `mpegts` como opción a colocar en el comando.
- `-mpegts_service_id`: Número de servicio.
- `-metadata service_name=`: Define el nombre que va a tener el servicio. Es el nombre del canal que se visualiza en el receptor.
- `-muxrate`: *Bitrate* total (y constante) que va a devolver el multiplexor (y por lo tanto el *bitrate* del archivo de salida). Aquí se debe colocar en bits el *bitrate* necesario según se definió en la sección 3.4.2. Cabe destacar que la suma de los *bitrates* de audio y video definidos en opciones anteriores, debe ser menor al valor del *muxrate*. Es aconsejable dejar un rango entre la suma de *bitrates* de video y audio y el valor de *muxrate*. Esto permite que el *encoder* de video, al ser por lo general de *bitrate* variable, pueda superar por momentos su valor medio definido, sin generar problemas en el *bitrate* total de salida del archivo.

Luego de generado el *transport stream* con el *bitrate* correcto y los parámetros que correspondan, este puede ser introducido en el flujo de GNU Radio para que sea enviado por el transmisor, logrando realizar una transmisión ISDB-T en base a un video almacenado.

5.2. Uso de *ffmpeg* para codificación de videos generados en tiempo real

La sección anterior tenía como objetivo explicar la generación de *transport streams* compatibles con ISDB-Tb a partir de videos ya almacenados. Para esta sección se detallará la generación en tiempo real de *transport streams* para que estos sean transmitidos en vivo según la norma. En particular se va a utilizar primero el video generado por una cámara web, y luego por la grabación de pantalla de una computadora, para ser recodificados según los requerimientos de la norma y luego transmitidos. Como primer paso, es recomendable la creación de un archivo del tipo FIFO (*First In, First Out*), también conocido como "tubería nombrada". Este tipo de archivo se utiliza para la comunicación entre procesos. En este caso, los datos generados por la cámara web o grabación de pantalla son codificados y trasladados a través de este archivo hacia la entrada del transmisor en GNU Radio. El comando `mkfifo` permite la creación de este tipo de archivo en Linux.

Al igual que con el comando utilizado para la generación de *transport streams* a partir de videos almacenados, se va a demostrar un comando de ejemplo, y se marcarán las diferencias que surgen con el anteriormente explicado. El siguiente comando se utiliza para la generación de un *transport stream* a partir de una cámara web, el cual será compatible con una transmisión de 13 segmentos en el que se utilicen los mismos parámetros que en el ejemplo de la sección 5.1. Por la resolución utilizada el contenido será del tipo HD (*High Definition*).

Comando:

```
ffmpeg -f alsa -i pulse -f video4linux2 -i /dev/video0 -s 1280x720
-c:v libx264 -b:v 2M -profile:v main -level:v 3 -r ntsc -s 1280x720 -pix_fmt
yuv420p -c:a aac -b:a 64k -profile:a aac_low -f mpegts -mpegts_service_id
0x70e0 -metadata service_name=‘‘Example’’ -muxrate 5727280 in.fifo
```

- `-f alsa`: Implica la utilización del *framework* "Advanced Linux Sound Architecture" como formato de entrada del audio. Este comando permite utilizar dispositivos que estén asociados a ALSA para recibir datos de estos (en este caso, el dispositivo que se quiere utilizar es el micrófono de la computadora)

- `-i pulse`: Se utiliza el servidor de sonido PulseAudio como entrada para capturar el audio del micrófono.

- `-f video4linux2`: Video4Linux2 es una colección de drivers y API que permite la captura de video en Linux. Se define este como el formato de entrada.

- `-i /dev/video0`: Ruta en la cual se encuentra el dispositivo que se va a utilizar como fuente de video. En el ejemplo, se utiliza la ruta de una cámara web interna de una computadora portátil.

El próximo comando permite generar en tiempo real un *transport stream* a partir del video generado por la grabación de pantalla de una computadora. El

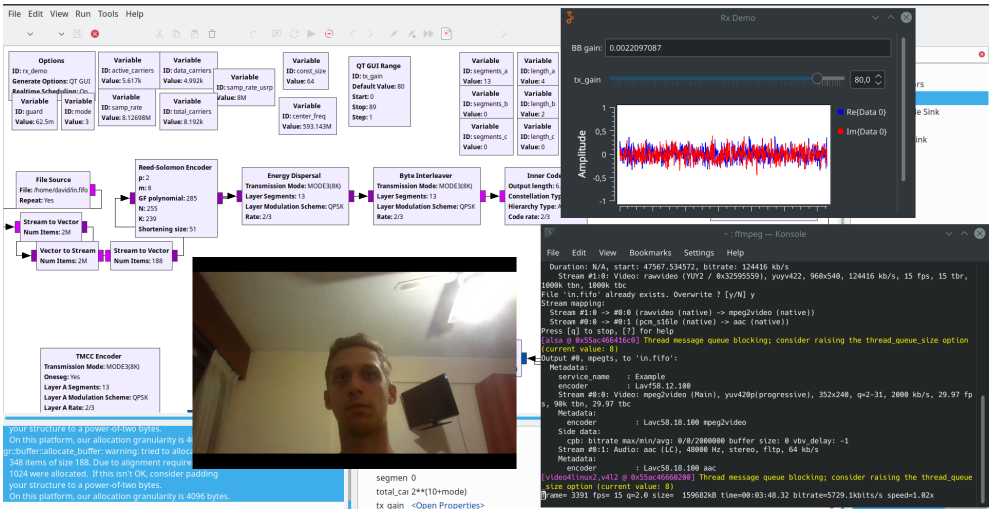


Figura 5.1: Transmisión ISDB-T en tiempo real tomando imagen y sonido desde cámara web.

transport stream será compatible con una transmisión en las mismas condiciones que el ejemplo anterior.

Comando:

`ffmpeg -s 1024x576 -r ntsc -f x11grab -i :0.0+100,200 -f alsa -i default -c:v libx264 -b:v 2M -profile:v main -level:v 3 -r ntsc -s 1280x720 -pix_fmt yuv420p -c:a aac -b:a 64k -profile:a aac_low -f mpegts -mpegts_service_id 0x70e0 -metadata service_name='Ingenieria DeMuestra' -muxrate 5727280 in.fifo`

- `-s 1024x576`: Define el tamaño de la sección de pantalla que se quiere grabar.
- `-f x11grab`: Formato que se utiliza para la grabación de pantalla en un sistema Linux.
- `-i :0.0+100,200`: Fija la coordenada desde la cual se tomará la sección de pantalla a grabar. Esta se define desde la punta superior izquierda de la pantalla. En este caso, la coordenada de inicio es 100 píxeles hacia la derecha y 200 píxeles hacia abajo desde la punta superior izquierda.

Con estos métodos, fue posible enviar transmisiones en vivo tanto de cámara web (tal como se puede ver en la figura 5.1), como de la vista de escritorio de una computadora personal, con buena estabilidad y calidad en la transmisión utilizando un transmisor USRP Ettus B200. La transmisión pudo ser recibida tanto por receptores SDR, como por sintonizadores ISDB-Tb comerciales.

Capítulo 6

Adaptador USB-VGA como transmisor de bajo costo

6.1. Introducción

Cuando en un equipo electrónico se producen cambios de corrientes y/o voltajes, se generan ondas electromagnéticas, ya sea intencionalmente o no. Las emisiones de dichas ondas pueden ser captadas y utilizadas para la transmisión de información.

La transmisión de video bajo la norma ISDB-T a través de un adaptador USB-VGA, está basada en el proyecto `Osmo-fl2k` [7]. El cual, aprovechándose de la emisión electromagnética del adaptador, logra transmitir distintos tipos de señales dentro de la señal VGA.

Existen varios proyectos en los cuales se utiliza el adaptador USB-VGA como transmisor. A modo de ejemplo se encuentra el proyecto fabricado por Fabrice Bellard en 2005, el cual utilizó una ATI Radeon 9200SE para transmitir señales de video DVB-T, PAL y NTSC [28]. Otro proyecto que utiliza la emisión de las ondas electromagnéticas de los cables de un conector VGA es `TEMPEST`. El mismo fue implementado en C y Java por Martin Marinov [29] y posteriormente Federico La Rocca realizó una re-implementación del mismo en GNU Radio [30]. El proyecto consiste en el espionaje de monitores a través de las emisiones electromagnéticas de los cables VGA. Con `TEMPEST` y un equipo SDR con sus respectivas antenas, es posible sintonizar la emisión de los armónicos de la señal que se transmite a través de los cables y reproducirla en el monitor que está espiando, como se puede observar en la figura 6.1. Este proyecto utiliza el mismo principio que `Osmo-fl2k`, pero con finalidades totalmente diferentes.

El proyecto `Osmo-fl2k` busca minimizar los costos en *hardware* en la transmisión de datos. Los SDR con capacidad de transmisión también se han vuelto más baratos, pero aún son caros. Este proyecto permite utilizar adaptadores USB 3.0 a VGA, basados en el chip Fresco Logic FL2000, observándose el adaptador y su chip en las figuras 6.2 y 6.3 respectivamente. La utilización de estos adaptadores se debe a que es posible operarlos de manera que la sincronización horizontal y vertical

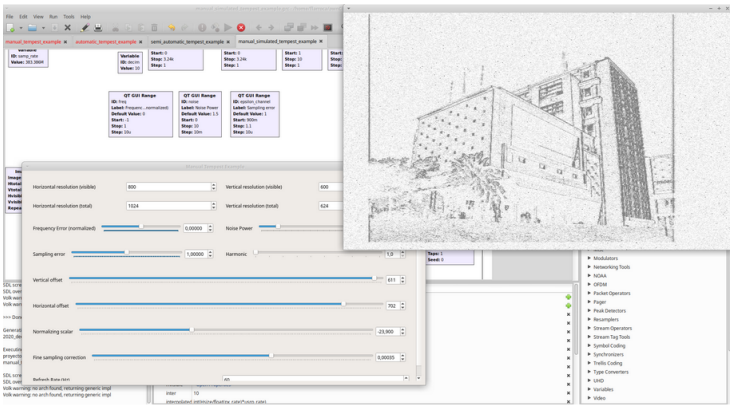


Figura 6.1: Espionaje de monitor utilizando implementación de TEMPEST [30].

Figura 6.2: Adaptador USB 3.0 a VGA basados en el chip Fresco Logic FL2000.

esté desactivada, pudiendo generar un flujo continuo de muestras. Esto no ocurre con otros adaptadores, dado que no es posible desactivar dicha sincronización, lo que produce cortes en la señal.

Figura 6.3: Chip Fresco Logic FL2000.

6.2. Funcionamiento

Se explicará en primera instancia cómo el adaptador USB-VGA transmite los distintos píxeles a través de sus cables. Este utiliza la modulación por amplitud de pulso (PAM, sigla proveniente del inglés *Pulse Amplitude Modulation*) para enviar dichos píxeles. El proyecto osmo-fl2k transmite la señal a través del pin rojo, correspondiente al pin 1 del adaptador USB-VGA. La explicación que se dará en este capítulo acerca de la transmisión de los píxeles, es válida para la transmisión a través de cada pin RGB (*red, green, blue*) del adaptador por separado, dado que los conversores digital-analógico de cada color trabajan de manera independiente.

La señal a transmitir por el adaptador USB-VGA es conformada, idealmente, por un pulso que vale $p(t) = 1$ cuando $0 \leq t \leq T$ y 0 en otro caso, siendo T el inverso de la frecuencia de muestreo del adaptador USB-VGA, a la cual llamaremos f_{out}. Sea $X(n)$ el pixel n-ésimo de la señal que se desea enviar y $x_T(t)$ la señal que sale del adaptador USB-VGA. Podemos escribir $x_T(t)$ como:

$$x_T(t) = \sum_{n=-\infty}^{\infty} X(n)p(t - nT). \tag{6.1}$$

Tomando como hipótesis que la señal $x_T(t)$ es estacionaria en sentido amplio, se tiene que la densidad espectral de potencia es:

$$S_{x_T}(f) = \frac{|P(f)|^2}{T} \sum_{k=-\infty}^{\infty} R_X(k)e^{-j2\pi fkT}. \tag{6.2}$$

El término $\sum_{k=-\infty}^{\infty} R_X(k)e^{-j2\pi fkT}$ corresponde a la DTFT (*Discrete Time*

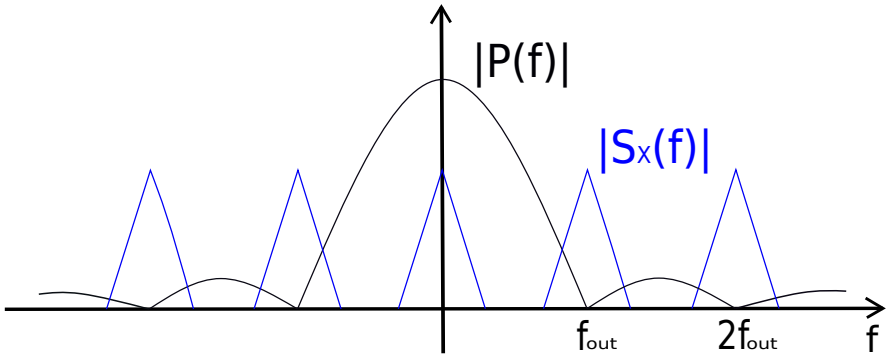


Figura 6.4: Densidad espectral de potencia de la señal multiplicada por densidad espectral de potencia del pulso conformador.

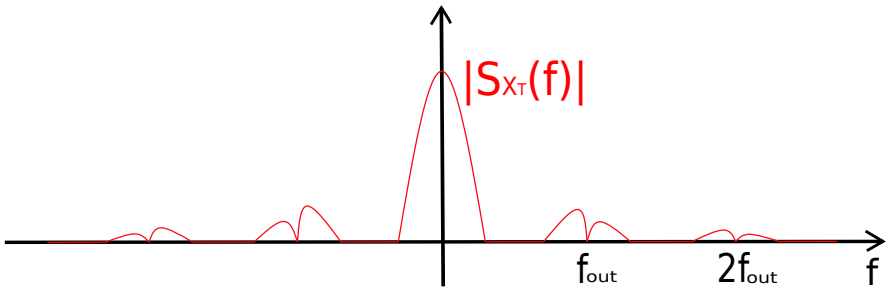


Figura 6.5: Densidad espectral de potencia de la señal enviada por el adaptador USB-VGA.

Fourier Transform) de la autocorrelación de $X(n)$, evaluada en $\theta = 2\pi fT$

$$S_X(e^{j\theta})|_{\theta=2\pi fT} = \sum_{k=-\infty}^{\infty} R_X(k)e^{-j2\pi fkT}. \tag{6.3}$$

Sustituyendo 6.3 en 6.2 obtenemos:

$$S_{x_T}(f) = \frac{|P(f)|^2}{T} S_X(f). \tag{6.4}$$

Sustituyendo $P(f)$ con la Transformada de Fourier del pulso cuadrado, el cual es nuestro pulso conformador, llegamos a que:

$$S_{x_T}(f) = T|sinc(fT)|^2 S_X(f). \tag{6.5}$$

Si analizamos la señal en el espectro, $S_X(e^{j\theta})$ es una señal periódica, que se repite en $f = \frac{1}{T}$, la cual se nombró f_{out}. En esa misma frecuencia se dan los ceros de $|sinc(fT)|^2$.

La multiplicación de los espectros hace que el espectro de X se atenúe y deforme en los múltiplos de f_{out}. En las figuras 6.4 y 6.5 se aprecia el espectro resultante de conformar nuestra señal de entrada con los pulsos $p(t)$.

Como fue explicado al principio de esta sección, los resultados del estudio de la transmisión de los píxeles por el adaptador USB-VGA, es el mismo que para

cada color RGB por separado. Para el caso de la transmisión con osmo-fl2k, se utilizará solo el pin asociado al color rojo para transmitir, por lo que la señal será procesada por el adaptador como se acaba de explicar. En la siguiente sección se dará en detalle cómo es la señal X de entrada al adaptador USB-VGA.

6.3. Transmisión de *transport streams* bajo la norma ISDB-T

El proyecto osmo-fl2k cuenta con un módulo para transmitir televisión digital terrestre bajo la norma europea (DVB-T). El mismo fue adaptado para transmitir *transport streams* ISDB-T. La explicación del funcionamiento del transmisor a través del adaptador USB-VGA será explicada en base al transmisor one-seg, aunque también fue implementado para transmitir full-seg. El transmisor one-seg genera complejos a una frecuencia $f_{ifft} = \frac{512m}{1008\mu s} \approx 507{,}94$ kHz, los cuales están modulados bajo la norma ISDB-T. El ancho de banda de la señal one-seg es de aproximadamente 428,57 kHz, como ya fue mencionado en la sección 4.2. A esta última frecuencia la llamaremos $f_{isdb-t} = 428{,}57$ kHz.

Dado que se generan valores complejos, es necesario volver a procesarlos de manera de obtener valores reales para poder ser enviados al aire. Este paso no sería necesario si la transmisión de los datos se hiciera a través de un SDR, dado que los drivers del equipo se encargan de procesar los datos complejos para poder transmitirlos. Para poder enviar los valores generados por el transmisor one-seg, se representan los complejos en fase y cuadratura, lo cual consiste en multiplicar la parte real de los complejos por un coseno y la parte imaginaria por un seno, ambos con el mismo argumento. Debido a la propiedad de ortogonalidad de las señales portadoras, es posible recuperarlas por separado. No solo es necesario representar los datos complejos de manera real para poder transmitirlos por el aire, sino que también hay que aumentar la frecuencia a la cual están las muestras (f_{ifft}) hasta llegar a la frecuencia de muestreo a la que trabaja el adaptador USB-VGA (f_{out}). Ambos procesos, en conjunto con la generación de la señal ISDB-T one-seg, se llevan a cabo en la etapa que se muestra en la figura 6.6, el cual recibe el nombre de `fl2k_transmit_isdb-t.grc`. Esta es la etapa completa de procesamiento de los datos a enviar a través del osmo-fl2k.

El bloque `Polyphase Arbitrary Resampler` es el encargado de elevar la frecuencia a la cual se encuentran las muestras ISDB-T hasta la frecuencia de muestreo del adaptador USB-VGA. El mismo se puede observar en la figura 6.7. Este bloque toma un flujo de datos y realiza un muestreo arbitrario. La tasa de muestreo puede ser cualquier número real r. El re-muestreo se realiza construyendo M filtros donde M es la tasa de interpolación. Para cada valor, se toma una salida del filtro actual, i, y el siguiente filtro i + 1 y luego se interpola linealmente entre los dos en función de la tasa de muestreo real que es deseada [31]. La tasa de interpolación se fija de manera que la frecuencia de las muestras de los datos quede en f_{out}:

$$f_{ifft}.N = f_{ifft}.\frac{f_{out}}{f_{ifft}} = f_{out}. \tag{6.6}$$

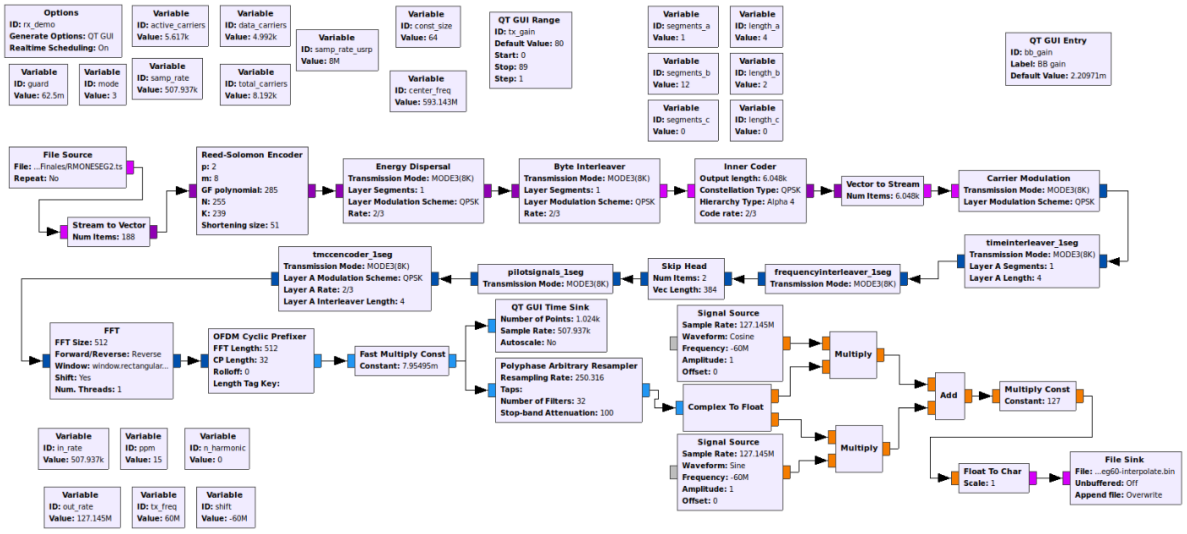


Figura 6.6: Diagrama de bloques de archivo `fl2k_transmit_isdb-t.grc`.

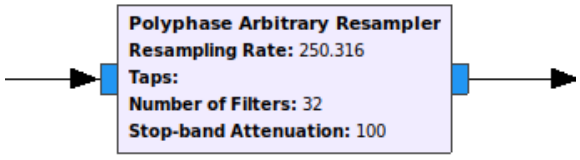


Figura 6.7: Re-muestreo de datos.

La siguiente etapa del proceso es la de fase y cuadratura, en la cual se representan los valores complejos con valores reales. Los bloques `Complex to Float`, `Signal Source`, `Multiply` y `Add` son los encargados de realizar dicho proceso. El bloque `Complex to Float` convierte la secuencia de complejos de la entrada en 1 o 2 secuencias de flotantes. Las salidas de flotantes corresponden como la salida (0) que es la parte real del complejo de entrada y la salida (1) que es la parte imaginaria [32]. El bloque `Signal Source` es un generador de señales, en el cual se le puede especificar el tipo de onda y las características de la señal como frecuencia, amplitud y *offset*. En nuestro caso se utilizan dos de estos bloques, uno para generar un coseno y otro para generar un seno, ambos con la misma frecuencia y amplitud. A la frecuencia a la cual se encuentran las sinusoides la llamaremos f_{cent}. Los bloques `Multiply` y `Add`, multiplican y suman sus entradas respectivamente. En la figura 6.8 se muestra cómo se interconectan los bloques para lograr la representación real de los complejos.

Llamamos X a la señal de entrada a los bloques de la figura 6.8, e Y a la salida de los mismos. El espectro de la señal X correspondiente a la figura 6.9, pasa a centrarse en la frecuencia f_{cent} con su parte en fase y su parte en cuadratura, como se puede ver en la figura 6.10. La etapa de fase y cuadratura no solo se utiliza para representar los valores complejos en reales, sino que también permite elegir

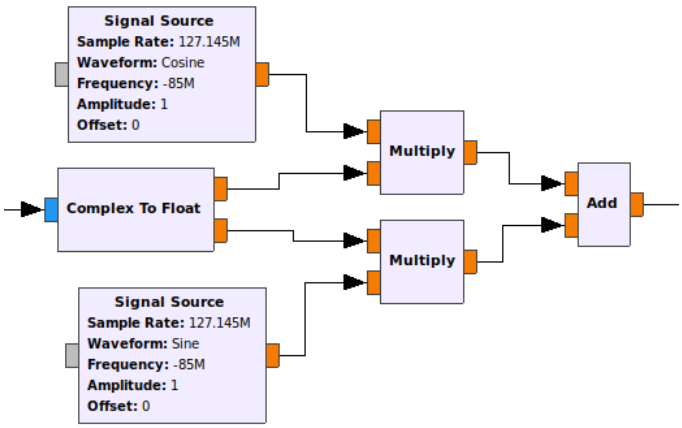


Figura 6.8: Diagrama de bloques de conversión en fase y cuadratura.

