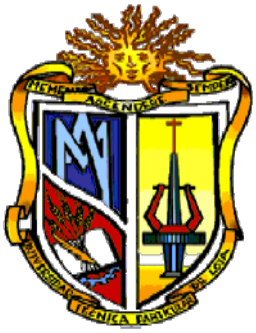

UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

ÁREA TÉCNICA

TITULACIÓN DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

Desarrollo de sistemas de comunicaciones usando tecnología libre de Radio Definida mediante Software

TRABAJO DE FIN DE TITULACIÓN

AUTORES: Peralta Vallejo, Max Andrés
Sócola Coronel, Alexander Patricio

DIRECTOR: Quiñones Cuenca, Manuel Fernando, Ing.

LOJA – ECUADOR

2014

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE FIN DE TITULACIÓN

Ingeniero.

Manuel Fernando Quiñones Cuenca

DIRECTOR DEL TRABAJO DE FIN DE TITULACIÓN

De mi consideración:

Que el presente trabajo de fin de titulación: “Desarrollo de sistemas de comunicaciones usando tecnología libre de Radio Definida mediante Software”, realizado por los profesionales en formación: Peralta Vallejo Max Andrés y Sócola Coronel Alexander Patricio; ha sido orientado y revisado durante su ejecución, por lo cual se aprueba la presentación del mismo.

Loja, diciembre de 2014

f) …….

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

Nosotros, Peralta Vallejo Max Andrés y Sócola Coronel Alexander Patricio, declaramos ser autores del presente trabajo de fin de titulación: Desarrollo de sistemas de comunicaciones usando tecnología libre de Radio Definida mediante Software, de la titulación de Electrónica y Telecomunicaciones, siendo el Ing. Manuel Fernando Quiñones Cuenca director del presente trabajo; y eximimos expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales.

Adicionalmente declaramos conocer y aceptar la disposición del Art. 67 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado que se realicen a través, o con el apoyo financiero, académico o institucional (operativo) de la Universidad"

f. .....

Autor: Peralta Vallejo Max Andrés

Cédula: 1104969165

f. .....

Autor: Sócola Coronel Alexander Patricio

Cédula: 1105028540

DEDICATORIA

Dedico este trabajo principalmente a mis padres, gracias a su apoyo incondicional he podido llegar a este punto en mi formación profesional; a mi hermano, que me ha brindado motivación y ayuda en momentos difíciles; a mi familia y amigos que han sido pilares fundamentales en mi formación como persona y a nuestro Padre Celestial que nos cuida y guía nuestro camino.

Max Andrés

Esta tesis la dedico a mis padres quienes supieron brindarme su apoyo en diferentes ámbitos, pero principalmente en el campo emocional y moral, ya que cuando me veían desanimado y sin ganas de continuar adelante con sus palabras de aliento me dieron la fuerza suficiente para seguir y convertirme en la persona que soy ahora. A mi hermana que con su presencia y cariño me ha dado más que un motivo para superarme y a Dios que con su presencia a cada instante ha guiado y guiará mi camino para ser una persona y profesional de Bien

Alexander Patricio

AGRADECIMIENTO

A nuestros padres, por su apoyo incondicional y confianza. A nuestros hermanos, que han estado siempre motivándonos y ayudándonos día a día.

A nuestro director de tesis y amigo: Ing. Manuel Quiñones, que nos supo brindar su confianza y tiempo para guiarnos en la realización de este trabajo; además de contribuir con su experiencia y conocimiento para el desarrollo de nuevos talentos en la titulación.

A nuestra familia, amigos y compañeros que han sido parte de nuestra formación tanto personal como profesional.

Max Andrés y Alexander

ÍNDICE DE CONTENIDOS

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE FIN DE TITULACIÓN ii
DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS..... iii
DEDICATORIA iv
AGRADECIMIENTO v
ÍNDICE DE CONTENIDOS vi
ÍNDICE DE FIGURAS ix
ÍNDICE DE TABLAS xi
RESUMEN..... 12
ABSTRACT 13
INTRODUCCIÓN..... 14
CAPÍTULO I..... 15
1. ALCANCE DE LA INVESTIGACIÓN..... 15
1.1 Objetivos 16
1.1.1 Objetivo general. 16
1.1.2 Objetivos específicos. 16
1.2 Justificación 16
1.3 Metodología 16
CAPÍTULO II..... 18
2. ESTADO DEL ARTE 18
2.1 Introducción 19
2.2 Orígenes de *Software Radio* 19
2.3 Radio Definida mediante Software (SDR)..... 19
2.3.1 Definición y aspectos fundamentales de SDR. 20
2.3.2 Arquitectura de *hardware* SDR..... 20
2.3.2.1 Antena. 21
2.3.2.2 Front end de RF. 21
2.3.2.3 Oscilador local. 21
2.3.2.4 Bloque de frecuencia intermedia (FI). 22
2.3.2.5 Conversión AD/DA. 22
2.3.2.6 Modulador/Demodulador. 24
2.4 Plataformas de *hardware* para SDR 24
2.4.1 SRL QuickSilver QS1R. 25
2.4.2 WR-G31DDC 'EXCALIBUR' 25

2.4.3 FiFi SDR. 26
2.4.4 BladeRF. 27
2.4.5 EasySDR. 28
2.4.6 HackRF. 28
2.4.7 USRP N210. 29
2.5 Software para SDR 30
2.5.1 GNU Radio. 30
2.5.2 Software de desarrollo de sistemas NI LabVIEW. 31
2.5.3 MATLAB®. 32
2.6 Modulaciones 33
2.6.1 Modulaciones analógicas. 33
2.6.1.1 Modulación de amplitud. 34
2.6.1.2 Modulación angular 35
2.6.1.2.1 Modulación de fase (PM). 35
2.6.1.2.1 Modulación de frecuencia (FM). 36
2.6.2 Modulaciones digitales. 39
2.6.2.1 Modulación (PSK). 40
2.6.2.1.1 Modulación por desplazamiento de fase binaria (BPSK). 40
2.6.2.1.2 Modulación (QPSK). 41
2.6.2.2 Modulación QAM. 42
2.6.2.3 Modulación GMSK. 43
CAPÍTULO III 45
3. MATERIALES Y MÉTODOS 45
3.1 Universal Software Radio Peripheral (USRP) 46
3.1.1 Hardware del USRP N210. 46
3.1.1.1 Tarjeta principal 47
3.1.1.2 Tarjeta secundaria. 49
3.1.1.3 Panel frontal. 52
3.1.1.4 GPSDO. 53
3.1.2 UHD. 53
3.2 GNU Radio 54
3.2.1 GNU Radio *Companion.* 58
3.3 GStreamer 59
3.4 Mplayer 60

3.4.1 Lista de *codecs* más importantes. 61
CAPÍTULO IV 63
4. FUNCIONAMIENTO Y PRUEBAS 63
4.1 Conexión del equipo USRP N210 64
4.1.1 Caracterización de la conexión con el computador. 66
4.2 Sistemas de comunicaciones analógicos 67
4.2.1 Analizador de espectros. 67
4.2.2 Sistema de transmisión FM. 69
4.2.3 Sistema de recepción FM. 71
4.2.4 Sistema de transmisión FM con múltiples portadoras y múltiples fuentes de audio. 72
4.3 Sistemas de comunicaciones digitales 74
4.3.1 Transmisor digital. 74
4.3.2 Receptor digital. 77
4.3.3 Resultados obtenidos. 79
4.3.3.1 Sistema 4-QAM. 79
4.3.3.2 Sistema 4-PSK. 82
4.3.3.3 Sistema GMSK. 85
4.4 Sistema de comunicación experimental para la transmisión y recepción digital de video 88
4.4.1 Transmisor digital de video. 88
4.4.2 Receptor digital de video. 90
4.5 Cálculo de la tasa de error de bit (BER) 94
CONCLUSIONES 97
RECOMENDACIONES 98
REFERENCIAS 99
ANEXOS 102
ANEXO A 103
ANEXO B 106
ANEXO C 115
ANEXO D 118
ANEXO E 142
ANEXO F 145
ANEXO G 147