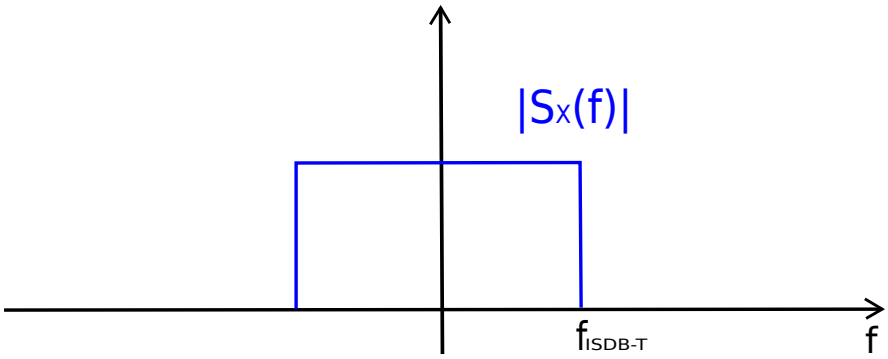


Figura 6.9: Densidad espectral de potencia de la señal en banda base.

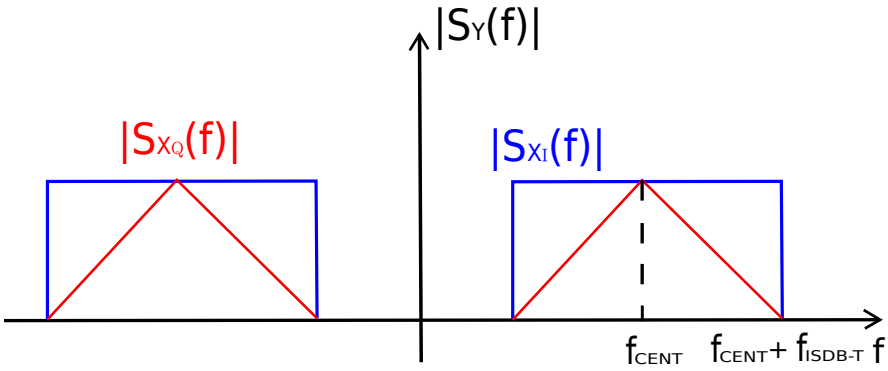


Figura 6.10: Densidad espectral de potencia de la señal en fase y cuadratura.

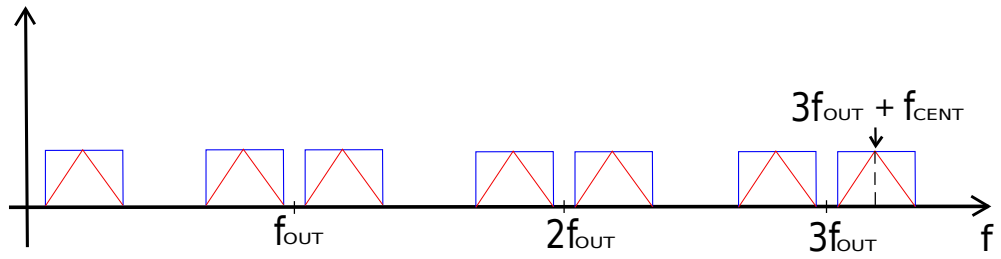


Figura 6.11: Densidad espectral de potencia de la señal en fase y cuadratura replicada cada múltiplos de f_{out}.

donde situar el espectro de la señal.

Como fue explicado en la sección 6.2, osmo-fl2k replicará el espectro a transmitir cada múltiplos de f_{out}, siendo f_{out} la frecuencia de muestreo a la cual está trabajando el adaptador USB-VGA. Combinando estos dos efectos, el espectro de la señal transmitida a la que llamaremos x_T no estará centrada en los múltiplos de f_{out}, sino que estará en $n.f_{out} \pm f_{cent}$, siendo n el número de armónico de la señal PAM y f_{cent} la frecuencia de las sinusoides que se utilizan para la representación real de los complejos. Dado que f_{out} y f_{cent} son variables, se puede elegir ambas de manera que el espectro esté en una frecuencia deseada. Por ejemplo, si se quiere que el espectro de la señal se encuentre en $f_c = 470$ MHz, se puede elegir $f_{out} = 138$ MHz y $f_{cent} = 56$ MHz. Con esta elección, el tercer armónico se ubicará en la frecuencia seleccionada. f_c depende del armónico n, por lo que se denota $f_c(n)$. En la figura 6.11 se puede visualizar este ejemplo.

$$f_c(n) = n.f_{out} \pm f_{cent} \tag{6.7}$$

$$\begin{aligned} f_c(3) &= 3.(138\ MHz) + 56\ MHz \\ &= 470\ MHz \end{aligned} \tag{6.8}$$

Finalmente para transmitir el archivo de video del transmisor one-seg procesado por `fl2k_transmit_isdb-t.grc` se utiliza en una terminal de Linux la función `fl2k_file`, creada por el proyecto osmo-fl2k. Dicha función envía el archivo por el pin asociado al color rojo del adaptador USB-VGA a través del comando:

```
fl2k_file -s fout file.bin
```

En donde f_{out} debe ser sustituido por la frecuencia de muestreo a utilizar en el adaptador USB-VGA. Dicha frecuencia debe ser tal que sea igual a una fracción multiplicada por la frecuencia de reloj del PLL (*Phase Locked Loop*) del adaptador. La cual es 160 MHz, con cierta tolerancia. El límite superior de f_{out} lo definen en conjunto el controlador USB al que esté conectado el adaptador USB-VGA, y este último. Específicamente, la cantidad máxima de muestras por segundo que el controlador pueda entregar al adaptador, y este pueda recibir, es lo que determina el valor máximo posible de f_{out}. Este límite se posiciona en el entorno de los 150 MHz

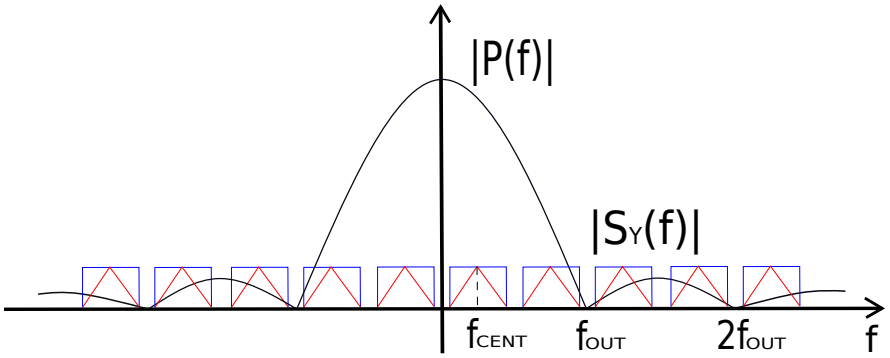


Figura 6.12: Densidad espectral de potencia de la señal transmitida por el adaptador USB-VGA.

para los controladores USB 3.0 convencionales que se encuentran en las computadoras personales [7]. El archivo *file.bin* es la salida de `fl2k_transmit_isdb-t.grc`, el cual es el video del transmisor one-seg procesado.

Luego de ejecutado este comando la señal se transmite a través del color rojo del USB-VGA como fue explicado en la sección 6.2. La misma será codificada con 8 bits, los cuales son la resolución del conversor digital-analógico del adaptador. La señal enviada se verá atenuada por los distintos lóbulos de $|sinc(fT)|^2$, como puede observarse en la figura 6.12. Esto hace que exista un compromiso con respecto a la potencia que tendrá la señal y la frecuencia que se seleccione para transmitir. Cuanto más alta sea la frecuencia, la atenuación en consecuencia de conformar con el pulso será mayor. Esto hace que los armónicos con más potencia sean los que se encuentran en el lóbulo principal del $|sinc(fT)|^2$, correspondientes a las frecuencias menores a f_{out}. Para el caso de televisión digital, la señal más potente que se puede lograr con la transmisión a través del USB-VGA, sin amplificadores, se encuentra en 177,143 MHz, correspondiente al canal más bajo al que pueden sintonizar los equipos que trabajan bajo la norma ISDB-T (canal 7).

En la figura 6.13 se muestra la señal enviada al aire. Como fue explicado a lo largo del este capítulo, el espectro es replicado cada f_{out}, lo cual resulta no eficiente, dado que interfiere en frecuencias en las cuales no es de interés transmitir. Además de no ser eficiente, surgen problemas legales por la utilización de frecuencias no permitidas, reguladas por la Unidad Reguladora de Servicios de Comunicaciones (URSEC). Esto último depende de la potencia a la que se este transmitiendo, si se utilizan amplificadores, la interferencia en otras frecuencias será mayor. Dado que el espectro de la señal transmitida es teóricamente infinito, se debe solucionar la transmisión de los armónicos no deseados. Una posible solución es la utilización de filtros pasa-banda, seleccionando la banda pasante en la frecuencia que se pretenda transmitir. En el capítulo 7 muestran los resultados de utilizar un filtro pasa-alto para atenuar los armónicos más bajos no utilizados, correspondientes a los de mayor potencia.

En este capítulo se presentó el estudio teórico del funcionamiento e implementación del adaptador USB-VGA como transmisor de televisión digital. A su vez se expuso el procesamiento que se debe hacer a la señal codificada bajo la norma

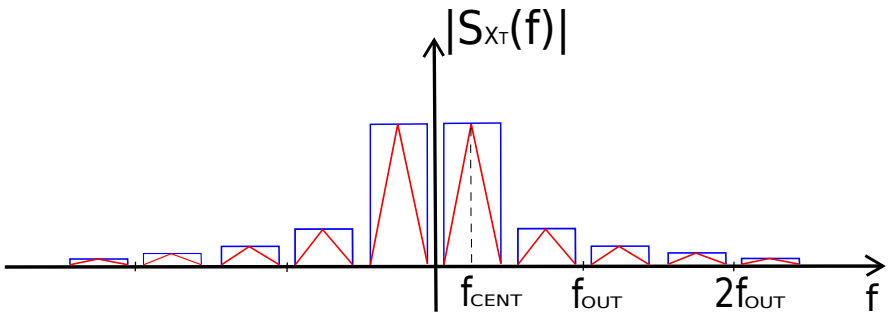


Figura 6.13: Densidad espectral de potencia de la señal transmitida por el adaptador USB-VGA.

ISDB-T para ser enviada por dicho *hardware*. En capítulos posteriores se explicarán las pruebas llevadas a cabo de funcionamiento del adaptador USB-VGA como transmisor ISDB-T, constatando el estudio teórico realizado.

Capítulo 7

Transmisión de *transport streams* bajo la norma ISDB-T a través de adaptador USB-VGA

En el capítulo 6 se dieron las bases teóricas del funcionamiento del adaptador USB-VGA como transmisor ISDB-T. La utilización del *hardware* alternativo como equipo transmisor requiere de un estudio práctico con respecto a su alcance, distorsión de la señal transmitida, entre otros. En este capítulo, se presentarán las pruebas prácticas necesarias para analizar el desempeño del equipo.

7.1. Hardware utilizado

El pin del adaptador USB-VGA que emite la señal que se desea enviar, es el número 1, correspondiente al rojo como se ve en la figura 7.1. Con el fin de emitir la mayor cantidad de potencia posible hacia el receptor, se le conectan diferentes equipos a dicho pin.

Los dispositivos utilizados son:

- Adaptador USB 3.0 a VGA basado en el chip Fresco Logic FL2000 (foto en figura 7.2).
- Adaptador VGA a BNC (foto en figura 7.3).
- Adaptador BNC a SMA (foto en figura 7.4).
- Amplificador de radiofrecuencia CommScope sva15prsms (foto en figura 7.5). Opera en la banda de frecuencias comprendidas entre 53 –1002 MHz y tiene una ganancia de 15,00 dB [33].
- Antena monopolo de $\frac{1}{4}$ de longitud de onda (foto en figura 7.6).

Para realizar una transmisión sin utilizar el amplificador, se conectan los equipos como se aprecia en la figura 7.7. Para las pruebas con amplificador, se conecta el mismo entre el adaptador BNC-SMA y la antena monopolo.

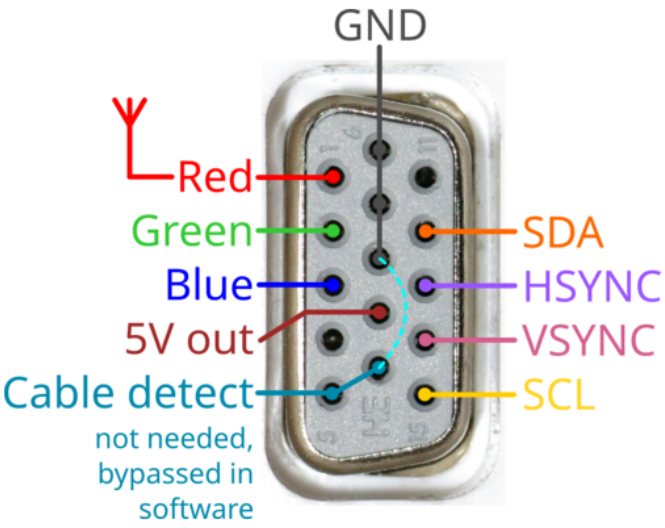


Figura 7.1: Pines del adaptador USB-VGA.

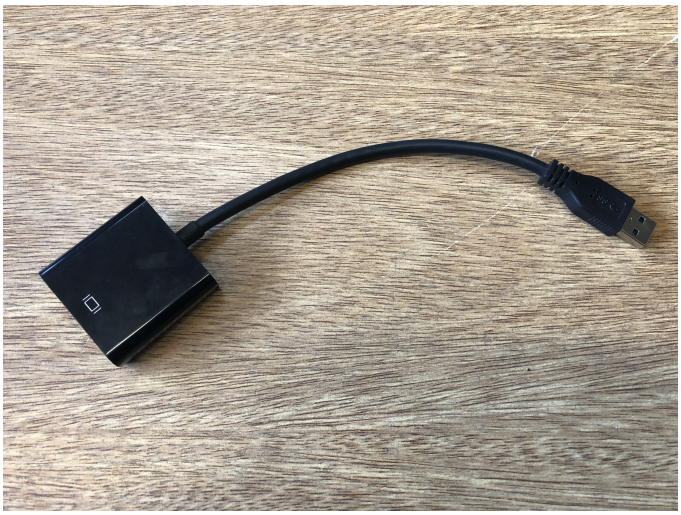


Figura 7.2: Adaptador USB-VGA.

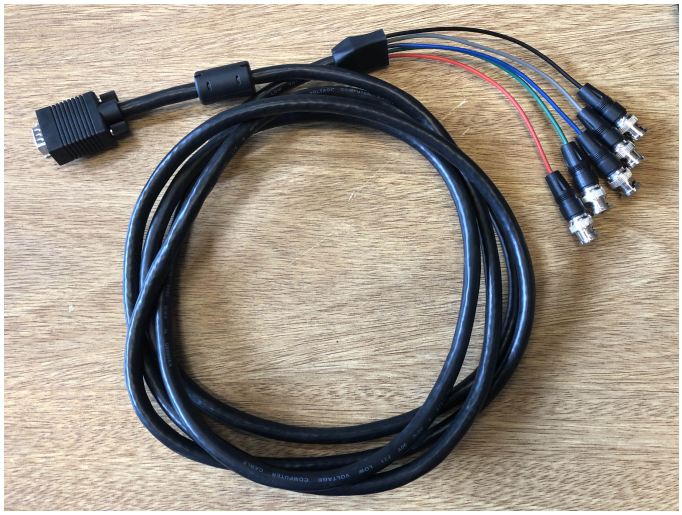


Figura 7.3: Adaptador VGA a BNC.

Figura 7.4: Adaptador BNC a SMA.

Figura 7.5: Amplificador CommScope sva15prsms.

Figura 7.6: Antena monopolo de $\frac{1}{4}$ de longitud de onda.

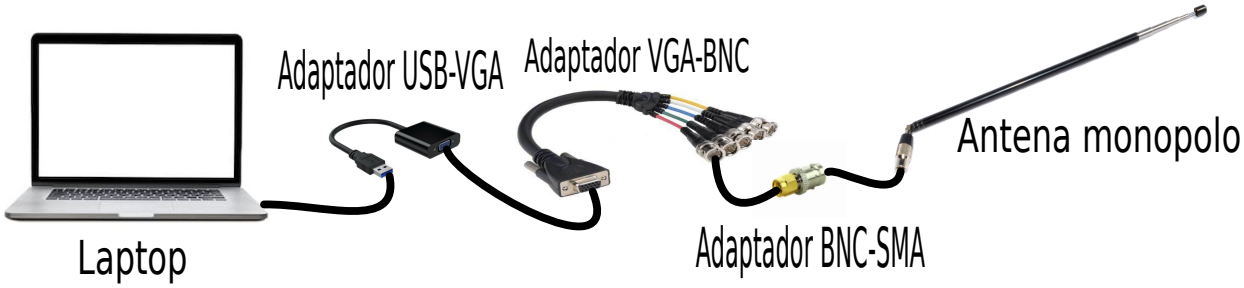


Figura 7.7: Diagrama de conexión.

7.2. Frecuencia de trabajo y configuración de antenas

Tal como se expresó en el capítulo 6, para una transmisión con la mayor potencia posible, es necesario utilizar el menor armónico disponible de los generados en la transmisión con el adaptador USB-VGA. Esto se puede visualizar gráficamente en la figura 6.13, en la cual el armónico mas cercano a DC es el que posee la mayor potencia.

Los equipos que complementen al adaptador USB-VGA como sistema transmisor, antenas y amplificadores, deben lograr trabajar en el rango de frecuencias en el que se decida transmitir. La frecuencia en la que se centrará el armónico 0, por lo explicado en el capítulo 6, se encontrará en el rango 0-f_{out}. Definido el rango en el que se encontrará el armónico 0 de la señal, es posible seleccionar una antena de transmisión que trabaje correctamente en ese rango, y configurarla de forma que emita con la mayor potencia posible a la frecuencia deseada. Una posible antena ampliamente utilizada y económica, que se complemente correctamente con el transmisor de bajo costo, puede ser un monopolo telescópico de $\frac{1}{4}$ de longitud de onda.

7.3. Pruebas de transmisión

Las pruebas de transmisión utilizando el $USB-VGA$ como equipo transmisor se dividieron en tres:

- Transmisión de video one-seg sin amplificador.
- Transmisión de video full-seg sin amplificador.
- Transmisión de video full-seg con amplificador.

Las mismas se llevaron a cabo en el Laboratorio de Medidas del Instituto de Ingeniería Eléctrica, como se observa en la figura 7.8. Se dejó el equipo receptor fijo en la habitación y se fue variando la ubicación del transmisor para las distintas pruebas, manteniendo línea de vista entre las antenas de los equipos.

Los parámetros del transmisor ISDB-T para todas las pruebas se encuentran en la tabla 7.1. La elección de estos se basa en que son parámetros típicos utilizados por las operadoras locales de televisión digital terrestre.

Modo	Intervalo de guarda	Código convolucional capa A (one-seg)	Código convolucional capa B (full-seg)	Entrelazamiento temporal capa A (one-seg)	Entrelazamiento temporal capa B (full-seg)
3	1/16	2/3	3/4	4	2

Tabla 7.1: Parámetros utilizados para las pruebas en transmisor ISDB-T.

Figura 7.8: Laboratorio de Medidas del Instituto de Ingeniería Eléctrica.

7.3.1. Modulation Error Rate (MER)

Para poder verificar la calidad de la señal, se utilizará en este caso la tasa de error de modulación (MER), la cual es una medida que suma todas las interferencias que sufre la señal digital que se transmite. Esta es una herramienta cuantitativa que permite valorar cómo es de buena una señal digital modulada. El MER corresponde al cociente entre la suma de los módulos de los símbolos esperados (o teóricos) sobre el módulo de la suma de los vectores de error, los cuales serían la desviación de los símbolos recibidos con respecto a los teóricos.

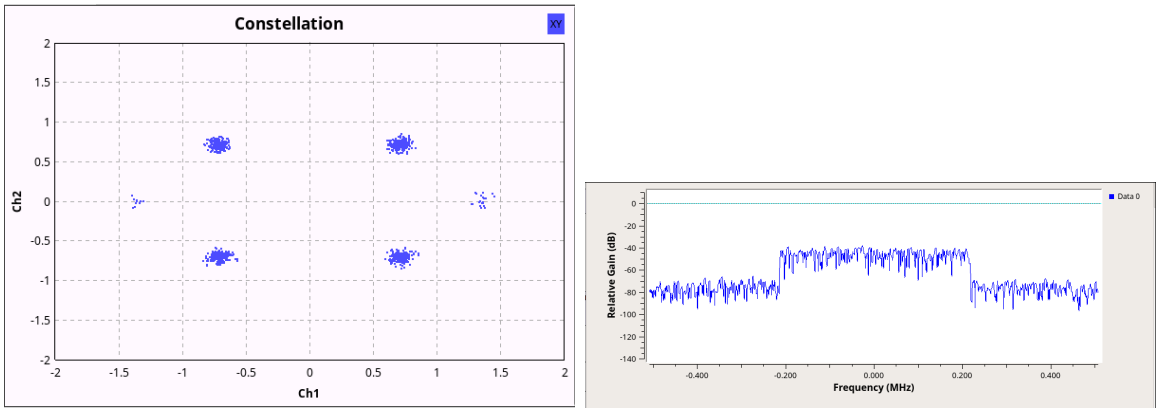
$$MER(dB) = 10 * log_{10}\left(\frac{\sum_{i=0}^{N-1} {I_i}^{*2} + {Q_i}^{*2}}{\sum_{i=0}^{N-1} \left(I_i^* - I_i\right)^2 + \left(Q_i^* - Q_i\right)^2}\right), \tag{7.1}$$

donde (I_i, Q_i) son la parte real e imaginaria respectivamente del i-ésimo símbolo complejo de un conjunto de N símbolos colectados. Los complejos (I_i^*, Q_i^*) corresponden al mapeo luego del bloque de decisión de los complejos antes mencionados.

Cuanto más alto es el valor del MER, mejor es la calidad de señal recibida. Para tomar dicha medida se utiliza el proyecto `gr-mer` [34] implementado por el Grupo Artes. El mismo es una implementación en GNU Radio que cuenta con un bloque para tomar medidas de MER, entre otros.

7.3.2. Transmisión one-seg sin amplificador

La frecuencia seleccionada para la transmisión one-seg fue 60 MHz. La elección de la misma fue para estar lo más cerca posible de $f = 0$ para que la señal a transmitir no se atenúe demasiado como fue explicado en el capítulo 6, pero que no sea tan baja como para que los equipos receptores puedan recibirla sin problemas.



(a) Constelación. (b) Amplitud de espectro recibido.

Figura 7.9: Transmisión one-seg con 60 cm de distancia entre transmisor y receptor.

Para esta prueba no se utilizó el amplificador, dado que se tomó como criterio que el sistema transmisor-receptor sea lo más económico posible.

Como equipo receptor se utilizó:

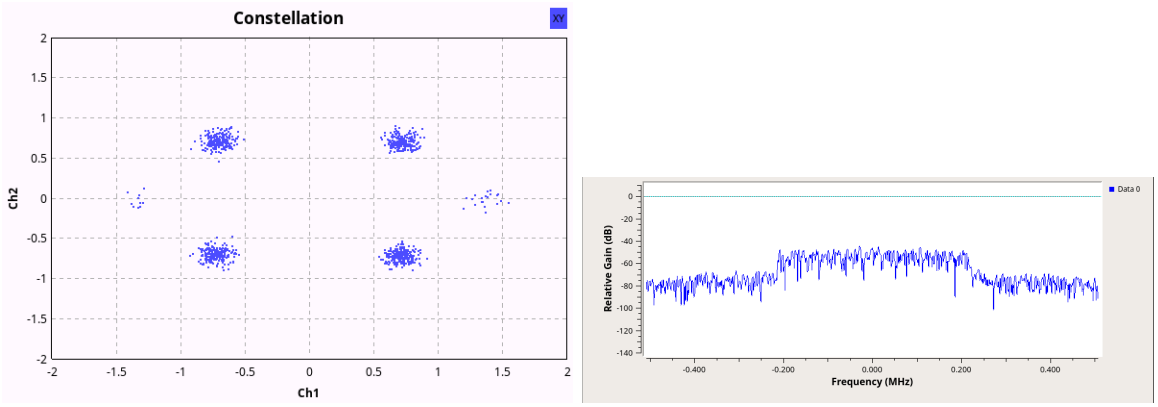
- Antena monopolo de cuarta longitud de onda.
- RTL-SDR [12].
- Laptop con procesador Intel Core i7-4712MQ y 16 GB de memoria RAM.

Como equipo transmisor se utilizó:

- Antena monopolo de cuarta longitud de onda.
- Adaptador USB-VGA.
- Laptop con procesador AMD Ryzen 5 3500U con una memoria RAM de 24 GB.

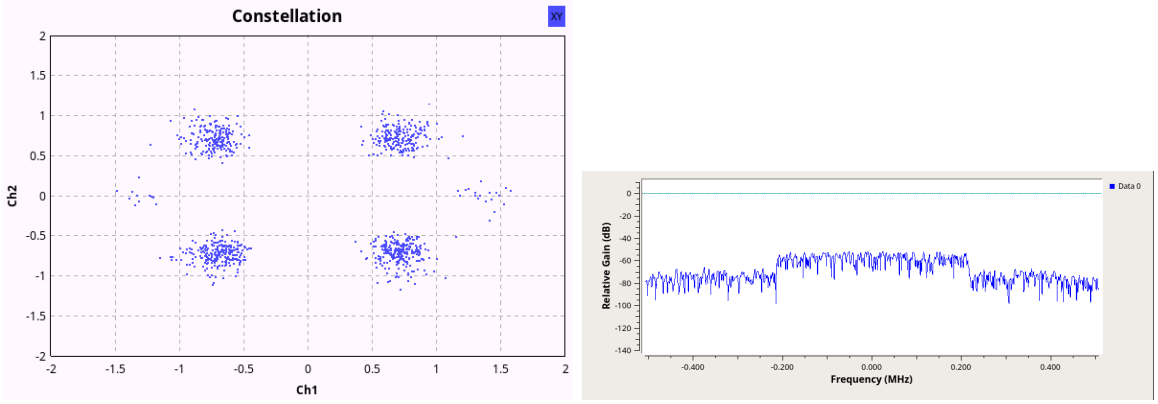
Para esta prueba se utilizó una modulación QPSK, con una ganancia en recepción de 20 dB, indicada en el bloque osmocom-source, el cual interconecta GNU Radio con el hardware RTL-SDR. La elección de la modulación se debe a que QPSK es más robusta con respecto a las otras modulaciones (16-QAM o 64-QAM), y dado que nuestra potencia es escasa, se optó por utilizar dicha modulación. Se fueron variando las distancias y tomando los resultados.

Se logró transmitir la señal one-seg de ISDB-T a través del adaptador USB-VGA de manera exitosa. La señal se logra recibir y demodular en tiempo real por el equipo receptor sin inconvenientes para todas las distancias entre los equipos involucrados. A medida que crece la distancia entre el transmisor y el receptor se observa cómo la potencia recibida disminuye, como puede observarse en las figuras 7.9, 7.10 y 7.11, lo cual era lo esperado. A su vez, en consecuencia de la disminución de la potencia recibida, la constelación se va distorsionando, pero en todos los casos mostrados se puede decodificar de manera correcta el *transport*



(a) Constelación. (b) Amplitud de espectro recibido.

Figura 7.10: Transmisión one-seg con 240 cm de distancia entre transmisor y receptor.



(a) Constelación. (b) Amplitud de espectro recibido.

Figura 7.11: Transmisión one-seg con 500 cm de distancia entre transmisor y receptor.

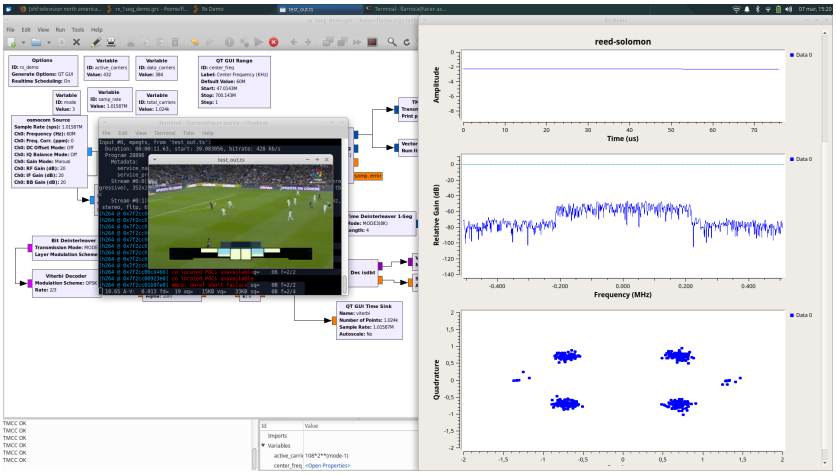


Figura 7.12: *Transport stream* one-seg a 240 cm de distancia entre transmisor y receptor.

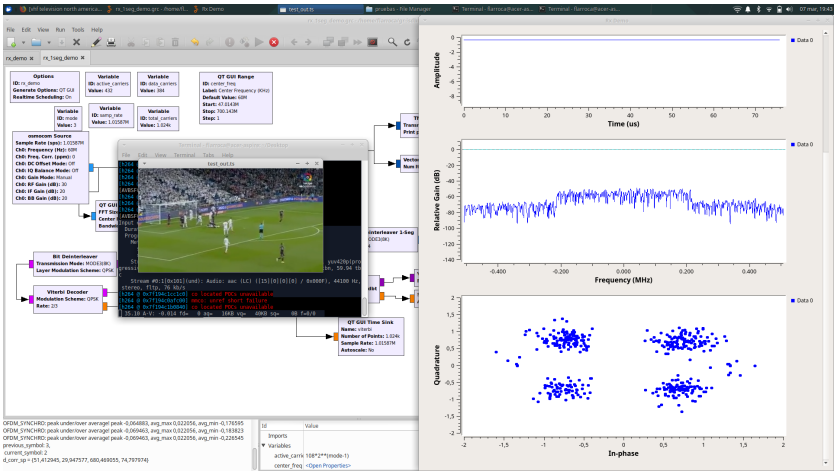


Figura 7.13: *Transport stream* de transmisión one-seg a 2000 cm de distancia entre transmisor y receptor.

stream. En la figura 7.12 se observa un ejemplo de la señal ISDB-T decodificada. Se continuó tomando las medidas de MER para ciertas distancias. Como era de esperarse, la misma disminuye a medida que la distancia aumenta, tal como se puede ver en la tabla 7.2.

Distancia (cm)	60	240	500
MER (dB)	24,5	21,6	15,7

Tabla 7.2: Medidas de MER en transmisión one-seg.

Se realizó una prueba de alcance máximo del sistema, para constatar cuál era la máxima distancia entre el transmisor y el receptor de manera de que la señal ISDB-T se pudiera decodificar correctamente. Para esto se dejó fijo el equipo transmisor y se fue alejando el equipo receptor. La prueba debió ser realizada fuera del laboratorio de medidas, dado que las dimensiones del mismo no permitían hacer la prueba correctamente. La máxima distancia en la que se logró transmitir y recibir de manera correcta fue de 2000 cm, lo que se puede apreciar en la figura 7.13. La correcta recepción de la señal enviada hace referencia a que se podía reproducir el *transport stream* de manera correcta, sin cortes en la reproducción.

Por último se modificó la modulación a 64-QAM para realizar una prueba a 60 cm de distancia entre el transmisor y el receptor para verificar su funcionamiento. Esta modulación no fue utilizada para el resto de las pruebas, dado que requiere una mayor potencia de transmisión para que el receptor pueda decodificar de manera correcta el *transport stream* enviado. Esto se debe a que los puntos de la constelación se encuentran más cerca entre sí. A pesar de esto, a una distancia de $60cm$, funcionó correctamente como se puede ver en la figura 7.14.

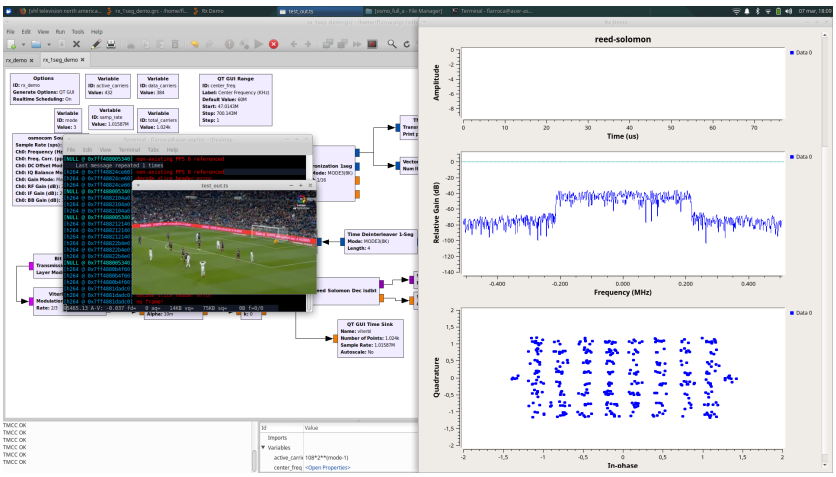


Figura 7.14: *Transport stream* de transmisión one-seg a 60 cm de distancia entre transmisor y receptor utilizando modulación 64-QAM.

7.3.3. Transmisión full-seg sin amplificador

La frecuencia seleccionada para la transmisión full-seg fue 80 MHz. La elección de la misma se debe a que el amplificador tiene su rango de trabajo de 50 a 1000 MHz y será utilizado para comparar resultados para el caso sin amplificador. Se realizaron pruebas de transmisión con el amplificador a 60 MHz observándose que la amplificación era casi nula, por lo que se eligió la frecuencia de 80 MHz para tener un correcto funcionamiento.

Como equipo receptor se utilizó:

- Antena monopolo de cuarta longitud de onda.
- USRP B200 [13].
- Laptop con procesador Intel Core i7-4712MQ y 16GB de memoria RAM.

Como equipo transmisor se utilizó:

- Antena monopolo de cuarta longitud de onda.
- Adaptador USB-VGA.
- Laptop con procesador AMD Ryzen 5 3500U con una memoria RAM de 24GB.

Al igual que para la prueba con one-seg se utilizó una modulación QPSK, y una ganancia en recepción de 20 dB. Se fueron variando las distancias y tomando los resultados.

Al igual que en en el caso de transmisión one-seg, la potencia en recepción va disminuyendo a medida que la distancia entre el transmisor y el receptor crece, como se puede apreciar en las figuras 7.15, 7.16 y 7.17. Las pruebas revelaron buenos resultados, pudiendo recibir y demodular en tiempo real la señal ISDB-T

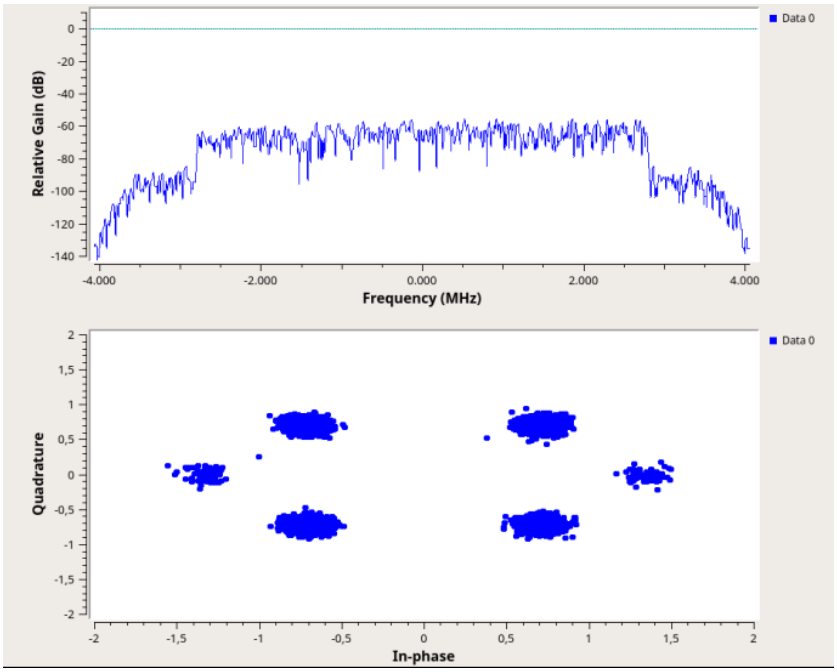


Figura 7.15: Transmisión full-seg con 60cm de distancia entre transmisor y receptor sin amplificador.

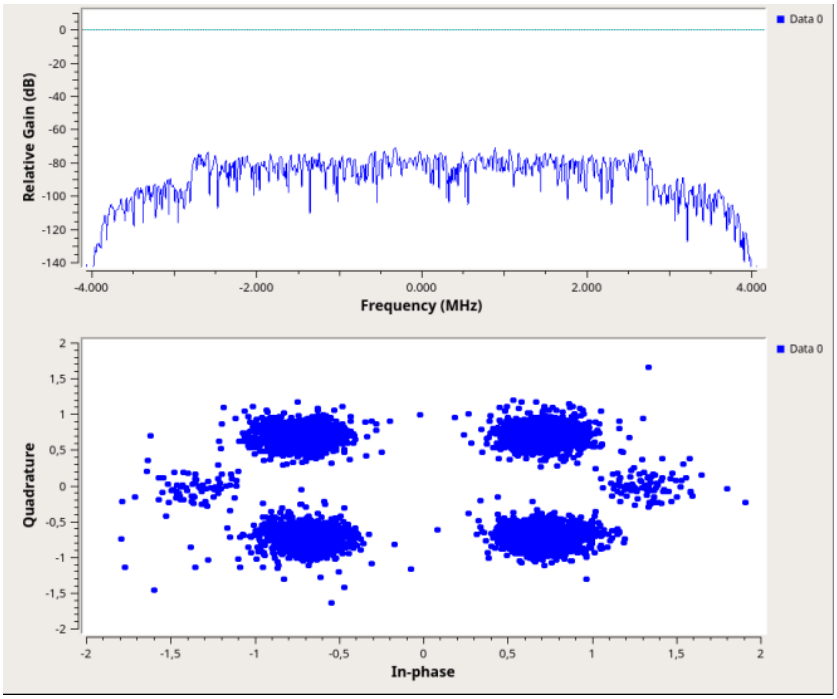


Figura 7.16: Transmisión full-seg con 240cm de distancia entre transmisor y receptor sin amplificador.

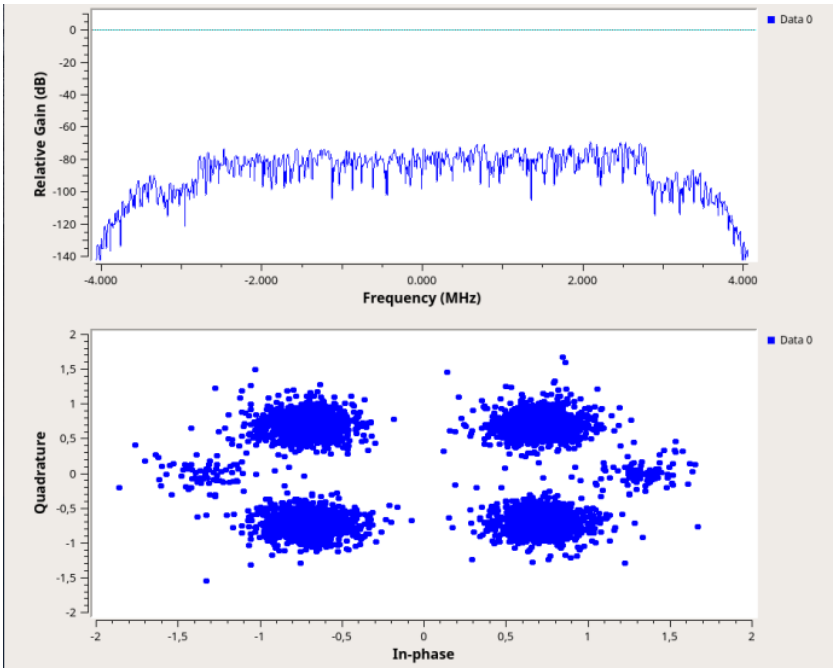


Figura 7.17: Transmisión full-seg con 500cm de distancia entre transmisor y receptor sin amplificador.

full-seg enviada a través del adaptador USB-VGA. Se lograron transmitir aproximadamente 5,571 MHz a través del adaptador para diferentes distancias entre el transmisor y el receptor. Un resultado importante, dado que en muchos equipos SDR, la limitante se encuentra en el ancho de banda de transmisión. Como se mencionó anteriormente, el ancho de banda transmitido en este caso es de aproximadamente 5,571 MHz, en cambio el de one-seg es de aproximadamente 428, 57 kHz. Dado que en ambos casos se utiliza la misma potencia total de transmisión, en el caso one-seg se recepciona mejor la señal que en el caso full-seg. Se tomaron medidas de MER para algunas distancias, los cuales se pueden apreciar en la tabla 7.3. Al recibirse con menor potencia que en el caso one-seg, se puede ver como las constelaciones están más distorsionadas, provocando peores valores de MER a mismas distancias.

Distancia (cm)	60	240	500
MER (dB)	18,1	17,1	13,7

Tabla 7.3: Medidas de MER en transmisión full-seg sin amplificador.

En la figura 7.18, se puede apreciar el *transport stream* decodificado a una distancia de 60 cm entre el transmisor y el receptor.

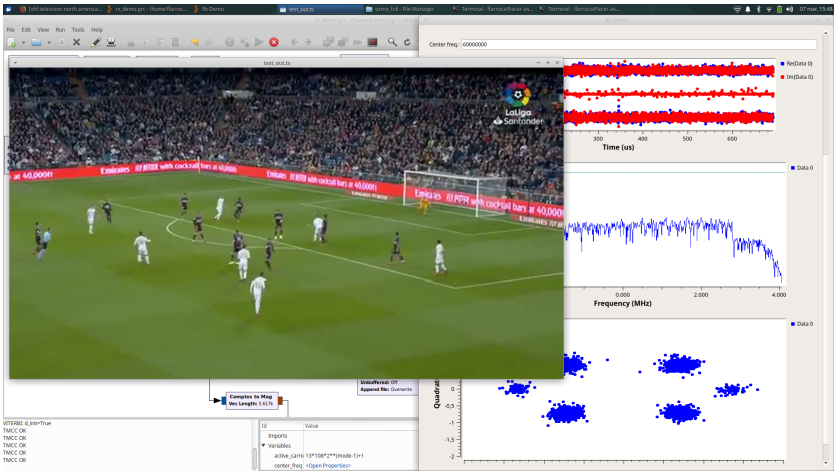


Figura 7.18: *Transport stream* de transmisión full-seg a 60cm de distancia entre transmisor y receptor sin amplificador.

7.3.4. Transmisión full-seg con amplificador

Esta prueba se realizó igual que la anterior, a diferencia de que se le agregó un amplificador, el cual amplifica 15 dB en la banda de 50 a 1000 MHz. Se pueden observar los espectros recibidos en las figuras 7.19, 7.20 y 7.21. A pesar de que la potencia de recepción es menor que si se utilizara el transmisor one-seg, se puede apreciar como los espectros y constelaciones recibidas mejoran si se compara con el caso full-seg sin amplificador. Estos efectos se ven reflejados a su vez en el MER (tabla 7.4), el cuál mejora con respecto del caso full-seg sin amplificador.

Distancia (cm)	60	240	500
MER (dB)	19,2	18,7	17,3

Tabla 7.4: Medidas de MER en transmisión full-seg con amplificador.

Como en las pruebas one-seg, se modificó la modulación a 64-QAM para realizar una prueba a 60 cm de distancia entre el transmisor y el receptor para verificar su funcionamiento, tal como se ve en la figura 7.23. Se demoduló y reprodujo el *transport stream* sin problemas para esa distancia.

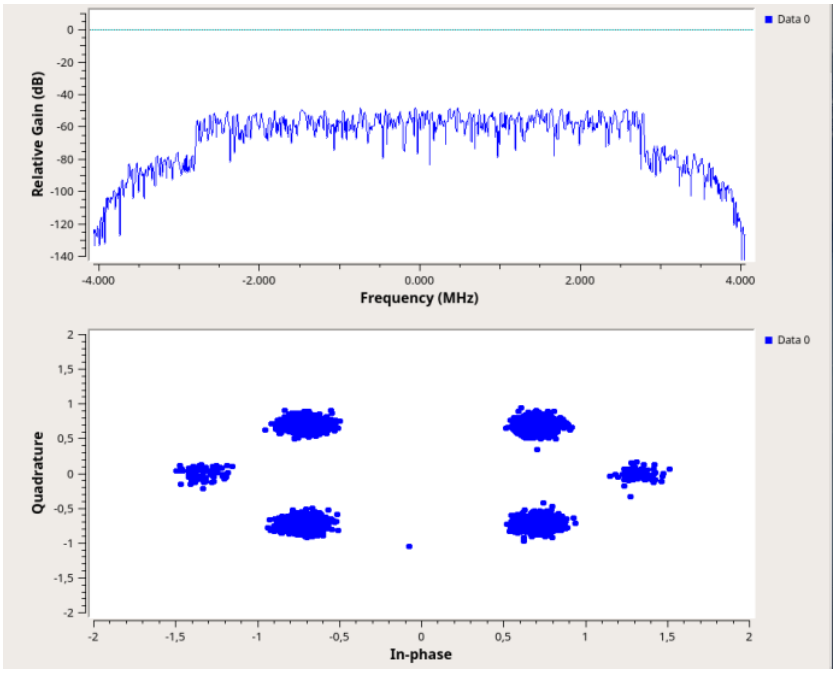


Figura 7.19: Transmisión full-seg con 60 cm de distancia entre transmisor y receptor con amplificador.

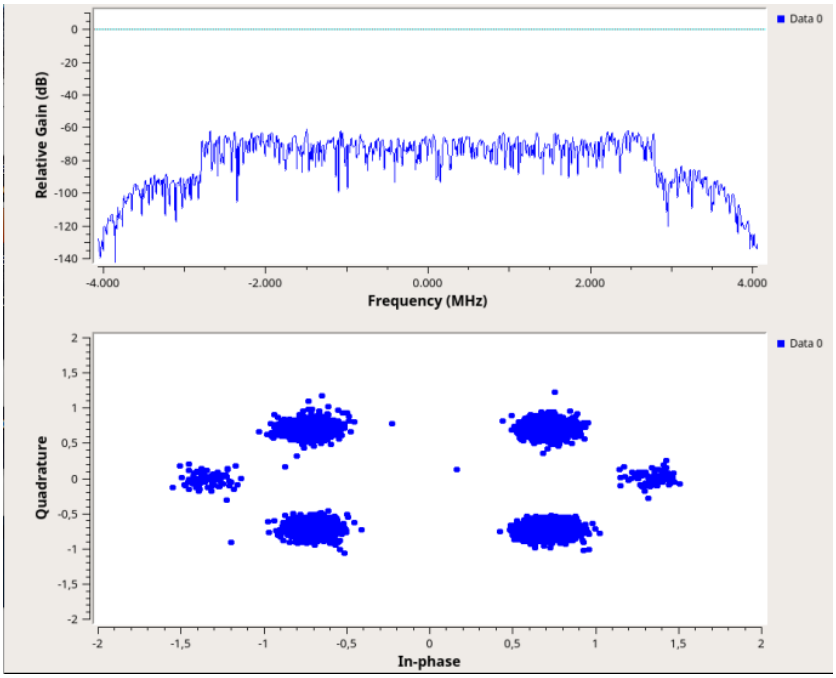


Figura 7.20: Transmisión full-seg con 240cm de distancia entre transmisor y receptor con amplificador.

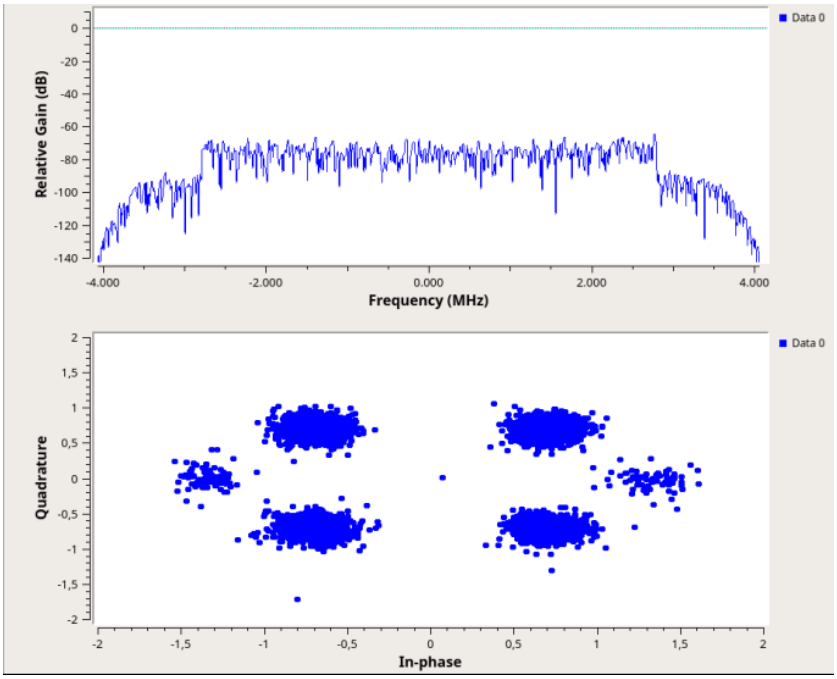


Figura 7.21: Transmisión full-seg con 500 cm de distancia entre transmisor y receptor con amplificador.