ÍNDICE DE FIGURAS

Figura 1.1. Fases de desarrollo del proyecto. 17

Figura 2.1. Arquitectura *hardware* de SDR. 21
Figura 2.2. *Digital down converter*..... 23
Figura 2.3. *Digital up converter*. 23
Figura 2.4. Dispositivo SRL QuickSilver QS1R 26
Figura 2.5. Dispositivo WR-G31DDC '*EXCALIBUR*'. 26
Figura 2.6. Dispositivo FiFi SDR. 27
Figura 2.7. Dispositivo bladeRF. 27
Figura 2.8. Dispositivo easySDR..... 28
Figura 2.9. Dispositivo HackRF. 29
Figura 2.10. Dispositivo de *hardware* USRP N210..... 30
Figura 2.11. Entorno de programación del *software* GNU Radio. 30
Figura 2.12. NI LabVIEW con la plataforma SDR. 31
Figura 2.13. MATLAB y Simulink como plataforma *software* de SDR. 32
Figura 2.14. Modulación AM con $m = 1$. 34
Figura 2.15. Modulación de fase. 36
Figura 2.16. Ancho de banda FM necesario en relación a mf. 38
Figura 2.17. Diagrama de sistemas de modulación I/Q..... 40
Figura 2.18. Modulación BPSK a) señal binaria, b) señal portadora, ... 41
Figura 2.19. Diagrama de constelación QAM. 43
Figura 2.20. Diagrama de fase para GMSK. 44

Figura 3.1. USRP N210. 46
Figura 3.2. Arquitectura *hardware* USRP N210..... 47
Figura 3.3. Funciones de la tarjeta principal. 47
Figura 3.4. Funciones de la tarjeta secundaria..... 49
Figura 3.5. Tarjeta secundaria WBX. 51
Figura 3.6. Antena VERT900. 51
Figura 3.7. Panel frontal del equipo USRP N200/210. 52
Figura 3.8. Arquitectura de GNU Radio..... 54
Figura 3.9. Integración de GNU Radio con bibliotecas externas. 55
Figura 3.10. Entorno de programación de GNU Radio *Companion*. 59
Figura 3.11. Captura de video con GStreamer desde la *webcam* del computador. 60
Figura 3.12. Reproductor Mplayer. 61

Figura 4.1. Diagrama de conexión del equipo con el computador. 64
Figura 4.2. Esquema de trabajo emisor-receptor 65
Figura 4.3. Respuesta del comando PING. 65
Figura 4.4. Esquema del analizador de espectros. 68
Figura 4.5. Configuración del bloque **UHD: USRP Source**. 68
Figura 4.6. Espectro de señales FM. 69
Figura 4.7. Esquema del transmisor FM. 69
Figura 4.8. Configuración del bloque **UHD: USRP Sink**. 70
Figura 4.9. Espectro FM transmitido (a) dominio frecuencial, (b) dominio temporal 71
Figura 4.10. Esquema del receptor FM. 71

Figura 4.11. Espectro de la señal FM recibida en la frecuencia 72
Figura 4.12. Esquema transmisor FM múltiples portadoras y múltiples fuentes de audio. 73
Figura 4.13. Configuración del **Signal Source**..... 73
Figura 4.14. Espectro transmisor FM con múltiples 74
Figura 4.15. Esquemático del transmisor digital. 74
Figura 4.16. Configuración del bloque **Signal Source**. 75
Figura 4.17. Configuración del bloque **QAM Mod**. 76
Figura 4.18. Configuración del bloque **PSK Mod**. 76
Figura 4.19. Configuración del bloque **GMSK Mod**. 76
Figura 4.20. Esquemático del receptor digital. 77
Figura 4.21. Configuración del bloque **QAM Demod**. 78
Figura 4.22. Configuración del bloque **PSK Demod**. 78
Figura 4.23. Configuración del bloque **GMSK Demod**. 79
Figura 4.24. Espectro de tonos de frecuencia 1 y 3 KHz. 79
Figura 4.25. Diagrama de constelación transmitida para 4-QAM. 80
Figura 4.26. Espectro transmitido para 4-QAM en la frecuencia 1.2 GHz. 80
Figura 4.27. Espectro recibido de modulación 4-QAM en la frecuencia 1.2 GHz. 81
Figura 4.28. Constelación recibida de modulación 4-QAM en la frecuencia 81
Figura 4.29. Espectro de los tonos recibidos y demodulados. 82
Figura 4.30. Espectro de tonos de frecuencia 1 KHz y 3 KHz. 82
Figura 4.31. Diagrama de constelación transmitido para 4-PSK. 83
Figura 4.32. Espectro transmitido para 4-PSK en la frecuencia 1.2 GHz. 83
Figura 4.33. Espectro recibido de 4-PSK en la frecuencia 1.2 GHz. 84
Figura 4.34. Constelación recibida de modulación 4-PSK en la frecuencia 84
Figura 4.35. Espectro de audio recibido y demodulado. 85
Figura 4.36. Espectro de tonos de frecuencia 1 KHz y 3 KHz. 85
Figura 4.37. Diagrama de constelación transmitido para GMSK. 86
Figura 4.38. Espectro transmitido para GMSK en la frecuencia 86
Figura 4.39. Espectro recibido de GMSK en la frecuencia 86
Figura 4.40. Diagrama de constelación recibido para GMSK. 87
Figura 4.41. Espectro de audio recibido y demodulado. 87
Figura 4.42. Inicialización del conducto **txfifo.ts** para el flujo de video. 89
Figura 4.43. Esquema del transmisor digital de video. 89
Figura 4.44. Configuración del bloque **File Source**..... 89
Figura 4.45. Conducto en modo *REPRODUCIENDO*. 90
Figura 4.46. Transmisión de video con modulación GMSK en la 90
Figura 4.47. Receptor digital de video en GNU Radio *Companion*. 91
Figura 4.48. Preparación del conducto rxfifo.ts. 91
Figura 4.49. Detección del formato *.ts* en el receptor. 91
Figura 4.50. Espectro recibido del transmisor digital de video en la 92
Figura 4.51. Señal de video demodulada y decodificada. 92
Figura 4.52. Señal decodificada de video con audio. 93
Figura 4.53. Reproducción de video con audio. 93
Figura 4.54. Espectro GMSK con nivel de ruido 10mV 94
Figura 4.55. a) imagen transmitida, b) imagen recibida. 95
Figura 4.56. Espectro GMSK con nivel de ruido 80mV 95
Figura 4.57. a) imagen transmitida, b) imagen recibida. 95

ÍNDICE DE TABLAS

Tabla 2.1. Comparación de alternativas para hardware SDR. 25
Tabla 2.2. Tipos de modulaciones mono portadoras. 33
Tabla 2.3 Características de modulación BPSK. 41
Tabla 2.4. Características de modulación QPSK. 42

Tabla 3.1. Características de tarjetas RF secundarias WBX y FRX900. 51
Tabla 3.2. Características de la antena VERT900. 52
Tabla 3.3. Valores de entrada permitidos en el panel frontal. 53
Tabla 3.4 Módulos de GNU Radio. 57
Tabla 4.1. Resultados obtenidos para el porcentaje de error de bit. 96

RESUMEN

En el presente trabajo de investigación se desarrollan sistemas de comunicaciones usando radio definida mediante *software* (SDR). Esta tecnología permite la integración de *software* libre y *hardware* flexible, logrando que un mismo elemento de *hardware* sea capaz de realizar diferentes funciones en distintos instantes de tiempo a través de cambios en la programación mediante *software* reconfigurable. Para cumplir con este propósito se ha utilizado el dispositivo USRP N210 que usa una interfaz Gigabit Ethernet para la comunicación con el computador, Ubuntu 13.04 (sistema operativo base) y GNU Radio para realizar la programación de los diferentes sistemas de comunicaciones.