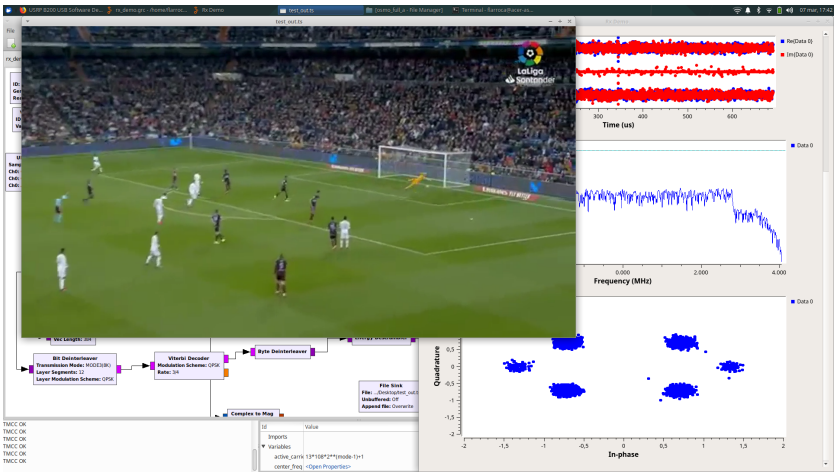


Figura 7.22: *Transport stream* de transmisión full-seg a 240 cm de distancia entre transmisor y receptor con amplificador.

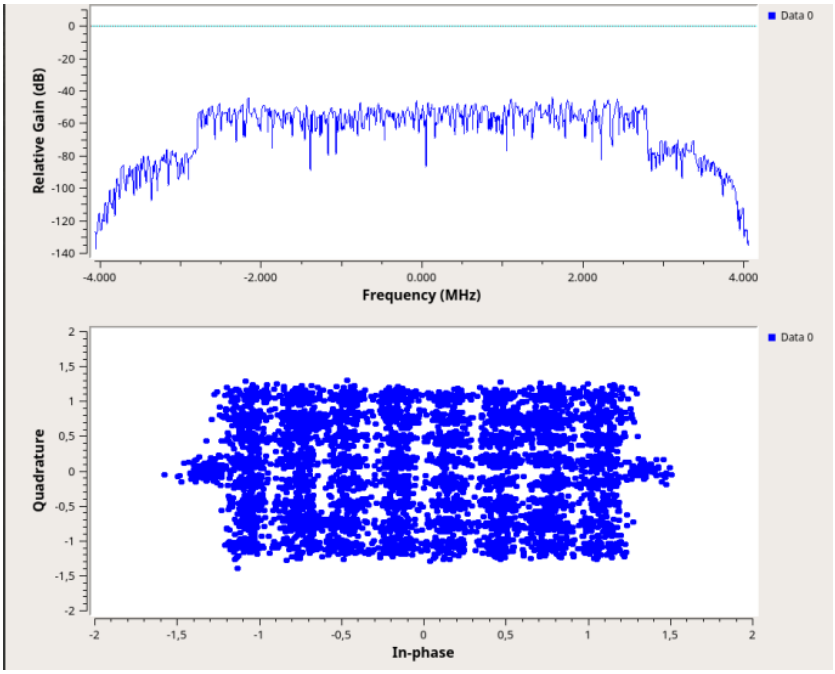


Figura 7.23: Modulación 64QAM en transmisión full-seg con amplificador, a 60cm de distancia entre transmisor y receptor.

7.3.5. Transmisión full-seg con amplificador y filtro de radio frecuencia

Como fue explicado en la sección 6.2, la señal transmitida por el adaptador USB-VGA tiene espectro teóricamente infinito, considerando las réplicas de la señal cada múltiplos de la frecuencia de muestreo del adaptador. Esto hace que el espectro de la señal a transmitir, ocupe frecuencias no deseadas, haciendo un uso ineficiente del espectro y en los peores casos interfiriendo otras señales. Dada la potencia de transmisión del hardware, los primeros armónicos son los que más interfieren en el espectro, dado que el resto de ellos se desvanecen muy rápidamente por el efecto de conformar con el pulso rectangular. Con el fin de solucionar este problema se decidió comprar un filtro pasa alto de radiofrecuencia, para lograr atenuar la réplica más baja de nuestra señal. Esta es la que presenta más potencia en el espectro, debido a que se sitúa más cerca de banda base (frecuencia igual 0).

El filtro utilizado es un TruSpec HPF-54MHz (se puede observar en la figura 7.24), el cual es un filtro pasa alto con una frecuencia de corte de 54 MHz. Su curva de respuesta en frecuencia se puede ver en la figura 7.25. Para estudiar el desempeño del filtro, se situó el espectro de la señal a 85 MHz, y se utilizó la frecuencia de muestreo del adaptador en 127,143 MHz. De esta manera, como se puede apreciar en la figura 7.26, la parte baja del armónico 1 de la señal se encuentra alrededor de los 42 MHz y la parte alta alrededor de los 212 MHz.

La parte alta del armónico 1, se encuentra bastante atenuada por el efecto de

Figura 7.24: TruSpec HPF-54MHz.

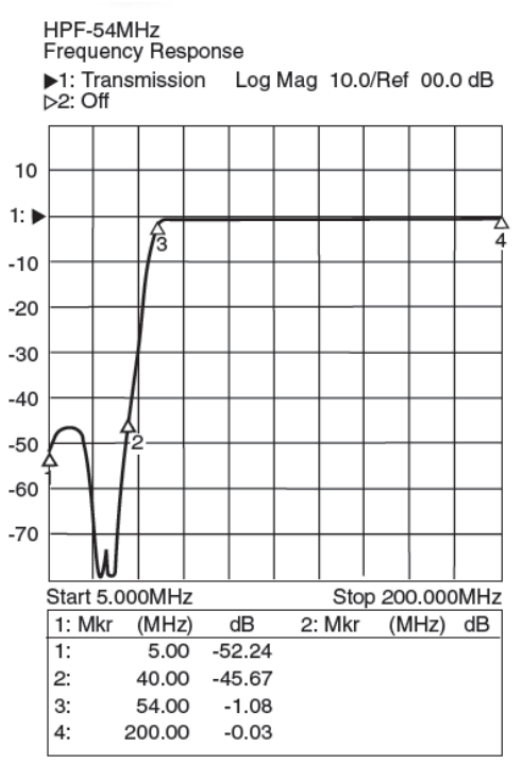


Figura 7.25: Respuesta en frecuencia del TruSpec HPF-54MHz.

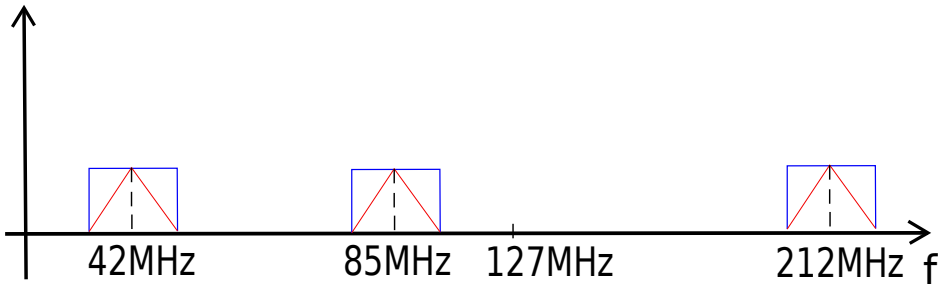


Figura 7.26: Espectro de la señal enviada con su primer armónico.

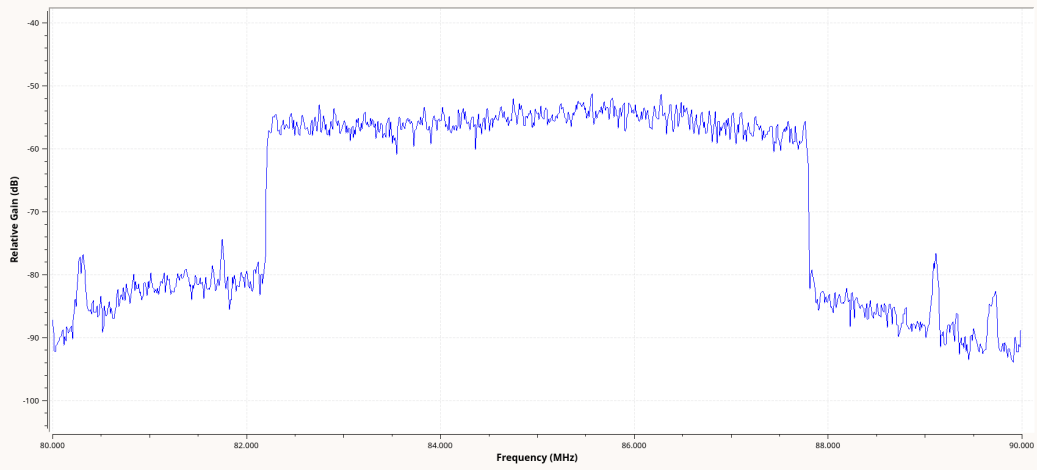


Figura 7.27: Espectro de la señal a 85 MHz sin filtro pasa alto.

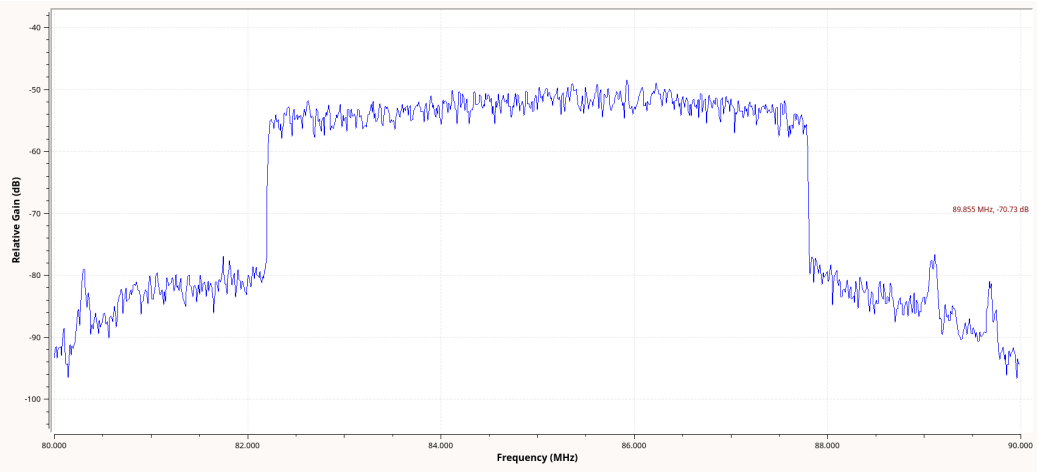


Figura 7.28: Espectro de señal a 85 MHz con filtro pasa alto.

conformar con el pulso rectangular. En cambio la parte baja del mismo, tiene una potencia comparable a la de la señal que se desea transmitir.

En las figuras 7.27, 7.28, 7.29, 7.30 y 7.31, se puede apreciar el efecto del filtro sobre la señal transmitida. Se ve cómo la réplica que se encuentra a 42 MHz es atenuada de manera considerable. La atenuación es tal, que a nivel de espectro se visualiza prácticamente lo mismo que si no existiera la transmisión. Todo esto sin producir ningún tipo de alteración en la señal que se transmite a 85 MHz. Dado que los otros armónicos se encuentran atenuados, por el simple hecho de la transmisión por el adaptador USB-VGA, se puede tomar como una posible solución del problema espectral la utilización de un filtro pasa alto como es este el caso.

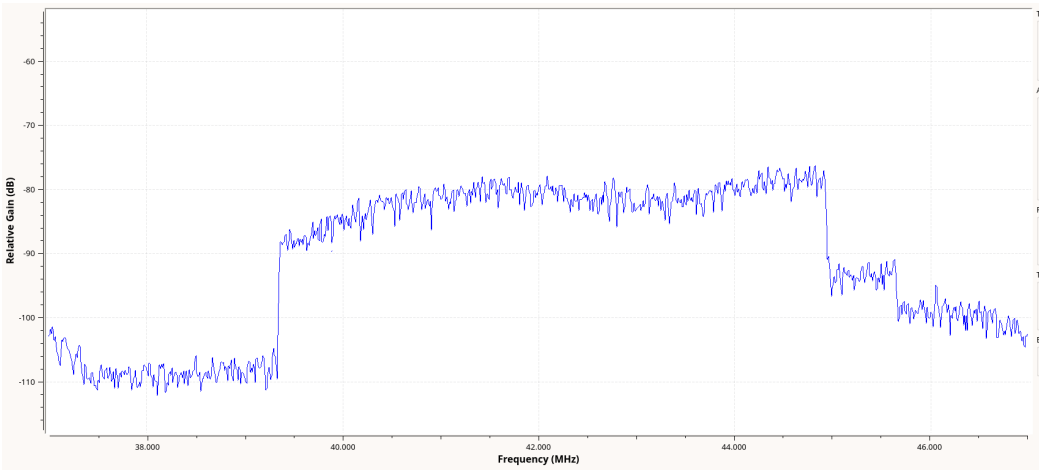


Figura 7.29: Espectro de la señal a 42 MHz sin filtro pasa alto.

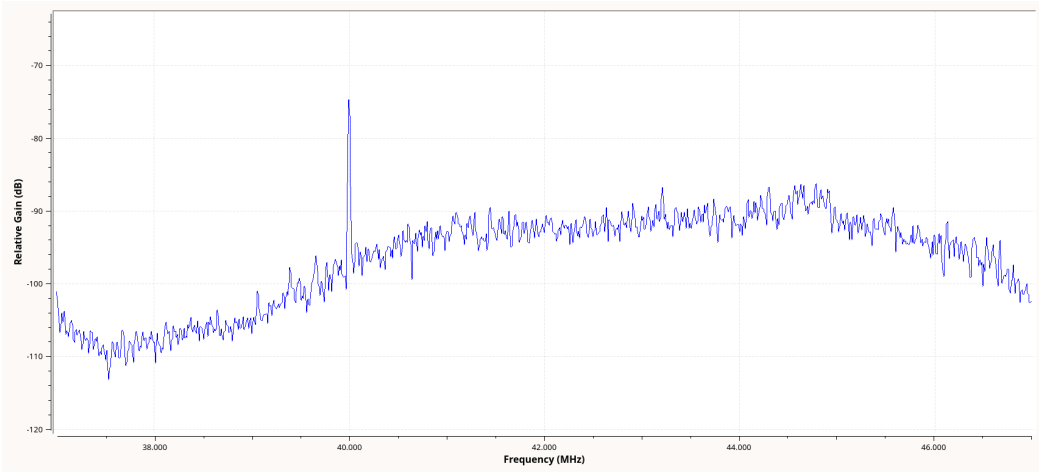


Figura 7.30: Espectro de la señal a 42 MHz con filtro pasa alto.

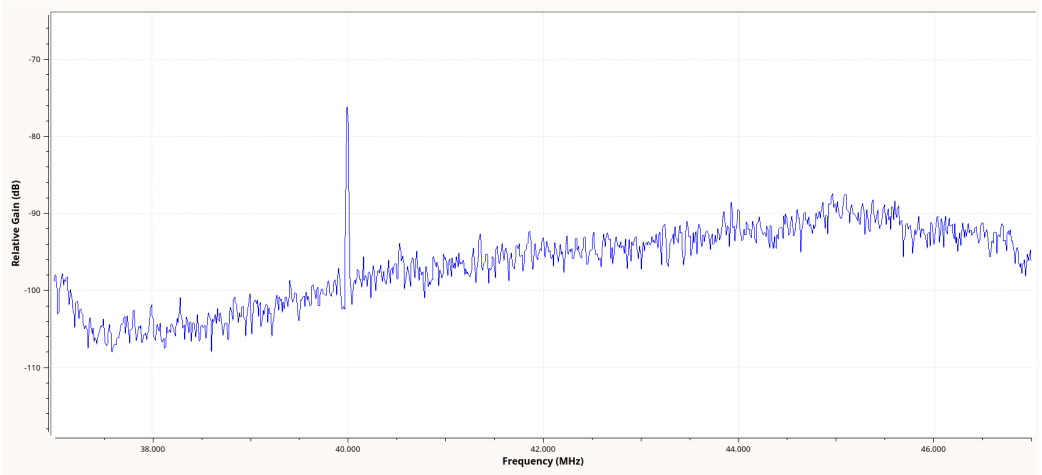


Figura 7.31: Espectro visualizado a 42 MHz sin transmisión.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 8

Experimentos del transmisor ISDB-T variando el hardware

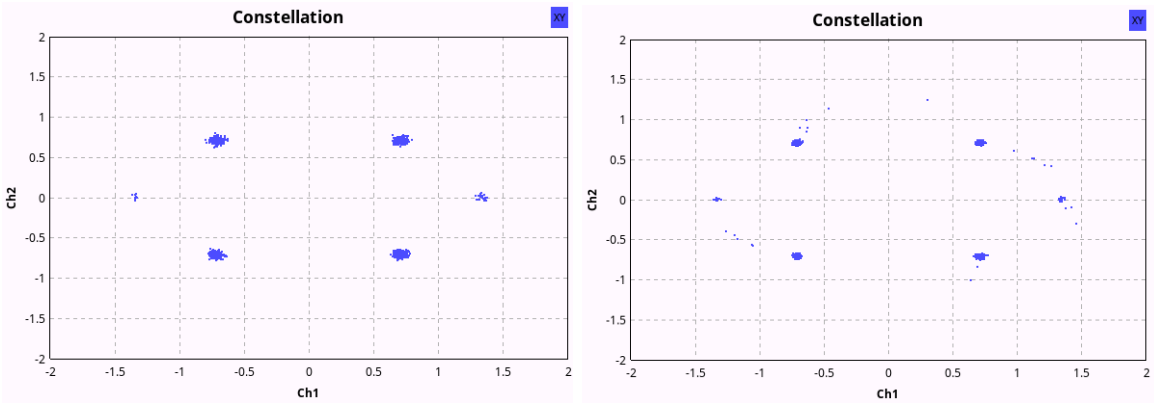
8.1. Comparación entre el hardware USB-VGA y equipos SDR comerciales

8.1.1. Introducción

La utilización del adaptador USB-VGA como *hardware* de transmisión ISDB-T tiene como principal fin abaratar de gran manera el equipamiento utilizado. La finalidad de esta sección es estudiar cuánto se gana o se pierde en temas de desempeño, utilizando un *hardware* o el otro. Para esto se realizaron distintas pruebas en iguales condiciones, intercambiando el equipo transmisor. Las comparaciones en transmisión se realizaron entre el adaptador USB-VGA y el equipo SDR USRP B200. Como receptor se utilizó un equipo RTL-SDR para las pruebas one-seg. Para las pruebas de full-seg se debió cambiar el equipo receptor por un USRP B100, dado que el ancho de banda necesario en recepción no era alcanzado por el RTL-SDR.

8.1.2. Transmisión one-seg

Para comenzar a comparar los equipos, se realizaron pruebas en las que la distancia entre el equipo transmisor y el equipo receptor era muy corta. La finalidad de dichas pruebas fue poder minimizar los efectos del canal, para poder observar posibles modificaciones que pueda sufrir la señal con el solo hecho de ser transmitida a través del equipo. Para una primera prueba se trabajó con el armónico 0 de la señal transmitida con el adaptador USB-VGA, el cual se encuentra en una frecuencia de 60 MHz. En la figura 8.1 se observan las constelaciones recibidas para los distintos casos. Se puede observar que las mismas son similares, y poseen buena exactitud respecto a lo esperado para modulaciones QPSK, BPSK y DBPSK. A pesar de esto, se visualiza que la constelación obtenida de la transmisión con USRP B200 es algo más precisa en las zonas asociadas a estos esquemas de



(a) Adaptador USB-VGA. (b) USRP B200.

Figura 8.1: Constelaciones transmitiendo a 60 MHz con distancias cercanas entre transmisor y receptor.

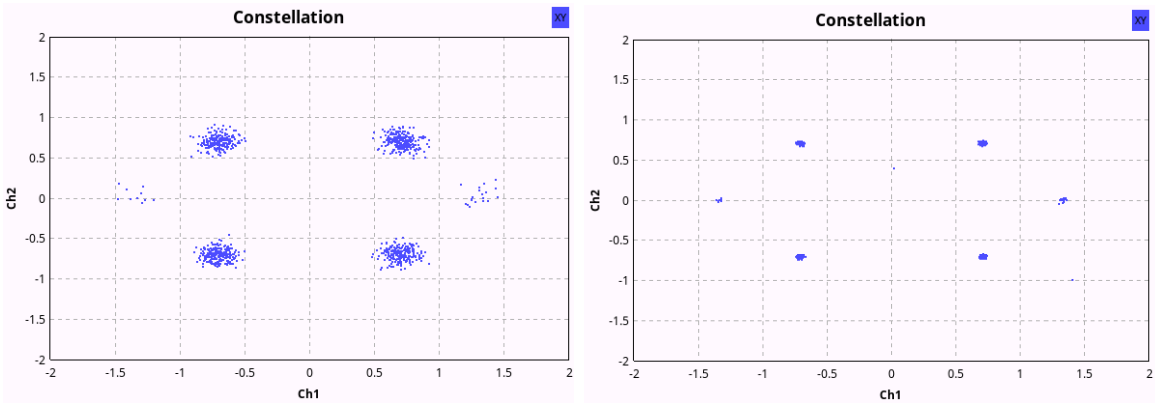
modulación. El comportamiento visto en las constelaciones es verificado con la medida del MER. Se puede observar en la tabla 8.1, que utilizando una frecuencia de transmisión de 60 MHz, los valores son similares, aunque el valor asociado al USRP B200 es levemente mayor. El resultado de esta prueba es de gran consideración,

	USB-VGA	USRP B200
MER(dB)	29,0	31,2

Tabla 8.1: Medidas de MER en transmisión one-seg a $60MHz$.

dado que refleja que al transmitir a frecuencias relativamente bajas, como lo es 60 MHz, y utilizando el ancho de banda de un solo segmento, la señal transmitida por el adaptador USB-VGA se comporta de manera similar que al utilizar el USRP B200. Siendo esto último sin considerar el efecto del canal y recibiendo con un RTL-SDR.

Se repitió la prueba transmitiendo a una frecuencia de 173 MHz, correspondiente al armónico 1 con el USB-VGA, con el objetivo de comparar el adaptador USB-VGA con el USRP B200 a frecuencias más altas. Los resultados obtenidos varían con respecto a utilizar una frecuencia de transmisión de 60 MHz. En la figura 8.2 puede notarse que la constelación de la señal transmitida por el USRP B200 se mantiene similar a la prueba anterior. En cambio la constelación transmitida por el USB-VGA se visualiza más ruidosa, tanto frente a la prueba con el adaptador a 60 MHz, como frente a la transmisión con el USRP B200 a 173 MHz. El hecho de transmitir con el adaptador en el armónico 1, hace que conformar con el $|P(f)|$ tenga más efecto sobre la señal, como fue explicado en el capítulo 6. La señal en este armónico se transmite con menor potencia que si esta fuera transmitida en el armónico 0. Esto se ve reflejado a su vez en las medidas de MER (tabla 8.2), en donde la medida del USB-VGA es bastante menor considerando la del USRP B200. También se puede notar en esta prueba que el MER relevado es mayor para



(a) Adaptador USB-VGA. (b) USRP B200.