Una de las ventajas más representativas de utilizar SDR es la reducción de costos así como el tamaño de los dispositivos ya que utilizan menos componentes de *hardware* y poseen flexibilidad para reconfigurar sus funciones dependiendo de las características del sistema y los requerimientos del usuario.

PALABRAS CLAVES: SDR, GNU Radio, USRP N210, Ubuntu, *software* reconfigurable, *hardware* flexible.

ABSTRACT

In the present research, communications systems using software defined radio (SDR) are developed. This technology allows the integration of free software and flexible hardware, making the same hardware device be able to perform different functions at different time instants through changes in programming by reconfigurable software. To fulfill this purpose we have employed the USRP N210 device, which uses a Gigabit Ethernet interface for communication with the computer, Ubuntu 13.04 (base operating system) and GNU Radio for programming different communication systems.

One of the most representative advantages of using SDR is the reduction of costs and the size of the devices since they use less hardware and they have flexibility to reconfigure their functions depending on system characteristics and user requirements.

KEYWORDS: SDR, GNU Radio, USRP N210, Ubuntu, reconfigurable software, flexible hardware.

INTRODUCCIÓN

Actualmente existen sistemas de comunicaciones que son empleados para una función específica y que operan dentro de una banda determinada; en algunos casos, se necesita de varios dispositivos para poder implementarlos. Por ejemplo un transmisor de radio FM utiliza diferentes etapas: pre-amplificación, modulación, oscilador de alta frecuencia, amplificador de potencia y finalmente la antena. Lo que se realiza en la presente investigación es que a partir de las ventajas que ofrece la Radio Definida mediante *Software* se pueden implementar dispositivos que permiten reemplazar algunos componentes de *hardware* por rutinas de *software* reduciendo considerablemente costos y el tamaño de dichos sistemas.

El desarrollo y programación de los diferentes sistemas de comunicaciones se lo realizó en GNU Radio *Companion* que es un entorno gráfico de programación a través de bloques que contienen moduladores analógicos y digitales, filtros, operadores aritméticos, etc., por medio de la combinación de los mismos se efectúa el procesamiento de señales; la comunicación con el USRP N210 se la realiza a través de la interfaz Gigabit Ethernet, el USRP es el dispositivo que realiza la conversión analógica a digital (ADC) o digital a analógica (DAC), así como la conversión de radiofrecuencia (RF) a una frecuencia intermedia (IF) o viceversa; este proceso se lo realizó en el sistema operativo Ubuntu que es de libre distribución.

En los diferentes capítulos de esta investigación se ha documentado información referente a la tecnología SDR, sus principales características y aplicaciones. Se dará una descripción de los materiales y requerimientos necesarios para aplicarla; así como las pruebas y funcionamiento de los diferentes sistemas de comunicaciónes tanto analógicos como digitales que se ha logrado implementar y finalmente se dará posibles tendencias o investigaciones que se puedan dar para un posible trabajo futuro.

CAPÍTULO I

ALCANCE DE LA INVESTIGACIÓN

1.1 Objetivos

1.1.1 Objetivo general.

Desarrollar e implementar sistemas de comunicaciones usando tecnología libre de Radio definida mediante *Software*.

1.1.2 Objetivos específicos.

- Montar sistemas de comunicaciones de bajo costo empleando radio definida mediante *software* y dispositivos USRP.

- Emplear sistemas y dispositivos adicionales con la finalidad de ampliar el alcance y las aplicaciones posibles de los sistemas de comunicaciones desarrollados.

- Realizar un manual de instalación, configuración y de la aplicación de los sistemas de comunicaciones implementados.