Figura 8.2: Constelaciones transmitiendo a 173 MHz con distancias cercanas entre transmisor y receptor.

una transmisión a 173 MHz con el USRP B200 frente a una transmisión a 60 MHz con el mismo equipo. Esto se debe a que 60 MHz es una frecuencia que está al limite del rango de trabajo del USRP B200.

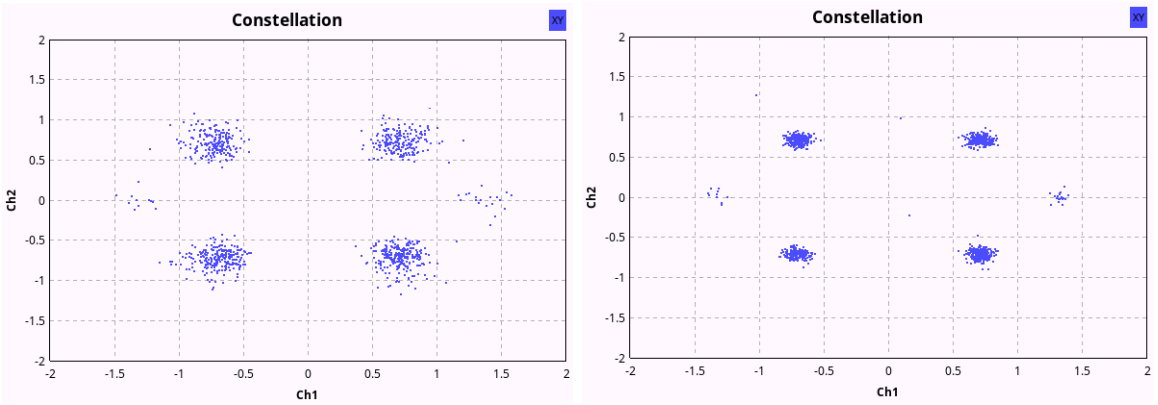
	USB-VGA	USRP B200
MER(dB)	19,3	34,5

Tabla 8.2: Medidas de MER en transmisión one-seg a $173MHz$.

Se prosiguió realizando pruebas de distancia para ambos equipos y verificando su funcionamiento. En este caso solo se utilizó la frecuencia de transmisión de 60 MHz. Para todas las distancias se relevó la medida del MER, reproduciendo el *transport stream* recibido en todos los casos para corroborar el correcto funcionamiento de ambos. Se puede observar en la tabla 8.3 que los valores de MER son similares entre los equipos, aunque un poco superiores en el USRP B200, en donde la mayor diferencia se encuentra a los 500 cm. En la figura 8.3 se muestra la constelación a distancia de 500 cm entre el transmisor y receptor. Se ve que a pesar de ser más bajo el MER que se obtiene utilizando el adaptador USB-VGA, la señal recibida es lo suficientemente buena como para lograr ser demodulada y el *transport stream* reproducido. Se entiende que la potencia de transmisión del adaptador USB-VGA es menor a la potencia de transmisión del USRP B200, lo cual genera diferencias entre los valores de MER relevados entre los equipos, que favorecen al USRP B200.

Distancia (cm)	60	120	240	500
MER USB-VGA(dB)	24,5	22,6	21,6	15,7
MER USRP(dB)	26,4	26,1	21,6	21,0

Tabla 8.3: Medidas de MER en transmisión one-seg para distintas distancias.



(a) Adaptador USB-VGA. (b) USRP B200.

Figura 8.3: Constelaciones transmitiendo a 60 MHz a una distancia de 500 cm entre transmisor y receptor.

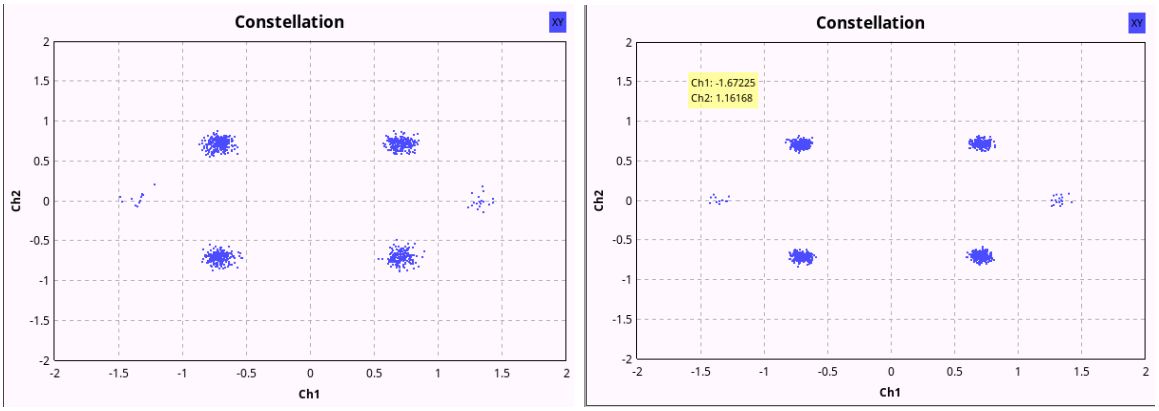
8.1.3. Transmisión full-seg

Para la comparación entre el USB-VGA y el USRP B200 transmitiendo la señal ISDB-T full-seg, se repitieron las pruebas del caso one-seg. Se comenzó estudiando los efectos de transmitir a una distancia muy cercana, con el fin de estudiar el comportamiento de la señal minimizando las pérdidas de propagación en el canal. Para esta prueba, se seleccionó la frecuencia de transmisión en 80 MHz, haciéndose que el armónico 0 de la señal transmitida por el adaptador USB-VGA se situara en dicha frecuencia. No se utilizó el amplificador en este caso, porque se quiere comparar solo el desempeño del adaptador USB-VGA. En la figura 8.4 se aprecian las constelaciones para ambos *hardware*. Al igual que para el caso one-seg, la señal transmitida por el USB-VGA se recibe con menor precisión en las zonas asociadas a los esquemas de modulación. La diferencia entre las constelaciones se reflejan en la medida del MER (tabla 8.4), en la cual se puede ver que la medida para el caso del adaptador USB-VGA es inferior que la del USRP B200.

	USB-VGA	USRP B200
MER(dB)	22,6	26,7

Tabla 8.4: Medidas de MER en transmisión one-seg a 80 MHz.

Se prosiguió aumentando la distancia entre el transmisor y el receptor, tomando para cada caso la medida del MER y la constelación. Además se reprodujo el *transport stream* recibido en cada caso, para constatar el correcto funcionamiento. En la tabla 8.5 se observan las medidas del MER frente a la distancia de los equipos. Se puede ver cómo a medida que aumenta la distancia, la diferencia entre ambas medidas es mayor. La potencia de transmisión del adaptador USB-VGA es más limitada que la del USRP B200, por lo que al aumentar el ancho de banda, la transmisión con el USB-VGA se ve afectada.



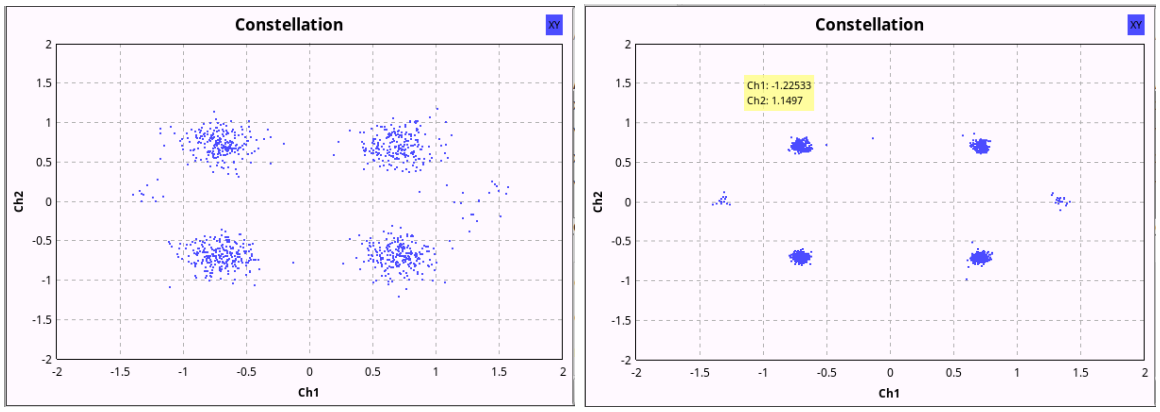
(a) Adaptador USB-VGA. (b) USRP B200.

Figura 8.4: Constelaciones transmitiendo a 80 MHz con distancias cercanas entre transmisor y receptor.

Distancia (cm)	60	120	240	500
MER USB-VGA(dB)	18,1	16,9	17,1	13,7
MER USRP(dB)	23,9	31,1	35,2	25,2

Tabla 8.5: Medidas de MER en transmisión full-seg para distintas distancias.

Con el estudio realizado en estas secciones, se puede concluir que los resultados obtenidos difieren entre el caso one-seg y el caso full-seg. En el caso one-seg se puede observar un comportamiento similar entre los equipos, con el USRP B200 obteniendo resultados ligeramente superiores. Esto siempre y cuando se utilice el armónico 0 para transmitir a través del adaptador USB-VGA. En el caso en el que se trabaje con armónicos más altos, el rendimiento del USB-VGA como transmisor decae considerablemente. Para el caso full-seg se pueden encontrar mayores diferencias en el rendimiento entre ambos. El mayor ancho de banda de la señal a transmitir perjudica al adaptador USB-VGA y su limitada potencia de transmisión. El caso de transmitir con el adaptador USB-VGA con un armónico mayor en full-seg no fue tratado explícitamente en esta sección, aunque sí se realizó una prueba que es expuesta en la sección siguiente. En esta se obtuvo un MER de 12 dB para una transmisión a 177,143 MHz, lo cual va en línea con lo obtenido sobre esta misma prueba en one-seg. Al trabajar en armónicos más altos la potencia de transmisión disminuye.



(a) Adaptador USB-VGA. (b) USRP B200.

Figura 8.5: Constelaciones transmitiendo full-seg a 80 MHz a una distancia de 500 cm entre transmisor y receptor.

8.2. Funcionamiento contra sintonizador ISDB-Tb comercial

Existe la posibilidad de transmitir las señales generadas por el transmisor ISDB-T en radio definida por *software*, hacia un sintonizador ISDB-Tb comercial. La transmisión se puede realizar tanto con transmisores SDR, como con el adaptador USB-VGA descripto a lo largo de la documentación. En la figura 8.6 se puede visualizar una transmisión utilizando el adaptador USB-VGA hacia un sintonizador ISDB-Tb integrado dentro de una televisión.

A la hora de recibir en un sintonizador comercial, a diferencia del *software* receptor implementado en GNU Radio, se cuentan con muchas menos herramientas para poder depurar los problemas que puedan generarse con la señal en recepción. Los sintonizadores comerciales tienden a ser plataformas cerradas, en las cuales no es posible ver el procesamiento interno que se le esté haciendo a la señal, ni qué errores están surgiendo al intentar recibirla. Además, a lo largo de las pruebas que se hicieron frente a distintos sintonizadores, fue posible entender que no hay un criterio unificado de cómo se realiza el procesamiento de la señal recibida. Algunos sintonizadores funcionan con distintos tipos de transmisiones, y no así otros, lo cual dificulta aun más la posibilidad de depurar los problemas en recepción.

Cabe destacar que para recibir en un sintonizador ISDB-Tb comercial, es necesario transmitir en las frecuencias designadas para este tipo de transmisión. El rango de frecuencias de los canales se ubica entre 177,143 MHz y 803,143 MHz (no todo ese rango completo está cubierto por canales exclusivamente de TV). Se probaron transmisiones tanto con el equipo USRP Ettus B200, como con el adaptador USB-VGA. Siempre se trabajó con el transmisor full-seg, ya que no fue posible con ningún sintonizador de los que se tenía disponibles, encontrar la señal si se transmite en un solo segmento.

El hecho de que la menor frecuencia de transmisión posible sea 177,143 MHz,

Figura 8.6: Transmisión ISDB-Tb hacia TV con USB-VGA.

implica según lo explicado en el capítulo 6, que al transmitir con el adaptador USB-VGA se utilice un armónico mayor al armónico 0 (el armónico que otorga la mayor potencia de transmisión). En particular, se utiliza el armónico 1 generado por el adaptador USB-VGA para transmitir a 177,143 MHz.

Para entender cómo podría ser el comportamiento de una transmisión full-seg con el adaptador USB-VGA en el armónico 1, se decidió realizar el siguiente experimento. Este consta del adaptador USB-VGA conectado al amplificador CommScope SVA15PRSMS (presentado en la sección 7.1) transmitiendo a 177,143 MHz en full-seg hacia el USRP Ettus B200, para observar la constelación, y medir el valor de MER a una distancia de 50 cm. El MER obtenido es de 12 dB, valor bajo para una recepción de este tipo, y la constelación se ve ruidosa en recepción, tal como se muestra en la figura 8.7. Esta prueba se realizó con las capas moduladas en QPSK, la modulación más robusta posible para este sistema.

De cualquier manera, al transmitir utilizando modulación QPSK con una tasa de código convolucional baja (transmisión robusta), en algunos sintonizadores comerciales fue posible recibir correctamente la transmisión. Trabajando a distancias cortas, se logró reproducir el *transport stream* enviado sin problemas. Los *transport streams* modulados en 64QAM, modulación que depende de una recepción con buena potencia, fueron imposibles de recibir.

Distinto fue el comportamiento utilizando el equipo Ettus USRP B200 como transmisor. Ya se había visto en la sección 8.1.3, que para la frecuencia de 80MHz en transmisión, los valores de MER, así como la constelación usando este equipamiento, son lo suficientemente buenos como para poder tener una recepción estable. Se puede entender que un comportamiento similar, o todavía mejor, se puede encontrar a una frecuencia de 177,143 MHz. Esto se puede justificar con la

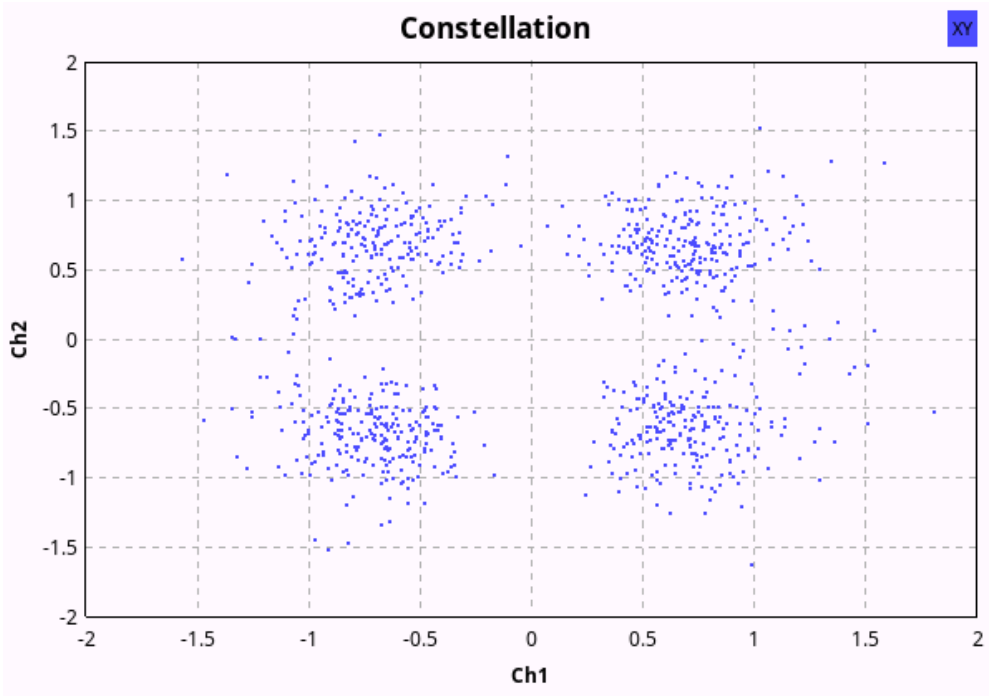


Figura 8.7: Transmisión ISDB-Tb entre USB-VGA y USRP en Full-Seg en 177,143MHz a 50cm de distancia.

frecuencia a la que esta trabajando el B200, siendo 80 MHz una frecuencia mucho más cercana al límite de su rango de trabajo, comparado a 177,143 MHz. Este buen comportamiento como transmisor se refleja a la hora de recibir en el sintonizador comercial, con ciertas consideraciones. Las transmisiones que parten de un BTS (*Broadcast Transport Stream*, definido en 3.3.1) que haya sido específicamente generado para ser transmitido por ISDB-Tb, son las que muestran mayor robustez a la hora de ser recibidos por muchos de los sintonizadores ISDB-Tb comerciales. Tanto transmisiones 64QAM, como el resto de las modulaciones, muestran buena estabilidad en la recepción. Situación similar ocurre con transmisiones de una sola capa en 13 segmentos, con un único TS (*Transport Stream*) transmitido que ocupa estos 13 segmentos. Ya con transmisiones en 2 o más capas, con TSs independientes introducidos en cada capa, las recepciones fueron más problemáticas, y mucho más dependientes del sintonizador que se esté utilizando. Algunos pocos sintonizadores mostraron una transmisión correcta y estable, pero la mayoría dieron problemas. Desde la imposibilidad de lograr encontrar correctamente más de un servicio a la vez con el sintonizador, hasta problemas con cortes continuos de audio e imagen, fueron parte de los inconvenientes de este tipo de transmisión.

Como evaluación primaria, se entiende que uno de los causantes de algunos de estos problemas es la falta de tablas PSI (*Program-specific information*) correctas en los TSs que se introducen al transmisor. La definición y utilización en profundidad de las tablas PSI se puede encontrar en la norma ISO/IEC 13818-1 [10]. En forma resumida, las tablas PSI son diversas tablas con metadatos sobre los

streams elementales que están contenidos dentro de los TSs. Estas tablas se encuentran dentro de los TSs, más precisamente dentro de TSPs (*Transport Stream Packets*) exclusivos para estas. Se entiende que una incorrecta implementación de estas tablas en los TSs genera inconvenientes para que el sintonizador logre encontrar los distintos servicios que se encuentran en cada capa. Una forma de poder solucionar este problema, sería utilizar un multiplexador que genere un BTS a partir de los distintos TSs, el cual inyecta las tablas necesarias para la correcta recepción. Por esta razón es también que las transmisiones en base a BTSs funcionan correctamente en este apartado. De cualquier manera, no se han encontrado soluciones de código abierto que permitan realizar esta funcionalidad. Respecto a lograr depurar el resto de los problemas que ocurren en recepción, como ser la gran variabilidad entre la estabilidad de recepción en distintos sintonizadores ISDB-Tb, es un trabajo que se dejará para continuar desarrollando a futuro.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 9

Placa Raspberry Pi como transmisor SDR

9.1. Introducción

9.1.1. Raspberry Pi

Las Raspberry Pi son una serie de minicomputadoras de placa única desarrolladas por la fundación Raspberry Pi [35], cuyo objetivo principal es promover la educación en ciencias de la computación en países en desarrollo. Su naturaleza de código abierto, bajo costo, y su creciente comunidad de desarrollo, hacen de la Raspberry Pi un equipo interesante para todo tipo de proyectos relacionados con la computación y la electrónica. Puede ser utilizado tanto como computadora personal, como para controlar motores DC, siendo estos ejemplos clásicos de su uso. En el transcurso del proyecto, se trabajó con un modelo en particular de la placa, la Raspberry Pi 3B+. Esta puede observarse en la figura 9.1 y cuenta con las siguientes especificaciones:

- Procesador: Broadcom BCM2837B0 1,4GHz
- Memoria RAM: 1GB LPDDR2 SDRAM
- Conectividad inalámbrica: IEEE 802.11.b/g/n/ac, Bluetooth 4.2
- Conectividad de Red: Gigabit Ethernet over USB 2.0
- Puertos: GPIO 40 pines, HDMI, 4x USB 2.0, Micro SD, Micro USB, CSI, DSI

Cabe destacar la existencia de los pines GPIO (*General Purpose Input/Output*), pines de entrada y salida en la placa, que pueden ser controlados por el usuario en tiempo de ejecución. Estos pines van a permitir el uso de la placa como transmisor SDR.

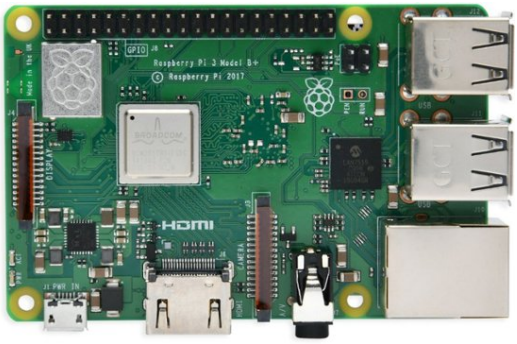


Figura 9.1: Raspberry Pi 3B+.

9.1.2. Proyectos RpiTX y RpiDATV

Los proyectos RpiTX [8] y RpiDATV [9] son dos proyectos independientes creados por Evariste Courjaud. Ambos comparten el objetivo de lograr transmitir distintos tipos de modulaciones sobre radiofrecuencia a través de la Raspberry Pi. Es posible descargarlos a través de Github, donde se encuentra el código fuente a disposición (son proyectos de código abierto). RpiTX brinda a la placa la capacidad de transmitir señales con variedad de modulaciones, en un rango de frecuencia entre 5 kHz y 1500 MHz. Mediante este proyecto, es posible transmitir señales FM, SSB, entre otros. RpiDATV por otro lado, permite realizar transmisión de señales que cumplen con el estándar DVB-S [36]. DVB-S es un estándar europeo para transmisión de televisión satelital, el cual utiliza una modulación QPSK para enviar los datos. Por lo tanto, con el proyecto RpiDATV es posible modular y enviar señales DVB-S a través de uno de los pines GPIO. Además el proyecto incluye un re-codificador capaz de generar *transport streams* compatibles con la norma. Ambos proyectos funcionan en una amplia gama de modelos de Raspberry Pi, desde el modelo Zero, hasta el modelo 3B+. Cabe destacar que el proyecto RpiTX está basado en la librería Librpitx [37], la cual proporciona las principales funciones necesarias para el proyecto.

El ecosistema de *software* y *hardware* abierto en el que se basa Raspberry Pi es el principal impulsor de este tipo de desarrollos. La disponibilidad de documentación que describe el *hardware* disponible en la placa, es la base de ambas implementaciones. En particular, la hoja de datos que describe el funcionamiento entre el procesador de la placa y sus distintos periféricos ocupa un lugar preponderante en el desarrollo de estos proyectos [38].

Como fue explicado anteriormente, ambos proyectos hacen uso de pines pertenecientes al GPIO. Es posible adaptar una antena transmisora para conectarla al pin correspondiente, y así mejorar la ganancia y direccionalidad de la transmisión en el caso en el que se requiera. Cada pin en particular puede cumplir diferentes funcionalidades, y los distintos tipos de modulaciones que se pueden lograr con estos dependen de cuál se esté utilizando. Son dos pines distintos los utilizados como transmisores en los proyectos, uno en RpiDATV y uno en RpiTX (esto se explicará con mayor detalle posteriormente). En la figura 9.2 pueden observarse

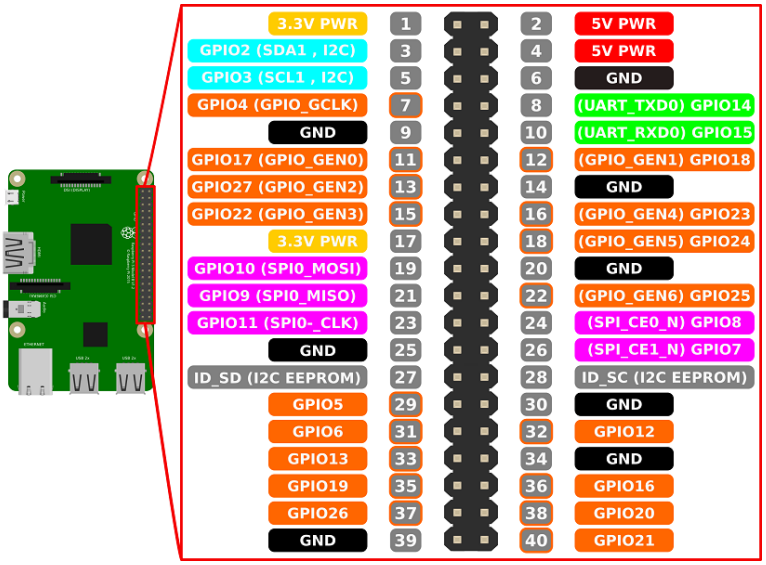


Figura 9.2: Pines GPIO en Raspberry Pi 3B+ [39].