1.2 Justificación

En la actualidad existen sistemas de comunicaciones que tienen funcionalidades restringidas por su configuración de *hardware*, que en caso de necesitar modificaciones en su operación es necesario el reemplazo total de equipos o la compra de dispositivos adicionales. Con los sistemas de Radio Definida mediante *Software* es posible realizar tareas típicamente ejecutadas por componentes físicos a través de rutinas de *software* e implementar nuevas funcionalidades. Esto resulta de gran utilidad, no sólo por la reducción de costos asociados a la compra y renovación de equipos, sino porque permite reconfigurar todo el sistema de comunicación en caso de ser necesario.

Cabe mencionar que los sistemas SDR son una buena opción por sus características y tiempo de despliegue en sistemas de comunicaciones emergentes.

Es por estos motivos que se desarrolla el presente trabajo de investigación, demostrando que es posible implementar diferentes sistemas de comunicaciones tanto analógicos como digitales con un mismo dispositivo de *hardware* y un computador.

1.3 Metodología

Durante el presente proyecto de tesis, se ha seguido una metodología de avance por fases, las cuales se presentan a continuación:

La primera fase comprende una recolección de información para la ejecución del proyecto. Se han analizado artículos, documentos técnicos y publicaciones científicas con la finalidad de obtener la mayor información posible sobre sistemas de comunicaciones en SDR *(Software Defined Radio)*. Se resalta la importancia de la integración de los SDR en los sistemas de comunicaciones actuales; para ello se analiza la arquitectura que posee el dispositivo *Universal Software Radio Peripheral* (USRP), sus funciones en el desarrollo de los sistemas y su integración con el *software* GNU Radio.

La segunda fase comprende el desarrollo y programación de los sistemas de comunicación mediante SDR; abarcando el montaje de equipos USRP, instalación del sistema operativo Ubuntu sobre el cual se levanta todos los programas y *drivers* necesarios. Una vez instalado y realizadas las actualizaciones de *software* requeridas por el sistema operativo, se procede a instalar el *software* de desarrollo GNU Radio, el cual servirá como plataforma para la programación y pruebas de los sistemas de comunicaciones.

La tercera fase está enfocada a documentar todos los pasos a seguir para contar con los sistemas de comunicaciones en perfecto funcionamiento; además, las conclusiones y recomendaciones obtenidas al realizar varias pruebas con los sistemas implementados.

La metodología a seguir se resume en la Figura 1.1.

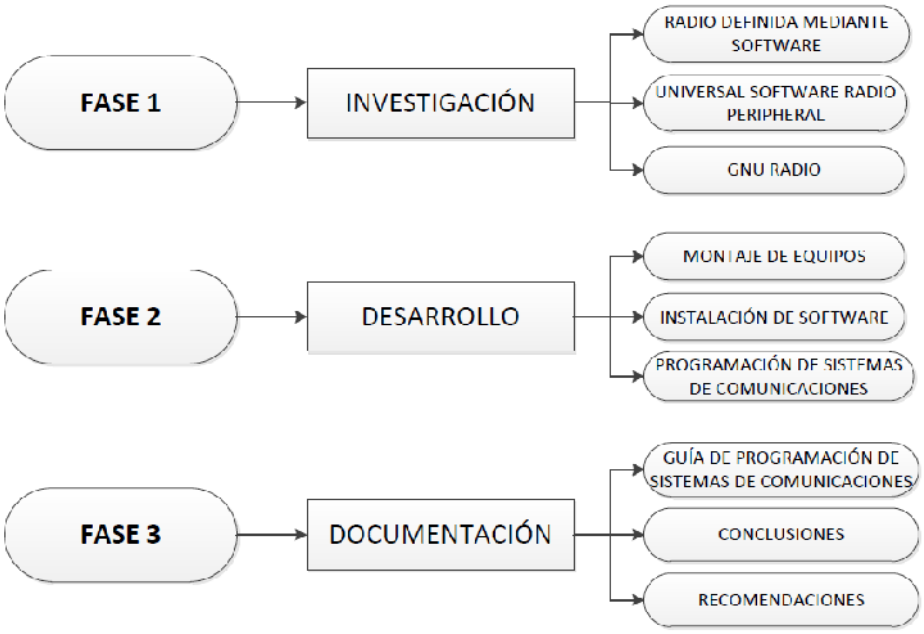

Figura 1.1. Fases de desarrollo del proyecto.
Fuente: Imagen propia de los autores.