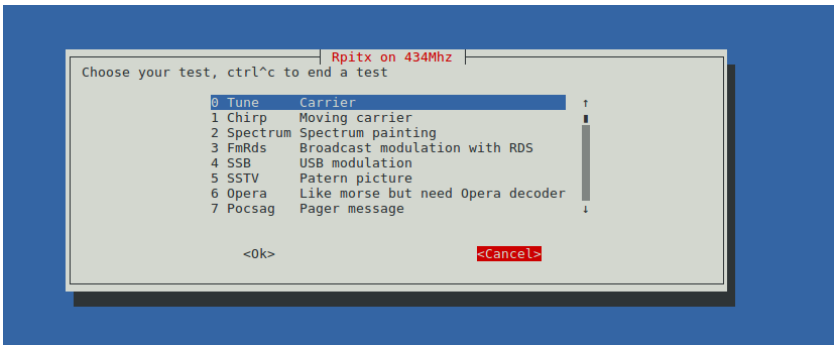


Figura 9.3: Menú en consola en proyecto RpiTX [8].

los distintos pines GPIO que contiene la placa Raspberry Pi 3B+.

Para realizar una transmisión utilizando estos proyectos, es necesario ingresarle a la placa Raspberry Pi los datos que serán transmitidos. Existen distintos métodos para lograr esto. Tanto utilizando un archivo almacenado en la placa (o en algún periférico USB que se le conecte) con los datos a enviar, como a través de una conexión TCP/IP entre una computadora externa que genere los datos, y la placa que la recibirá y transmitirá al aire. Para algunos tipos de modulación, como ser FM o SSB en RpiTX, o DVB-S en RpiDATV, los proyectos brindan un menú por consola para realizar transmisiones básicas en forma simple, tal como se visualiza en la figura 9.3. Cabe destacar que la tasa de muestreo máxima con la que la Raspberry Pi puede trabajar se define en 250 kHz según el código fuente del proyecto RpiTX [8].

9.2. Modulación en RpiTX y RpiDATV

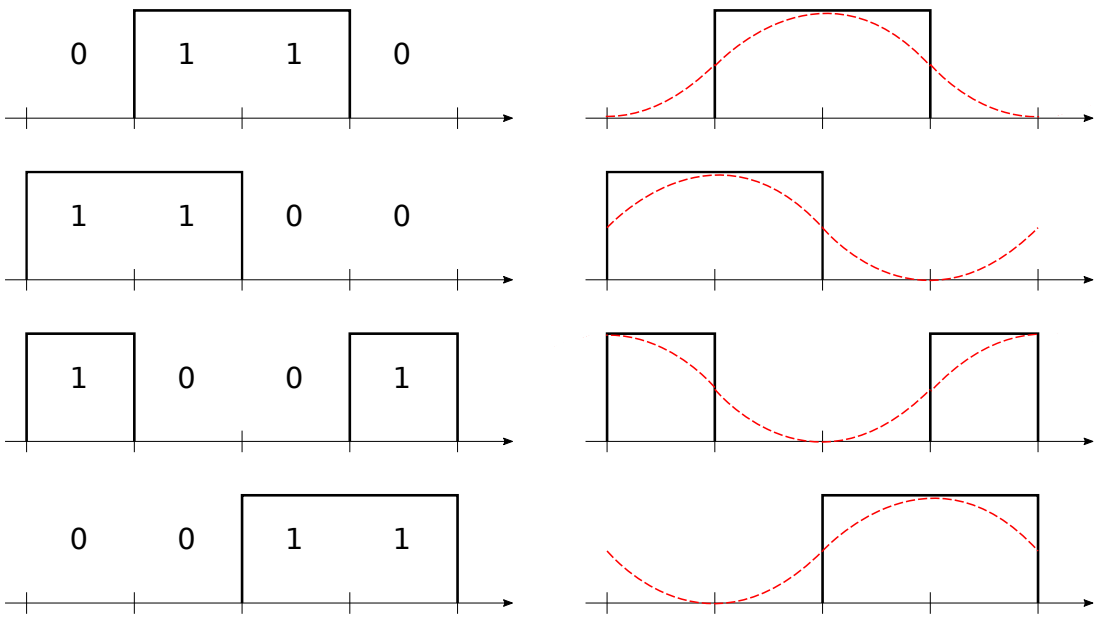
Por la naturaleza de la Raspberry Pi, los pines GPIO cumplen en general el propósito de enviar niveles continuos de voltaje a través de ellos. El control de diversos equipos a través de la placa, como ser motores o LEDs por ejemplo, requieren de este tipo de funcionalidad. También se brinda la posibilidad de enviar señales de reloj a otros componentes a través de los pines. Por consiguiente, los pines GPIO tienen la capacidad de transmitir tanto pulsos y ondas cuadradas.

Los pulsos derivan del oscilador principal de la placa, y el uso de distintos PLLs (*Phase Locked Loop*) integrados en esta. Las ondas cuadradas van a ser la base de las transmisiones a través de la placa, y serán moduladas tal como si se trabajara con una onda sinusoidal. El hecho de que la onda cuadrada contenga una componente espectral en su frecuencia fundamental, permite utilizarla de forma equivalente a la onda sinusoidal. De cualquier manera, es necesario recalcar la cantidad de armónicos que se generan a diferencia de la onda sinusoidal. Este comportamiento es el mismo que sucede con el adaptador USB-VGA, y su explicación se puede encontrar en el capítulo 6. También al igual que con el adaptador USB-VGA, es posible aprovechar los armónicos para transmitir a frecuencias más altas que la del armónico fundamental, a costa de una menor potencia de transmisión. A continuación, se explicarán las distintas modulaciones que se pueden lograr con estos proyectos, y finalmente se estudiará la forma en la que se utilizarán algunas de estas modulaciones para intentar lograr una transmisión ISDB-T con la placa.

9.2.1. RpiDATV

Como se explicó anteriormente, RpiDATV se centra en la transmisión de señales bajo la norma DVB-S. Esto implica la necesidad de aplicar una modulación en fase a la onda portadora para lograr un esquema del tipo QPSK. Para lograr este tipo de modulación, se utiliza uno de los periféricos disponibles en la placa Raspberry Pi, el periférico PWM (*Pulse-Width Modulation*), aunque no con el fin de utilizar dicha modulación. El periférico PWM posee un modo serializador, en el que se van tomando bits en forma serial desde algún *buffer* o registro, y son enviados uno por uno hacia el pin como niveles de voltaje alto y bajo. Estos bits son tomados con cierta frecuencia regulable, lo cual análogamente genera pulsos de ancho regulable que podrán ser transmitidos a través del pin.

La modulación en fase a través de este sistema, se basa en mantener al serializador con una frecuencia constante, y agrupar pulsos de niveles altos y bajos para lograr generar una señal de onda cuadrada con sus diferentes rotaciones de fase. Para enviar una señal QPSK, es necesario poder definir 4 símbolos, siendo estos la onda cuadrada rotada en 4 fases equiespaciadas. Cada símbolo QPSK se genera agrupando de forma particular 4 pulsos entregados por el serializador. La forma de agruparlos se puede visualizar en la figura 9.4. También en esa figura se puede observar a cada uno de los símbolos junto a su armónico fundamental, mostrando como estos 4 símbolos también equivalen a 4 rotaciones de fase de una onda sinusoidal. Analizando la cadencia con la que el serializador envía los pulsos, es posible encontrar la frecuencia a la que se encontrará la transmisión. Sea f_p la



(a) Símbolos QPSK. (b) Símbolos QPSK y su armónico fundamental.

Figura 9.4: Símbolos QPSK.

frecuencia de generación de pulsos, se puede observar que con 4 pulsos se genera 1 período de la onda cuadrada. Esto implica que la frecuencia de la onda cuadrada (y por consiguiente su armónico fundamental) valdrá $f_{onda} = \frac{f_p}{4}$. También es posible encontrar la transmisión en los distintos armónicos superiores que son generados por la onda cuadrada. El pin que permite enviar la señal obtenida del periférico PWM es el GPIO 12.

El proyecto RpiDATV presenta una forma de transmitir señales utilizando modulación de fase. Sin embargo, este no contiene modos que puedan ser útiles para enviar una señal bajo la norma ISDB-T. Por tal motivo, se estudiará en la siguiente sección el proyecto RpiTX, el cual presenta un modo que potencialmente puede permitir la transmisión ISDB-T.

9.2.2. RpiTX

RpiTX basa su funcionamiento en generar una señal de reloj, y permitir tanto variaciones en la frecuencia, como variaciones en la amplitud de esta. Esto le otorga flexibilidad al proyecto en cuanto a la variedad de esquemas de modulación que es posible transmitir. La forma en la que se genera la señal de reloj, y se realizan en

esta variaciones de frecuencia y amplitud se explicará a continuación.

Generación de la señal de reloj y variación de su frecuencia

La señal de reloj parte de un resonador de cristal de la placa que trabaja generando oscilaciones a 19,2 MHz. Este oscilador puede ser enrutado hacia distintos PLLs que funcionan como multiplicadores de la frecuencia original. El SoC (*System on a chip*) de la Raspberry Pi contiene 5 PLLs independientes que se utilizan para aportar la señal de reloj a distintos componentes de la placa. Los multiplicadores de estos 5 PLLs pueden ser configurables para lograr variedades de frecuencias posibles. La señal de estos PLLs puede ser enrutada hacia los denominados "APER", bloques divisores de frecuencia, los cuales trabajan con valores enteros pequeños. Estos realizan una división "gruesa" de la frecuencia del PLL. Estos luego son enrutados a los "periféricos GPCLK" (*General Purpose Clock*), periféricos encargados de posibilitar el envío de estas señales de reloj al exterior a través de pines GPIO. Los periféricos GPCLK tienen la capacidad de dividir la frecuencia de reloj recibido por la suma entre un valor racional y un valor entero, y así realizar una división mucho más fina de la señal [40]. Esto se puede visualizar en la siguiente expresión:

$$f_{GPCLK} = \frac{f_{source}}{DIVI + \frac{DIVF}{4096}}. \tag{9.1}$$

Este divisor puede ser modificado en tiempo real, permitiendo variar la frecuencia con distintos valores discretos, logrando una buena resolución gracias al divisor racional. Además, como funcionalidad extra, existe la posibilidad de variar rápida y alternadamente el divisor racional entre valores adyacentes. Esto permite trabajar con valores de frecuencia intermedios a estos, pudiendo lograr una resolución del orden del micro-Hz. El diagrama de flujo que muestra la generación de la señal de GPCLK se puede visualizar en la figura 9.5. Para utilizar la señal de reloj, es necesario trabajar con pines que estén interconectados a los periféricos GPCLK. En el proyecto RpiTX se utiliza el pin GPIO 4, pin capaz de enviar la señal GPCLK.

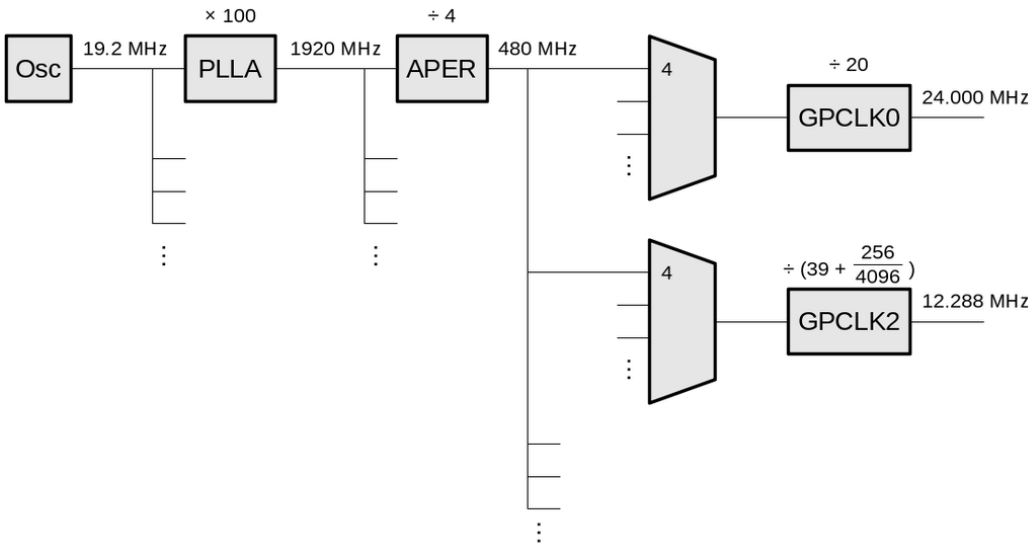


Figura 9.5: Diagrama "Clock Tree" [40].

Variación de amplitud

A la señal de reloj descripta en la sección anterior, es posible realizarle variaciones en amplitud. Puntualmente, es posible variar la intensidad de corriente que se entrega al pin en tiempo real. Un grupo de *buffers* triestado en paralelo son los que definen la intensidad de corriente que se entrega. Estos pueden ser activados individualmente para brindar hasta 8 niveles distintos de corriente. Además, es posible colocar al pin en un nivel bajo de voltaje ("apagarlo"). En conjunto, esto se traduce en 9 niveles posibles de variación de amplitud de la señal. Un diagrama del funcionamiento del conjunto de *buffers* que definen la corriente se puede visualizar en la figura 9.6. Estos *buffers* son controlados por un registro de 3 bits en la placa. Según el valor del registro, se entregan al pin los valores de corriente que se pueden visualizar en la tabla 9.1.

Valor	Corriente
000	2 mA
001	4 mA
010	6 mA
011	8 mA
100	10 mA
101	12 mA
110	14 mA
111	16 mA

Tabla 9.1: Corriente entregada según valor del registro.

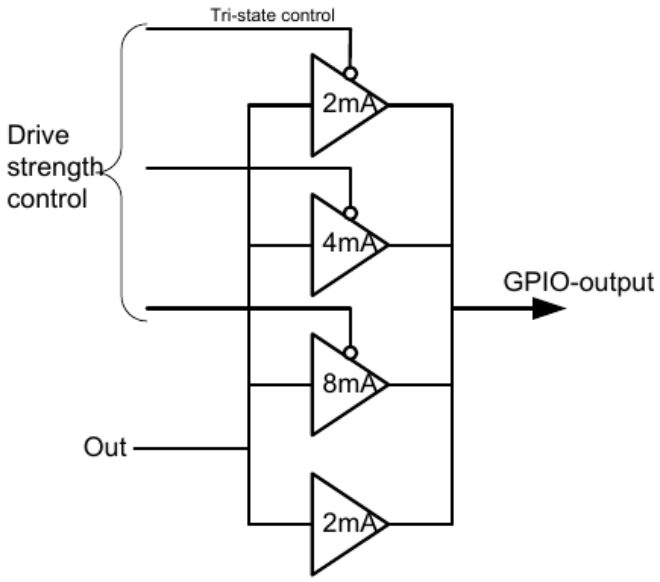


Figura 9.6: *Buffers* triestado que regulan corriente en GPIO [41].

9.3. Transmisiones en RpiTX a través de datos IQ

Una de las funcionalidades más útiles de RpiTX, y necesaria para una potencial transmisión ISDB-T, es la posibilidad de realizar una transmisión en base a datos IQ. El *software* del transmisor ISDB-T en GNU Radio entrega en su salida datos de este tipo. Varios esquemas de modulación implementables en GNU Radio pueden ser representados en datos IQ, como por ejemplo AM, FM, entre otros. Esto permite tener flexibilidad en cuanto a los esquemas de modulación que la placa puede transmitir. Este tipo de datos representa tanto a la parte real como a la parte imaginaria de cada complejo, como un dato de tipo flotante de 32 bits cada parte. Estos flotantes son ordenados en serie, colocando primero la parte real de un complejo (dato "I"), luego la parte imaginaria del complejo (dato "Q"), luego la parte real del siguiente complejo, y así sucesivamente. Estos datos ordenados pueden ser almacenados en un archivo y posteriormente levantados por RpiTX, o pueden ser enviados en un flujo constante a través de una conexión TCP/IP entre una computadora y la Raspberry Pi, para ser luego tomados por RpiTX para su transmisión.

La forma en la que RpiTX transmite estos complejos, es enviar a través del pin una señal de reloj cuya amplitud representa el módulo de cada valor complejo modulado con los 9 niveles disponibles. Y la frecuencia de la misma es igual a la suma de la frecuencia instantánea de los complejos con la frecuencia a la cuál se desea transmitir. Este proceso puede observarse paso por paso en uno de los archivos fuente de la librería Librpitx [37]. El archivo en cuestión es `iqdmasync.cpp` y para comprender su funcionamiento, las sentencias más importantes de este código serán explicadas a continuación.

Código 9.1: Variación de frecuencia de la señal de reloj.

```
{sampletab[Index*registerbysample]=(0x5A<<24)|GetMasterFrac(mydsp.
    frequency);}
```

En el código 9.1, la variable `mydsp.frequency` está definida por el valor de la frecuencia instantánea entre dos complejos consecutivos. Para realizar este cálculo, se toman dos complejos consecutivos (dos pares de datos IQ), se calcula la fase de cada complejo, y se obtiene la frecuencia instantánea como derivada discreta de la fase según la ecuación: $f = \frac{(phase2-phase1).S}{2\pi}$ donde S es el valor de la tasa de muestreo asociada a los datos IQ. Lo que realiza el código 9.1 es definir para cada par de complejos que se procesa, la frecuencia instantánea que la señal de reloj (onda cuadrada que se transmite al aire) debe tener. La función `GetMasterFrac` permite a partir de la frecuencia instantánea entre los dos complejos, y de la frecuencia central de transmisión que se define al correr el proyecto RpiTX, calcular los valores de DIVI y DIVF que se muestran en la ecuación 9.1. Los valores de DIVI y DIVF son los que deben ser escritos en un registro de la Raspberry Pi para lograr que la señal de reloj tenga la frecuencia instantánea f_{GPCLK} deseada según la ecuación 9.1. El registro a editar es un registro de 32 bits en el que en los primeros 8 bits se debe introducir el valor hexadecimal 0x5A para permitir la edición del resto del registro.

Código 9.2: Variación de amplitud de la señal de reloj.

```
int IntAmplitude=(int)(mydsp.amplitude*8.0)-1;
int IntAmplitudePAD=IntAmplitude;
if(IntAmplitude<0) {IntAmplitudePAD=0;IntAmplitude=-1;}
sampletab[Index*registerbysample+1]=(0x5A<<24) + (IntAmplitudePAD&0
    x7) + (1<<4) + (0<<3);
if(IntAmplitude==-1)
               {
                       sampletab[Index*registerbysample+2]=(
                           Originfsel & ~(7 << 12)) | (0 << 12);
               }
```

En el código 9.2, la variable `mydsp.amplitude` contiene el valor del módulo del complejo a transmitir. Cabe aclarar que los datos IQ que se ingresan, deben tener en su amplitud valores entre -1 y 1, lo cual equivale a un módulo menor o igual a 1 para cada valor complejo. El valor del módulo es multiplicado por 8, este es restado por 1, y el valor entero de esta operación es guardada en la variable `IntAmplitude`. Lo que se obtiene en esta variable, son 9 posibles valores enteros con un rango entre -1 y 7, lo cual equivale a una cuantización del módulo del complejo en 9 niveles. Estos se utilizan para variar la amplitud de la señal de reloj, tal como se explicó en la sección 9.2.2. El nivel -1 de `IntAmplitude` equivale a un módulo del complejo igual a 0, lo cual corresponde a apagar la señal de reloj que se envía a través del pin. Esta acción se observa en el ultimo `if` dentro del código 9.2 en el que se procede a modificar uno de los registros de la Raspberry Pi para ordenar el apagado de la señal de reloj. Para el resto de los valores de

`IntAmplitude`, la intensidad de corriente que se otorga en la señal de reloj se define en la cuarta sentencia de 9.2, en la que se escribe en un registro de la placa el valor entre 0 y 7 (en binario) que definirá la corriente según la tabla 9.1. En el próximo capítulo se aplicarán estos conceptos para llevar a cabo transmisiones con distintas modulaciones a través de RpiTX.

Capítulo 10

Pruebas de Raspberry Pi como transmisor SDR

En base a lo explicado en el capítulo 9, se mostrará el comportamiento de la placa Raspberry Pi como transmisor. Para cualquiera de estas transmisiones, es necesaria la instalación del proyecto RpiTX en una Raspberry Pi corriendo el sistema operativo Raspbian (distribución de GNU/Linux basada en Debian) [42]. En las siguientes pruebas que se expondrán, se utilizó una Raspberry Pi 3 B+, aunque hay una gran variedad de placas Raspberry Pi compatibles con el proyecto RpiTX. Una lista de todos los modelos compatibles se encuentra en la web del proyecto RpiTX [8].

El pin GPIO 4 es el que actuando como antena, transmite las señales al aire. Es posible conectar una antena a este pin para lograr mejor ganancia y directividad. Para las pruebas expuestas en este capítulo, se utilizó un conector de 1 pin hembra-hembra, en el que una de las terminales está conectada al pin GPIO 4, y la otra terminal hacia una antena monopolo de $\frac{1}{4}$ de longitud de onda. Esta configuración se puede observar en la figura 10.1. Se expondrán pruebas de transmisión FM, AM e ISDB-T, todas partiendo de datos IQ generados desde una computadora. El propósito de realizar pruebas AM y FM tiene que ver con lo expuesto en la sección 9.2.2. La señal de reloj generada por la placa puede ser variada tanto en amplitud como en su frecuencia instantánea, y las modulaciones AM y FM permiten ver por separado el rendimiento de cada una de estas funcionalidades. AM dependerá de las variaciones de amplitud, y de qué tanto afecte los pocos niveles disponibles. FM dependerá de las variaciones de frecuencia instantánea, y de la resolución disponible para definirla.

10.1. Transmisión FM

Para realizar una transmisión FM, se utiliza una computadora que genera datos IQ a través de GNU Radio para posteriormente ser enviados a la Raspberry Pi y transmitidos al aire por esta. No es completamente necesario este tipo de configuración para una transmisión FM con una Raspberry Pi. La generación de

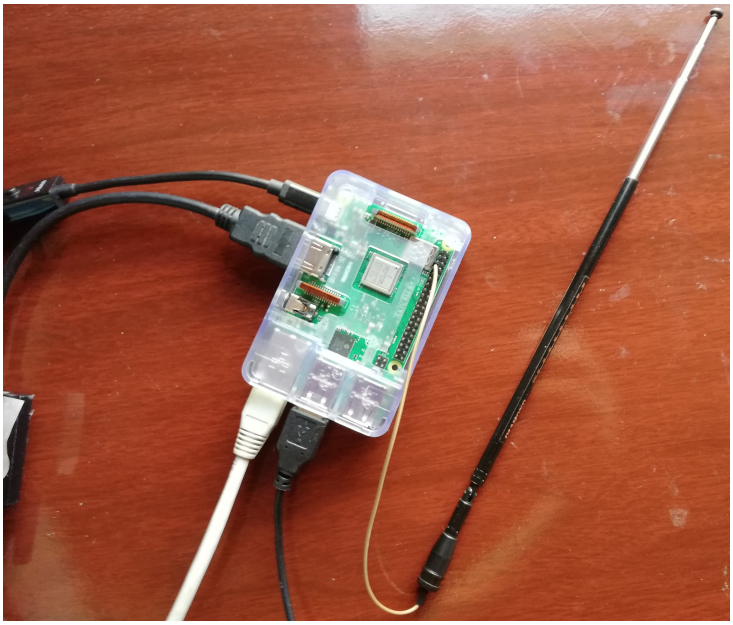


Figura 10.1: Configuración de Raspberry Pi para transmisión.