CAPÍTULO II

ESTADO DEL ARTE

2.1 Introducción

El presente capítulo tiene la finalidad de dar a conocer los orígenes de la radio definida mediante *software*, la arquitectura de *hardware* que emplean los transmisores y receptores en esta tecnología y los tipos de modulaciones empleados en los sistemas de comunicaciones tanto analógicos como digitales.

2.2 Orígenes de *Software Radio*

El término SR *Software Radio* fue acuñado por Joseph Mitola III, en 1991, para referirse a un tipo de radios reprogramables o reconfigurables, equipos donde un mismo elemento de *hardware* es capaz de realizar diferentes funciones, en distintos instantes de tiempo, con la introducción de cambios en su configuración mediante *software* [19].

Por eso se considera que un SDR es una versión de un SR implementable con la tecnología disponible, donde la conversión se realiza en la etapa de Frecuencia Intermedia (IF), tras un filtrado selectivo [19].

La primera implementación conocida del concepto SDR fue el proyecto militar estadounidense *SpeakEasy*, cuyo objetivo principal era establecer más de diez tipos de tecnologías de telecomunicaciones inalámbricas (las más usadas por el ejército americano) en un solo equipo programable, que operaría en un rango de frecuencias desde los 2MHz hasta los 200MHz. Un objetivo adicional del proyecto era que el prototipo debía tener la posibilidad de actualizar su código para así tener en cuenta posibles futuros estándares. Dicho proyecto empezó en 1991 y sólo en 1995 fue posible lograr todos los objetivos planteados. Sin embargo, en el proyecto inicial sólo se podía mantener una comunicación a la vez, por lo cual se modificó y se planteó una segunda fase del mismo, en la que se trabajaron aspectos como la disminución de peso y coste, el incremento de la capacidad de procesamiento, la simultaneidad de comunicaciones y el diseño basado en *software* libre [19].

En 1996 Joseph Mitola funda el SDR *Forum* dando una descripción detallada de la tecnología. En la actualidad este foro se conoce como *Wireless Innovation Forum*, considerándose este como uno de los centros neurálgicos que acerca la tecnología *Software Defined Radio* y *Cognitive Radio* al mercado. En él, los miembros colaboran para acercar la experiencia obtenida. Este foro cuenta con miembros pertenecientes a diferentes tipos de organizaciones, ya sean organizaciones comerciales, de defensa o gubernamentales [18].

2.3 Radio Definida mediante Software (SDR)

SDR es una tecnología de comunicación de radio basada en un protocolo de comunicaciones inalámbricas definidas por *software* en lugar de su implementación en *hardware*, es capaz de

ser reprogramado y reconfigurado con el fin de operar con diferentes formas de onda y protocolos. SDR proporciona una solución eficiente y de bajo costo al construir o implementar dispositivos inalámbricos multimodo, multiportadora, multibanda y multifuncionales que pueden ser actualizados o mejorados a través de optimizaciones de *software* [20].

2.3.1 Definición y aspectos fundamentales de SDR.

Los desarrollos en radios inteligentes y adaptativos se han enmarcado a lo que hoy es un Radio Definido mediante *Software*, el cual es definido, según el *Wireless Inovation Forum*, de la siguiente manera: "*Radio en el cual algunas o todas las funciones de la capa física son definidas mediante software*".