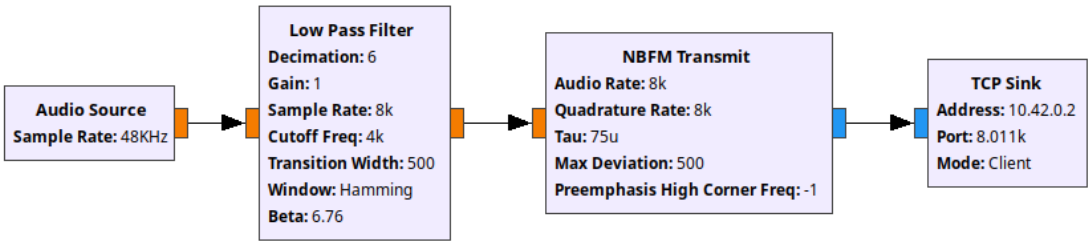


Figura 10.2: Transmisor FM de banda angosta en GNU Radio.

los datos a transmitir, tanto como la transmisión en sí, son posibles de hacerse enteramente dentro de la placa sin la participación externa de una computadora que esté generando los datos. De cualquier manera, para la transmisión ISDB-T presentada posteriormente en este capítulo, sí es necesaria la interconexión de datos entre computadora y Raspberry Pi, y esta configuración es una buena forma de probar este funcionamiento. La placa Raspberry Pi por si sola no posee la capacidad de procesamiento para poder ejecutar internamente el software transmisor ISDB-T y luego transmitir al aire.

Para realizar una transmisión FM, se elabora un *flowgraph* transmisor de FM en GNU Radio. El diagrama del transmisor FM en GNU Radio se puede visualizar en la figura 10.2. Este transmisor recibirá el audio del micrófono de la computadora con el bloque "Audio Source" a una tasa de muestreo de 48 kHz, cuya señal será filtrada a 4 kHz por un filtro pasa bajos y la tasa de muestras disminuida a un valor de 8 kHz. Los datos serán luego modulados en FM de banda angosta con el bloque "NBFM Transmit". El envío de datos desde la computadora que está

corriendo el transmisor en GNU Radio hacia la Raspberry Pi se hace a través de un cable de red conectado entre ambas. Se utiliza el protocolo de transporte TCP para la comunicación. En el transmisor en GNU Radio se puede observar el bloque "TCP Sink" como bloque final del flujo. A este bloque se le define el puerto TCP que se utilizará en el envío, así como la IP destino que en este caso es la IP que la Raspberry Pi tiene asignada. A su vez en una terminal de la Raspberry Pi se usa el siguiente comando para realizar la transmisión al aire:

```
nc -l 8011 | sudo rpitx -i- -m IQFLOAT -s 8000 -f 89300
```

El comando se describe de la siguiente forma:

- `nc -l 8011 |` : Se utiliza la herramienta `netcat` (`nc`) para escuchar el puerto TCP 8011 que se utilizó en este ejemplo. Sobre este puerto llegan a la placa los datos generados por el transmisor FM en GNU Radio. Este puerto debe coincidir con el definido en el bloque "TCP Sink" en el *software* transmisor. A su vez, la utilización del símbolo *pipe* (`|`), permite generar una tubería en el que los datos recibidos por TCP, son ingresados al *software* del proyecto RpiTX.
- `sudo rpitx -i- -m IQFLOAT -s 8000 -f 89300`: Se ejecuta la función `rpitx` perteneciente al proyecto RpiTX para realizar la transmisión. Se utiliza la opción `-i-` para recibir los datos desde la salida de la función `netcat` a través del *pipe*. La opción `-m` define el tipo de dato que la función `rpitx` espera. En este caso es `IQFLOAT`, tipo de dato ya explicado en la sección 9.3. La opción `-s` define la tasa de muestreo que se utilizará para la transmisión, siendo esta de 8 kHz. Por último, la opción `-f` define la frecuencia a la que se realizará la transmisión en kHz, siendo en este caso una frecuencia de 89,3 MHz.

Con este comando, se inicia la transmisión. Como fue explicado en la sección 9.3, la placa calcula el módulo de los complejos, el cual en FM corresponde a una constante y la modula con los niveles de amplitud disponibles. La frecuencia instantánea de los complejos, correspondiente al mensaje en dicha modulación, es sumada a la frecuencia a la cuál se desea enviar.

La transmisión es recibida en una computadora utilizando como receptor un equipo SDR Ettus B200 con una antena monopolo de $\frac{1}{4}$ de longitud de onda. En la figura 10.3 se observa el espectro recibido. En la figura 10.4 se observa la constelación recibida, en la que se nota el módulo constante de los complejos junto a la rotación continua de fase. La señal recibida es correctamente demodulada, y se puede escuchar el audio enviado con buena calidad. También al estar utilizando una frecuencia de FM comercial, es posible escuchar la transmisión desde cualquier receptor FM estándar. Con esta prueba se confirma que las variaciones en la frecuencia instantánea de la señal de reloj, tienen una resolución lo suficientemente buena como para lograr una transmisión de este tipo.

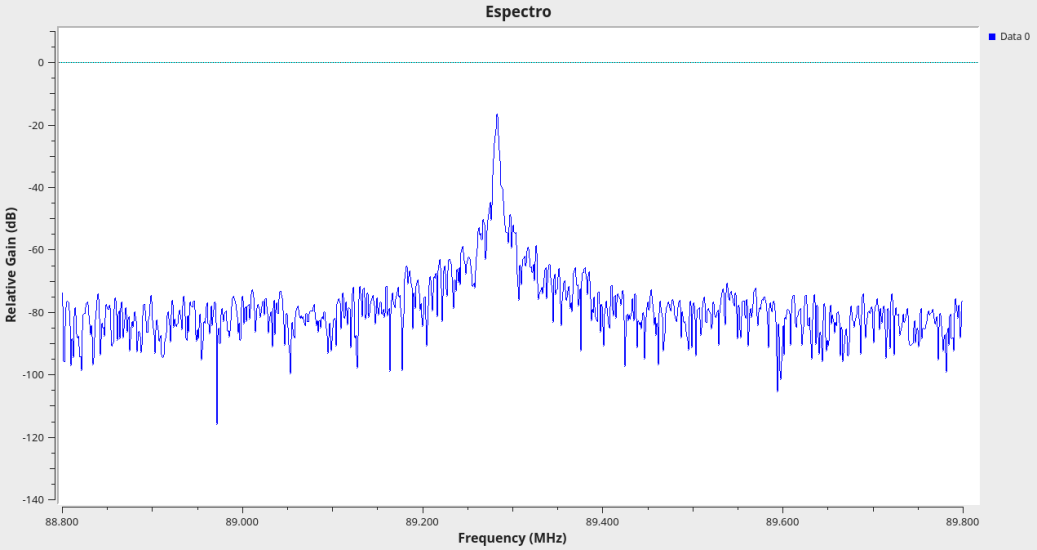


Figura 10.3: Espectro recibido de transmisión FM con Raspberry Pi.

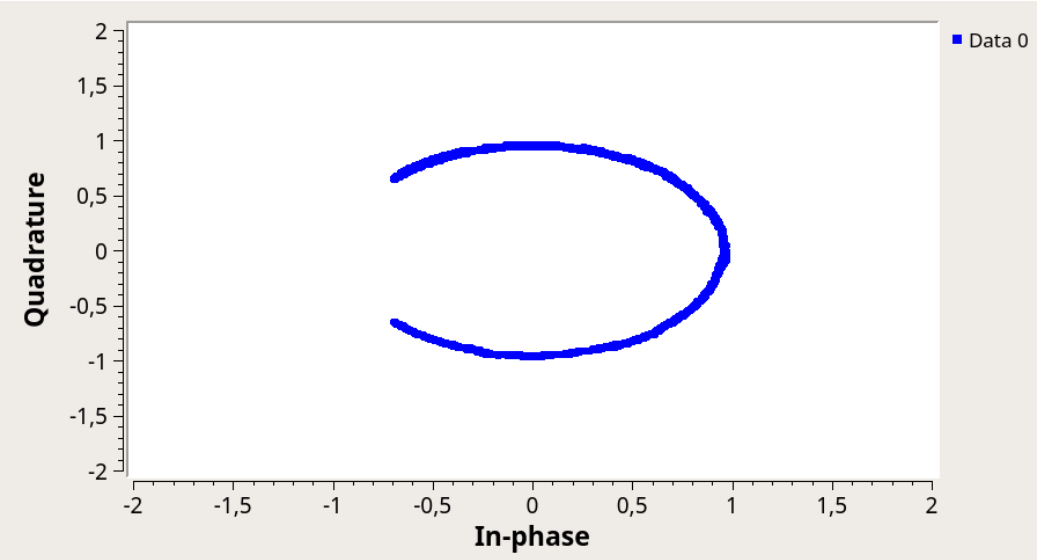


Figura 10.4: Constelación recibida de transmisión FM con Raspberry Pi.

10.2. Transmisión AM

En las mismas condiciones que la prueba de transmisión FM (utilizando una computadora junto a la Raspberry Pi, y trabajando con datos IQ), se intenta transmitir AM con la placa Raspberry Pi. El transmisor AM utilizado en la computadora, que enviará las muestras a la Raspberry Pi, se puede visualizar en la figura 10.5. En esta modulación el módulo de los complejos tendrá la información del mensaje a transmitir, el cuál será representado con los 9 niveles disponibles. La frecuencia instantánea que tienen los complejos en este caso es constante.

El comando utilizado para la transmisión en la Raspberry Pi es el siguiente:

```
nc -l 8011 | sudo rpitx -i- -m IQFLOAT -s 8000 -f 28000
```

Esta modulación se pudo recibir correctamente con un receptor RTL-SDR, logrando observar el espectro que se ve en la figura 10.6. Igualmente, al realizar la demodulación de la señal se pudo percibir que la calidad de audio es sumamente mala. Este fenómeno se puede adjudicar a la baja resolución disponible para la modulación en amplitud de la señal de reloj. En una transmisión AM, los únicos 9 niveles de amplitud posibles para utilizar en este tipo de modulación con la Raspberry Pi, van a causar un efecto indeseado de ruido de cuantización. De cualquier manera, también podría existir algún otro efecto o problema que esté aportando al mal funcionamiento de esta modulación en la placa.

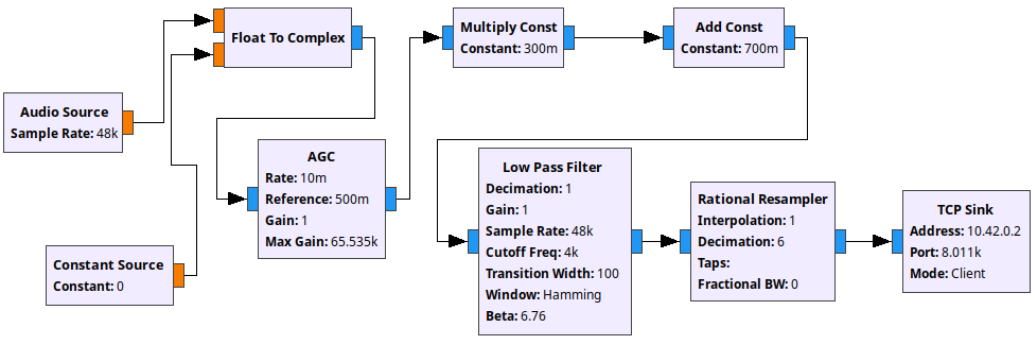


Figura 10.5: Transmisor AM en GNU Radio.

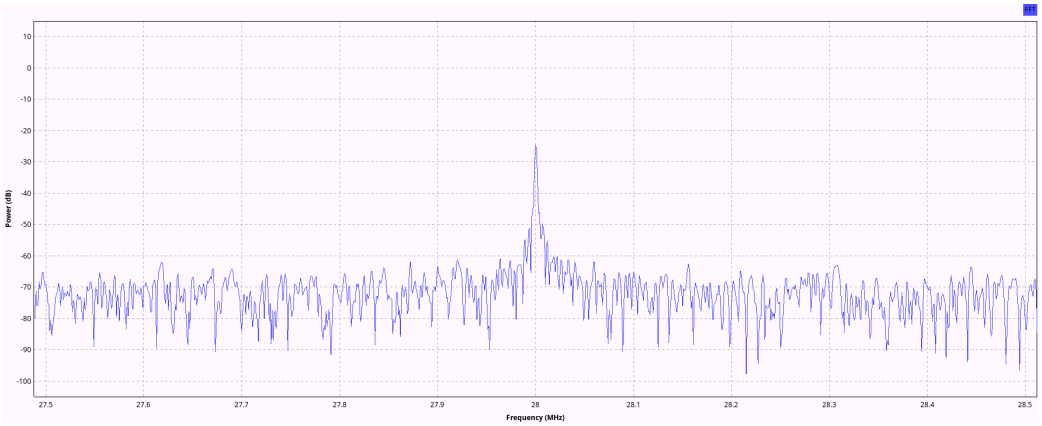


Figura 10.6: Espectro recibido de transmisión AM con Raspberry Pi.

10.3. Transmisión ISDB-T

En esta sección, se intentará implementar una transmisión bajo la norma ISDB-T a través de la placa Raspberry Pi con el proyecto RpiTX. El sistema completo se compone del *software* transmisor ISDB-T one-seg enviando los datos IQ desde una computadora a la placa Raspberry Pi para ser transmitidos, y de un receptor SDR conectado a una computadora recibiendo esa transmisión. La metodología del envío de datos desde la computadora con el *software* transmisor ISDB-T en GNU Radio hacia la Raspberry Pi, es similar a la transmisión FM y AM. El flujo en GNU Radio del transmisor ISDB-T finaliza en un bloque "TCP Sink", que es el encargado de enviar los datos hacia la Raspberry Pi con la IP y puerto que se definan. Se utiliza al transmisor ISDB-T one-seg debido a lo ya explicado en la sección 4.2.1, este transmisor permite aprovechar en mejor manera la potencia limitada de transmisión de la Raspberry Pi.

La transmisión de una señal OFDM a través de la placa se puede ver como una combinación de las transmisiones realizadas anteriormente (AM y FM), dado que los complejos varían tanto en amplitud, como en frecuencia. Es importante aclarar que la tasa de muestreo para una transmisión ISDB-T one-seg (507,937 kHz) definida en el capítulo 4 es demasiado alta frente al máximo de 250 kHz que RpiTX soporta. De cualquier manera, se decidió para una primera prueba eliminar ese límite del código fuente del proyecto, e intentar transmitir con una tasa de muestreo de 507,937 kHz.

El comando utilizado para la transmisión ISDB-T es:

```
nc -l 8011 | sudo rpitx -i- -m IQFLOAT -s 507937 -f 80000
```

- `-f 80000`: Se decidió usar una frecuencia de 80 MHz para esta prueba. La Raspberry Pi no sufre de los mismos fenómenos que suceden con el adaptador USB-VGA y los armónicos en este modo de transmisión, por lo que la frecuencia de transmisión es de libre elección siempre y cuando el espectro

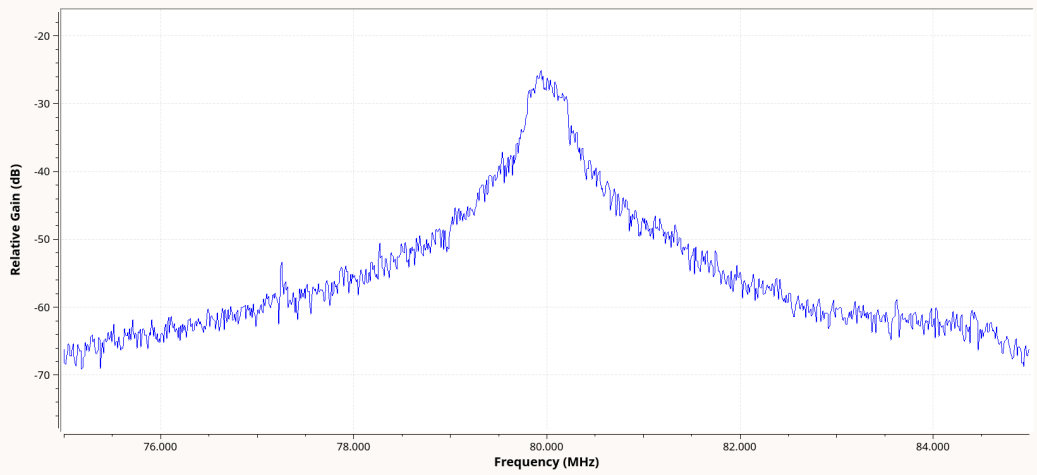


Figura 10.7: Espectro recibido de transmisión ISDB-T con Raspberry Pi.

este dentro del rango compatible según el proyecto RpiTX [1].

En la figura 10.7 se puede observar el espectro recibido por el sistema de recepción. Se visualiza un espectro que no aparenta ser de tipo OFDM, y efectivamente se puede comprobar que esta señal no se logra demodular al ser inyectada al *software* receptor ISDB-T one-seg. Tal como se explicó previamente, se entiende que la tasa de muestreo es demasiado alta para lo que RpiTX soporta.

Para evadir este problema se decidió para una nueva prueba, bajar la tasa de muestreo a un valor que sea soportado teóricamente por RpiTX. Se dividió por 4 la tasa de muestreo utilizada para la transmisión ISDB-T one-seg, pasando de 507,937 kHz a 126,984 kHz. Para que no se produzcan problemas en la reproducción de la señal recibida (debido a la tasa de muestreo menor que la estipulada), se le dio un tiempo al receptor para que este vaya recibiendo y procesando la señal a esta tasa menor, para luego sí intentar reproducir el *transport stream* recibido. Probando una transmisión de SDR a SDR con este método, se logró encontrar que la señal se podía recibir y demodular correctamente. Ahora, probando de Raspberry Pi a SDR, se encontró con la misma situación que la prueba anterior transmitiendo con la placa. No es posible demodular la señal en recepción y la forma del espectro que se visualiza es igual al de la figura 10.7, con la salvedad de que el ancho del espectro es menor debido a la menor tasa de muestreo que se está utilizando. También en esta prueba se procedió a relevar la constelación en recepción, la cual se puede observar en la figura 10.8. En la constelación se pueden visualizar como circunferencias los distintos niveles de amplitud asociados a la cuantización del módulo de los complejos. De cualquier manera, se puede observar un patrón no lineal entre cada nivel de cuantización, los niveles altos están más cercanos entre sí y los niveles bajos están más alejados entre ellos. Originalmente el módulo de cada complejo fue cuantizado linealmente (según se observa en la sección 9.3) y debido a esto se esperaban recibir niveles que estén equiespaciados entre ellos.

[1] El rango es de 5 kHz - 1500 MHz.

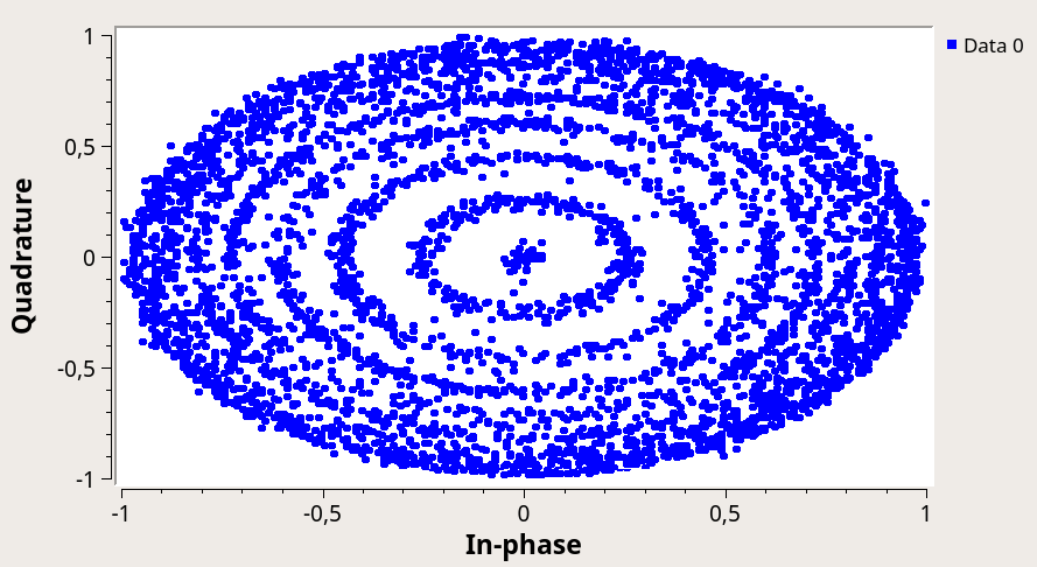


Figura 10.8: Constelación recibida de transmisión ISDB-T con Raspberry Pi.

Este comportamiento se entiende que introducirá errores al intentar demodular la señal. Encontrar la causa de este problema es un trabajo que necesita de un análisis mayor del funcionamiento de la placa, y se dejará para una investigación a futuro. Cabe destacar que este mismo fenómeno sucede en la transmisión de una señal AM, por lo cual se entiende que la mala calidad de esa transmisión está causada en parte por este mismo problema.

Otra posible fuente de error en la transmisión es el efecto del ruido de cuantización, comentado en la transmisión AM. Una señal OFDM, en forma similar a una señal AM, se ve perjudicada por la baja cantidad de niveles disponibles para la variación en amplitud de la señal de reloj. Como forma de poder mitigar este problema, se decidió aplicar *dithering* en transmisión. Esto implica agregar ruido blanco por fuera del ancho de banda de la señal para disminuir el efecto del ruido generado por la cuantización. En la figura 10.9 se puede observar el diagrama de GNU Radio que muestra la aplicación de *dithering* justo antes del envío de las muestras hacia la Raspberry Pi. Para esta prueba se trabajó a la misma tasa de muestreo total que en la prueba anterior (126,984 kHz), aunque las muestras de la señal ISDB-T se procesaron a la mitad de la tasa total de muestreo (63,492 kHz) para poder aplicar *dithering* correctamente. De cualquier manera, no fue posible demodular la señal recibida y la forma de espectro que se visualiza, así como la constelación, son iguales al de las pruebas anteriores. De este resultado se desprende que no es posible mitigar los distintos efectos que afectan a la señal, para poder lograr una recepción satisfactoria.

En base a los resultados obtenidos, se puede entender que la placa Raspberry Pi tiene serias limitaciones como transmisor SDR. Esto se hace notorio en toda modulación que requiera variaciones de amplitud de la señal a transmitir. Tanto la no linealidad en los niveles de cuantización, como la baja cantidad de niveles

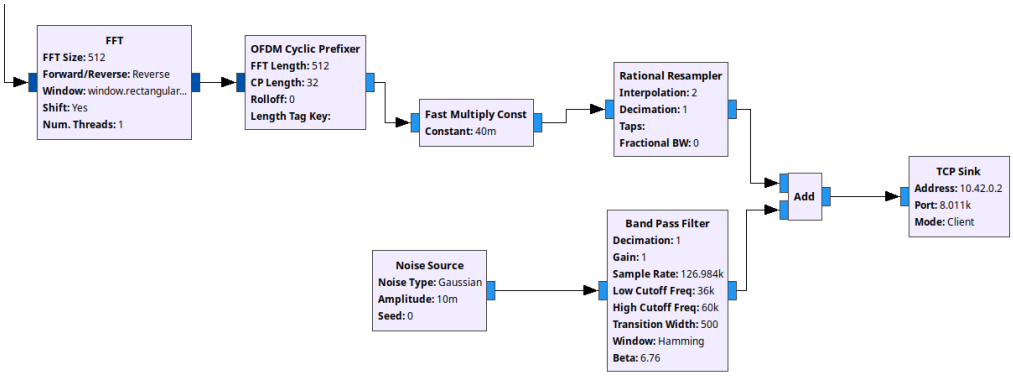


Figura 10.9: *Flowgraph* de última parte de transmisor ISDB-T con aplicación de *dithering*.

de amplitud disponibles, generan que las transmisiones sean de muy mala calidad (caso AM) o directamente inviables (caso ISDB-T). De cualquier manera, su buena resolución en las variaciones de frecuencia instantánea lo hacen un transmisor aceptable para FM.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 11

Conclusiones

El proyecto tuvo como tema principal la transmisión digital de señales de televisión utilizando la norma ISDB-T. En base a este, se recorrió una variedad de temas involucrados en el proceso de emisión de señales de televisión. Desde la codificación de vídeo previa, adaptación del *transport stream* a la norma ISDB-T y transmisión hacia el medio físico.

Como base del proyecto, se partió del *software* transmisor ISDB-T full-seg explicado en la sección 4.1. Este transmisor es capaz de trabajar con TSs independientes para cada capa jerárquica como entrada. Entre los logros del proyecto, se realizaron modificaciones al transmisor para que además admita BTSs como entrada. También se le realizaron modificaciones para lograr que este funcione para todas las combinaciones de parámetros posibles. Fue posible constatar el correcto funcionamiento del mismo a partir de múltiples pruebas.

Además, se investigó el proceso de codificación de vídeo en base a los requerimientos de la norma ISDB-Tb, con el objetivo de poder generar *transport streams* que luego sean inyectados al transmisor. Se logró este objetivo, tanto para videos almacenados previamente en el equipo, como también para videos generados en tiempo real. Se comprobó el funcionamiento a partir de la transmisión de una variedad de *transport streams* generados, utilizando *hardware* SDR convencional y un adaptador USB-VGA como transmisores, y equipos SDR y sintonizadores comerciales como receptores. Tanto en los receptores SDR como en varios de los sintonizadores comerciales, se pudo corroborar el correcto funcionamiento del sistema. De cualquier manera, surgieron algunos inconvenientes relacionados al proceso de generación de *transport streams*. Al transmitir estos hacia algunos de los sintonizadores ISBD-Tb comerciales, se encontró que podían surgir problemas al sintonizar múltiples servicios que se encuentren en distintas capas jerárquicas. Se entendió que una posible explicación de esto, es la falta de tablas PSI correctas en cada uno de los *transport stream* que se generan. Queda como trabajo a futuro investigar en mayor medida como introducir estas tablas, para lograr la sintonización correcta de los distintos servicios.

Otro objetivo de este proyecto fue implementar el transmisor con *hardware* SDR de bajo costo sustituyendo a los transmisores SDR convencionales. Teniendo en cuenta las limitaciones de potencia de transmisión de estos dispositivos, fue que

se implementó el transmisor one-seg de ISDB-T, siendo uno de los grandes aportes que ofrece este trabajo. Si bien este no estaba previsto en un principio, consideramos fundamental su implementación para llevar a cabo algunos objetivos del proyecto. Las pruebas realizadas y expuestas en la tesis comprobaron su correcto funcionamiento. A su vez, la implementación de cada bloque, como el estudio del flujo de datos a través de estos, permitió consolidar un mayor entendimiento de la norma ISDB-T.

Utilizando el adaptador USB-VGA como *hardware* SDR de bajo costo, se consiguieron resultados muy interesantes. Se logró transmitir tanto señales de televisión full-seg como one-seg a través de este. Seleccionando bajas frecuencias de transmisión (mencionadas en el documento), la señal pudo ser correctamente recibida por otros equipos SDR para diferentes distancias entre los equipos. Dado que la potencia total de transmisión del adaptador es constante, la señal one-seg presentó una mayor potencia en recepción que el caso full-seg, a iguales distancias. Debiéndose esto a que para una misma potencia total de transmisión, se está transmitiendo un menor ancho de banda. Basado en esto último, se logró determinar el mayor alcance del adaptador USB-VGA como transmisor ISDB-T sin la utilización de amplificadores. Se logró para una transmisión one-seg, un alcance máximo de 20 metros de distancia entre los equipos transmisor y receptor, recibiendo la señal correctamente. La transmisión de señales full-seg a través del adaptador permitió utilizar como receptor un sintonizador ISDB-Tb comercial. Transmitiendo en el canal 177,143 MHz y conectando un amplificador de radiofrecuencia al adaptador, se logró decodificar de manera correcta la señal en el televisor. Este resultado fue importante para el proyecto, dado que se logró transmitir televisión digital a una frecuencia disponible para los sintonizadores ISDB-Tb con un equipo transmisor cuyo costo es de 20 U$S.

Con las pruebas realizadas se puede constatar que el comportamiento del adaptador USB-VGA como transmisor es similar al de un USRP, siempre y cuando se trabaje a frecuencias bajas, pequeñas distancias y en transmisiones one-seg. Esto muestra que la principal limitante del equipo adaptador USB-VGA, es la potencia de transmisión. Esto se podría solucionar con amplificadores en las frecuencias en las que se desee transmitir. Otra contrapartida no menor que tiene la utilización del adaptador USB-VGA es que el espectro de la señal a transmitir es teóricamente infinita. A pesar de que los armónicos de la señal se van atenuando, los primeros armónicos no son despreciables, y ocupan parte del espectro que no debería ser utilizado. Esto se puede solucionar utilizando filtros pasa-banda de radio frecuencia, seleccionando la frecuencia en la que se desee transmitir. Otra solución sería utilizar dos filtros, uno pasa-bajos y otro pasa-altos, dejando la banda pasante en donde se desee transmitir. Esta última alternativa es más sencilla de implementar, dado que los filtros pasa-banda de radio frecuencia no son tan comunes en el mercado. A lo largo del proyecto se pudo probar satisfactoriamente la utilización de un filtro pasa-altos para eliminar parte de estos armónicos.

Este trabajo hace también un aporte sobre la placa Raspberry Pi 3B+ como transmisor de bajo costo. Se logró entender y documentar varios aspectos del funcionamiento de los proyectos RpiTX y RpiDATV, proyectos de los cuales no hay

prácticamente ninguna documentación previa en la cual poder basarse. Como se vislumbró en los análisis realizados, se entiende que la placa no es un buen transmisor SDR. Modulaciones que transmitan información en la amplitud de la señal van a verse seriamente afectadas por distintos fenómenos destructivos. Se entiende que existe trabajo por hacer para comprender en mejor manera estos problemas y posiblemente encontrar soluciones. De cualquier manera, consideramos que actualmente el funcionamiento de la placa como SDR puede ser aceptable en aplicaciones en donde la modulación en amplitud no sea determinante.

Finalmente, esta tesis es un aporte a la serie de investigaciones llevadas a cabo por la Facultad de Ingeniería de la Universidad de la República. Creemos además, que es un importante aporte para aficionados a los SDR, ya que al exponer las características de estos dispositivos de bajo costo como transmisores, se puede trasladar su uso a un sinnúmero de aplicaciones. Se obtuvieron resultados muy interesantes a partir de este trabajo y consideramos que los objetivos fueron logrados con éxito, tanto la investigación de los dispositivos de bajo costo como la continuación en el desarrollo en software del transmisor. Es un área con mucho potencial para crecer, y exhortamos la continuidad de esta línea de investigación.

Esta página ha sido intencionalmente dejada en blanco.

Referencias

[1] Associação Brasileira de Normas Técnicas. ABNT NBR 15601:2007 - Televisión digital terrestre — Sistema de transmisión ISDB-Tb. URL: `http://www.telemidia.puc-rio.br/~rafaeldiniz/public_files/normas/SBTVD/es/Transmicion/15601.pdf`. Online; accedido: 22-Mayo-2020.

[2] Association of Radio Industries and Businesses. ARIB STD-B31 - Transmission System for Digital Terrestrial Television Broadcasting. URL: `https://www.arib.or.jp/english/std_tr/broadcasting/std-b31.html`. Online; accedido: 28-Mayo-2020.

[3] Martin Ewing. *The ABC's of Software Defined Radio*. The American Radio Relay League, Inc., 2012.

[4] Pablo Belzarena, Pablo Flores Guridi, Gabriel Gómez Sena, Víctor Gonzalez Barbone, Federico La Rocca. Implementación de un transceptor de ISDB-T abierto y para metrología bajo el paradigma de Radio Definida por Software. Facultad de Ingeniería, Universidad de la República, Uruguay. URL: `https://iie.fing.edu.uy/investigacion/grupos/artes/projects/gr-isdbt/`.

[5] Pablo Belzarena, Pablo Flores Guridi, Gabriel Gómez Sena, Víctor González Barbone, Federico La Rocca. An open and free ISDB-T full seg receiver implemented in GNU radio. *Wireless Innovation Forum Conference on Wireless Communications Technologies and Software Defined Radio (WInnComm 16), Reston, Virginia, USA*, Marzo 2016.

[6] Javier Hernández, Santiago Castro. *Implementación de un Transmisor de ISDB-T Abierto Bajo el Paradigma de Radio Definida por Software*. Proyecto de fin de carrera, Facultad de Ingeniería, Universidad de la República, Uruguay, 2018.

[7] Steve Markgraf. OSMO-FL2K. URL: `https://osmocom.org/projects/osmo-fl2k/wiki`. Online; accedido: 28-Enero-2019.

[8] Evariste Courjaud. RpiTX. Repositorio Github. URL: `https://github.com/F5OEO/rpitx`. Online; accedido: 01-Abril-2020.

[9] Evariste Courjaud. RpiDATV. Repositorio Github. URL: `https://github.com/F5OEO/rpidatv`. Online; accedido: 01-Abril-2020.

[10] Moving Picture Experts Group. ISO/IEC 13818-1: Information technology — Generic coding of moving pictures and associated audio information: Systems. URL: https://ecee.colorado.edu/~ecen5653/ecen5653/papers/iso13818-1.pdf. Online; accedido: 22-Mayo-2020.

[11] Pablo Belzarena, Pablo Flores Guridi, Gabriel Gómez, Víctor González-Barbone, Federico La Rocca, Martín Randall, Paola Romero. Radio Definida por Software (SDR). Facultad de Ingeniería, Universidad de la República, Uruguay. URL: https://iie.fing.edu.uy/investigacion/grupos/artes-old/gr-isdbt/papers/poster_ing_demuestra.pdf. Online; accedido: 28-Enero-2020.

[12] RTL-SDR.com. About RTL-SDR. URL: https://www.rtl-sdr.com/about-rtl-sdr/. Online; accedido: 11-Mayo-2020.

[13] Ettus Research. *USRP B200/B210 Bus Series*. Online; accedido: 13-Mayo-2020.

[14] GNU Radio - The Free & Open Source Radio Ecosystem. URL: https://www.gnuradio.org/. Online; accedido: 22-Mayo-2020.

[15] Web del curso Sistemas de Comunicación. Facultad de Ingeniería, Universidad de la República, Uruguay. URL: https://eva.fing.edu.uy/course/view.php?id=602. Online; accedido: 22-Mayo-2020.

[16] Pablo Belzarena, Federico La Rocca. *Comunicaciones inalámbricas - notas del curso*. Facultad de Ingeniería, Universidad de la República, Uruguay, 2017.

[17] Ove Edfors, Magnus Sandell, Jan-Jaap van de Beek, Daniel Landstrom, Frank Sjoberg. An introduction to orthogonal frequency-division multiplexing. *Lulea University of Technology*, 1996.

[18] Pablo Flores Guridi. *La norma ISDB-T y un receptor implementado en SDR*. Tesis de maestría, Facultad de Ingeniería, Universidad de la República, Uruguay, 2016.

[19] Associação Brasileira de Normas Técnicas. ABNT NBR 15602-1:2007 - Televisión digital terrestre — Codificación de video, audio y multiplexación. Parte 1: Codificación de video. URL: http://www.telemidia.puc-rio.br/~rafaeldiniz/public_files/normas/SBTVD/es/Codificacion/ABNTNBR15602_2D1_2007Esp_2008.pdf. Online; accedido: 22-Mayo-2020.

[20] Associação Brasileira de Normas Técnicas. ABNT NBR 15602-2:2007 - Televisión digital terrestre — Codificación de video, audio y multiplexación. Parte 2: Codificación de audio. URL: http://www.telemidia.puc-rio.br/~rafaeldiniz/public_files/normas/SBTVD/es/Codificacion/ABNTNBR15602_2D2_2007Esp_2008.pdf. Online; accedido: 22-Mayo-2020.

[21] Associação Brasileira de Normas Técnicas. ABNT NBR 15602-3:2007 - Televisión digital terrestre — Codificación de video, audio y multiplexación. Parte 3: Sistemas de multiplexación de señales. URL: http://www.telemidia.puc-rio.br/~rafaeldiniz/public_files/normas/SBTVD/es/Codificacion/ABNTNBR15602_2D3_2007Esp_2008.pdf. Online; accedido: 22-Mayo-2020.

[22] Moving Picture Experts Group, Video Coding Experts Group. ITU-T H.264 - Infrastructure of audiovisual services – Coding of moving video - Advanced video coding for generic audiovisual services. URL: https://www.itu.int/rec/T-REC-H.264-201906-I/es. Online; accedido: 28-Mayo-2020.

[23] Moving Picture Experts Group. ISO/IEC 14496-3: Information technology — Coding of audio-visual objects - Part 3: Audio. URL: http://read.pudn.com/downloads98/doc/comm/401153/14496/ISO_IEC_14496-3%20Part%203%20Audio/C036083E_SUB1.PDF. Online; accedido: 28-Mayo-2020.

[24] Grupo ARTES. ISDB-T transceiver in GNU Radio. Repositorio Github. URL: https://github.com/git-artes/gr-isdbt. Online; accedido: 22-Mayo-2020.

[25] IKUSI. *FLASHD NANO*. Online; accedido: 13-Mayo-2020.

[26] Ettus Research. *Ettus Research USRP B100 Software Defined Radio*. Online; accedido: 13-Mayo-2020.

[27] ffmpeg. URL: https://www.ffmpeg.org/.

[28] Fabrice Bellard. Analog and Digital TV (DVB-T) Signal Generation. URL: https://bellard.org/dvbt/. Online; accedido: 28-Enero-2020.

[29] Martin Marinov. TempestSDR. URL: https://github.com/martinmarinov/TempestSDR. Online; accedido: 14-Mayo-2020.

[30] Federico La Rocca. Tempest. Repositorio Github. URL: https://github.com/git-artes/gr-tempest. Online; accedido: 14-Mayo-2020.

[31] GNU Radio. Polyphase Arbitrary Resampler. URL: https://wiki.gnuradio.org/index.php/Polyphase_Arbitrary_Resampler. Online; accedido: 28-Enero-2020.

[32] GNU Radio. Complex To Float. URL: https://wiki.gnuradio.org/index.php/Complex_To_Float. Online; accedido: 28-Enero-2020.

[33] COMMSCOPE. SVA15PRSMS. URL: https://www.commscope.com/globalassets/digizuite/81729-p360-sva15prsms-external.pdf?r=1. Online; accedido: 27-Abril-2020.

[34] Grupo ARTES. gr-MER. Repositorio Github. URL: https://github.com/git-artes/gr-mer. Online; accedido: 13-Mayo-2020.

[35] The Raspberry Pi Foundation. URL: https://www.raspberrypi.org/. Online; accedido: 12-Mayo-2020.

[36] European Telecommunications Standards Institute. EN 300 421 - Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for 11/12 GHz satellite services. URL: https://www.etsi.org/deliver/etsi_en/300400_300499/300421/01.01.02_60/en_300421v010102p.pdf. Online; accedido: 10-Junio-2020.

[37] Evariste Courjaud. Librpitx. Repositorio Github. URL: https://github.com/F5OEO/librpitx. Online; accedido: 03-Junio-2020.

[38] Broadcom Europe Ltd. *BCM2835 ARM Peripherals*.

[39] Dani S. Scratch en Raspberry Pi. Tutorial de instalación para computación física. URL: https://juegosrobotica.es/scratch-en-raspberry/. Online; accedido: 04-Abril-2020.

[40] Tomaž Šolc. Notes on the general-purpose clock on BCM2835. URL: https://www.tablix.org/~avian/blog/archives/2018/02/notes_on_the_general_purpose_clock_on_bcm2835/. Online; accedido: 06-Abril-2020.

[41] GPIO pads control. URL: https://www.raspberrypi.org/documentation/hardware/raspberrypi/gpio/gpio_pads_control.md. Online; accedido: 22-Mayo-2020.

[42] Raspbian. URL: https://www.raspbian.org/. Online; accedido: 14-Mayo-2020.

Índice de tablas

2.1. Dispositivos SDR más conocidos en el mercado. 6

3.1. Ceros necesarios para el *zero-padding* de la IFFT [18]. 23
3.2. Resoluciones y razón de aspecto para *codec* H.264 en full-seg. . . . 24
3.3. Tasa de cuadros y sistema de escaneo para *codec* H.264 en full-seg. 25
3.4. Resoluciones y razón de aspecto para *codec* H.264 en one-seg. . . . 25

4.1. Cantidad de portadoras por segmento para cada modo. 32
4.2. Ceros necesarios para el zero-padding de la IFFT. 32
4.3. Posiciones de portadoras TMCC y pilotos AC del segmento central relativo a la cantidad de portadoras activas en full-seg para cada modo de transmisión. 35
4.4. Parámetros utilizados para la prueba N^o1. 37
4.5. Parámetros utilizados para la prueba N^o2. 38
4.6. Parámetros utilizados para la prueba N^o3. 38

7.1. Parámetros utilizados para las pruebas en transmisor ISDB-T. . . 59
7.2. Medidas de MER en transmisión one-seg. 63
7.3. Medidas de MER en transmisión full-seg sin amplificador. 66
7.4. Medidas de MER en transmisión full-seg con amplificador. 67

8.1. Medidas de MER en transmisión one-seg a $60MHz$. 76
8.2. Medidas de MER en transmisión one-seg a $173MHz$. 77
8.3. Medidas de MER en transmisión one-seg para distintas distancias. 77
8.4. Medidas de MER en transmisión one-seg a 80 MHz. 78
8.5. Medidas de MER en transmisión full-seg para distintas distancias. 79

9.1. Corriente entregada según valor del registro. 91

Esta página ha sido intencionalmente dejada en blanco.

Índice de figuras

2.1. Estación de FM [11]. 6
2.2. RTL-SDR [12]. 7
2.3. USRP B200 [13]. 7
2.4. Diagrama de bloques de un demodulador AM implementado en GNU Radio Companion [15]. 9
2.5. Esquema básico de un sistema OFDM [17]. 10
2.6. El prefijo cíclico es una copia de la ultima parte del símbolo OFDM [17]. 11
2.7. Sistema OFDM ideal continuo en banda base [17]. 12
2.8. Modelo de sistema OFDM discreto [17]. 14

3.1. Canal de 6MHz. 18
3.2. Sistema transmisor ISDB-T [18]. 19
3.3. Formato de un TSP [18] 20
3.4. Diagrama del sistema para el entrelazamiento de bit y la modulación QPSK [18]. 21
3.5. Estructura de cuadro ISDB-T en modo de transmisión 1 [18]. . . . 22
3.6. Tabla de cálculo de *bitrates* [2]. 28

4.1. Transmisor de trece segmentos que parte de TSs independientes, implementado en GNU Radio. 30
4.2. Transmisor de trece segmentos que parte de un BTS, implementado en GNU Radio. 30
4.3. Transmisor one-seg implementado en GNU Radio. 33
4.4. Entrelazado temporal en transmisor de un segmento [18]. 34
4.5. Prueba N^o1 realizada en *software*. 38
4.6. Prueba N^o2. 39
4.7. Prueba N^o3. 39

5.1. Transmisión ISDB-T en tiempo real tomando imagen y sonido desde cámara web. 44

6.1. Espionaje de monitor utilizando implementación de TEMPEST [30]. 46
6.2. Adaptador USB 3.0 a VGA basados en el chip Fresco Logic FL2000. 46
6.3. Chip Fresco Logic FL2000. 47

6.4. Densidad espectral de potencia de la señal multiplicada por densidad espectral de potencia del pulso conformador. 48
6.5. Densidad espectral de potencia de la señal enviada por el adaptador USB-VGA. 48
6.6. Diagrama de bloques de archivo `fl2k_transmit_isdb-t.grc`. . . . 50
6.7. Re-muestreo de datos. 50
6.8. Diagrama de bloques de conversión en fase y cuadratura. 51
6.9. Densidad espectral de potencia de la señal en banda base. 51
6.10. Densidad espectral de potencia de la señal en fase y cuadratura. . 51
6.11. Densidad espectral de potencia de la señal en fase y cuadratura replicada cada múltiplos de f_{out}. 52
6.12. Densidad espectral de potencia de la señal transmitida por el adaptador USB-VGA. 53
6.13. Densidad espectral de potencia de la señal transmitida por el adaptador USB-VGA. 54

7.1. Pines del adaptador USB-VGA. 56
7.2. Adaptador USB-VGA. 56
7.3. Adaptador VGA a BNC. 57
7.4. Adaptador BNC a SMA. 57
7.5. Amplificador CommScope sva15prsms. 58
7.6. Antena monopolo de $\frac{1}{4}$ de longitud de onda. 58
7.7. Diagrama de conexión. 58
7.8. Laboratorio de Medidas del Instituto de Ingeniería Eléctrica. . . . 60
7.9. Transmisión one-seg con 60 cm de distancia entre transmisor y receptor. 61
7.10. Transmisión one-seg con 240 cm de distancia entre transmisor y receptor. 62
7.11. Transmisión one-seg con 500 cm de distancia entre transmisor y receptor. 62
7.12. *Transport stream* one-seg a 240 cm de distancia entre transmisor y receptor. 62
7.13. *Transport stream* de transmisión one-seg a 2000 cm de distancia entre transmisor y receptor. 63
7.14. *Transport stream* de transmisión one-seg a 60 cm de distancia entre transmisor y receptor utilizando modulación 64-QAM. 64
7.15. Transmisión full-seg con 60cm de distancia entre transmisor y receptor sin amplificador. 65
7.16. Transmisión full-seg con 240cm de distancia entre transmisor y receptor sin amplificador. 65
7.17. Transmisión full-seg con 500cm de distancia entre transmisor y receptor sin amplificador. 66
7.18. *Transport stream* de transmisión full-seg a 60cm de distancia entre transmisor y receptor sin amplificador. 67
7.19. Transmisión full-seg con 60 cm de distancia entre transmisor y receptor con amplificador. 68

7.20. Transmisión full-seg con 240cm de distancia entre transmisor y receptor con amplificador. 68
7.21. Transmisión full-seg con 500 cm de distancia entre transmisor y receptor con amplificador. 69
7.22. *Transport stream* de transmisión full-seg a 240 cm de distancia entre transmisor y receptor con amplificador. 69
7.23. Modulación 64QAM en transmisión full-seg con amplificador, a 60cm de distancia entre transmisor y receptor. 70
7.24. TruSpec HPF-54MHz. 71
7.25. Respuesta en frecuencia del TruSpec HPF-54MHz. 71
7.26. Espectro de la señal enviada con su primer armónico. 71
7.27. Espectro de la señal a 85 MHz sin filtro pasa alto. 72
7.28. Espectro de señal a 85 MHz con filtro pasa alto. 72
7.29. Espectro de la señal a 42 MHz sin filtro pasa alto. 73
7.30. Espectro de la señal a 42 MHz con filtro pasa alto. 73
7.31. Espectro visualizado a 42 MHz sin transmisión. 73

8.1. Constelaciones transmitiendo a 60 MHz con distancias cercanas entre transmisor y receptor. 76
8.2. Constelaciones transmitiendo a 173 MHz con distancias cercanas entre transmisor y receptor. 77
8.3. Constelaciones transmitiendo a 60 MHz a una distancia de 500 cm entre transmisor y receptor. 78
8.4. Constelaciones transmitiendo a 80 MHz con distancias cercanas entre transmisor y receptor. 79
8.5. Constelaciones transmitiendo full-seg a 80 MHz a una distancia de 500 cm entre transmisor y receptor. 80
8.6. Transmisión ISDB-Tb hacia TV con USB-VGA. 81
8.7. Transmisión ISDB-Tb entre USB-VGA y USRP en Full-Seg en 177,143MHz a 50cm de distancia. 82

9.1. Raspberry Pi 3B+. 86
9.2. Pines GPIO en Raspberry Pi 3B+ [39]. 87
9.3. Menú en consola en proyecto RpiTX [8]. 87
9.4. Símbolos QPSK. 89
9.5. Diagrama “Clock Tree” [40]. 91
9.6. *Buffers* triestado que regulan corriente en GPIO [41]. 92

10.1. Configuración de Raspberry Pi para transmisión. 96
10.2. Transmisor FM de banda angosta en GNU Radio. 96
10.3. Espectro recibido de transmisión FM con Raspberry Pi. 98
10.4. Constelación recibida de transmisión FM con Raspberry Pi. 98
10.5. Transmisor AM en GNU Radio. 99
10.6. Espectro recibido de transmisión AM con Raspberry Pi. 100
10.7. Espectro recibido de transmisión ISDB-T con Raspberry Pi. 101
10.8. Constelación recibida de transmisión ISDB-T con Raspberry Pi. . 102

10.9. *Flowgraph* de última parte de transmisor ISDB-T con aplicación de *dithering*. 103

Esta es la última página.
Compilado el jueves 20 agosto, 2020.
http://iie.fing.edu.uy/