SDR es una tecnología creada para mejorar la interoperabilidad entre diferentes servicios. Está compuesta de *software* y *hardware*, pudiendo ser reconfigurada dinámicamente para habilitar comunicaciones entre una amplia variedad de normas de comunicaciones, protocolos y radio enlaces [1]. SDR permite crear dispositivos inalámbricos y equipo de redes multibanda y multifuncionales, que pueden ser dinámicamente reconfigurados, o a través de actualizaciones de *software* y reconfiguraciones de *hardware*.

2.3.2 Arquitectura de *hardware* SDR.

La arquitectura de transmisores basados en *software* consiste en un subsistema digital y un subsistema analógico. Las funciones analógicas son restringidas a aquellas que no pueden ser mejoradas digitalmente, que son: antena, filtrado RF, combinación RF, preamplificación en recepción, transmisión de potencia de amplificación y generación de frecuencia de referencia [21]. En la Figura 2.1 se muestran las partes que conforman un *transceiver* de radio basado en *software.*

Como se aprecia en la Figura 2.1 de la arquitectura *hardware* para SDR, la parte analógica es la encargada de realizar todas aquellas operaciones como: alimentación a la antena, filtrado y combinación en RF, preamplificación, amplificación, y generación de la frecuencia de referencia. La idea de la arquitectura es que las etapas de conversión analógico/digital estén lo más cercanas posible a la antena, de hecho, la separación de portadoras y la conversión de frecuencias *up/down* son desempeñadas por los recursos de procesamiento digital, al igual que la codificación de canal y las modulaciones [17].

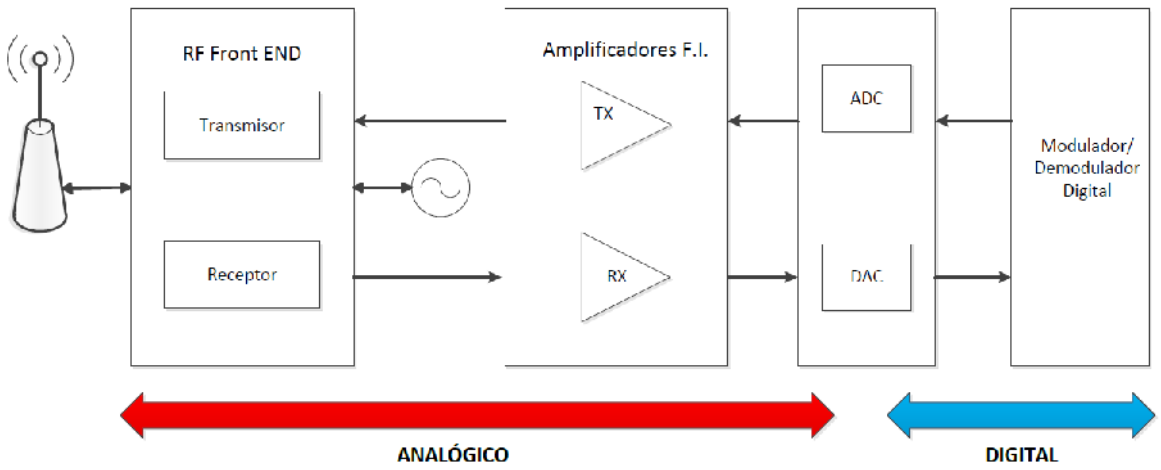

Figura 2.1. Arquitectura *hardware* de SDR.
Fuente: Imagen propia de los autores.

El *hardware* existente dentro del subsistema digital es el encargado de proporcionar toda la flexibilidad y reconfigurabilidad al radio. Normalmente este *hardware* está constituido por dispositivos DSP (*digital signal processor*), pero cada vez son más frecuentes las implementaciones que combinan DSP con FPGA (*field programmable gate array*) y ASIC (*application specific integrated circuit*) para el desarrollo de las diversas funciones que desempeña este subsistema, para las cuales cada tipo de dispositivo ofrece ventajas y desventajas significativas en su elección como plataforma *hardware* de implementación [17].

2.3.2.1 Antena.

La transmisión de la información se realiza mediante ondas electromagnéticas que son transmitidas al espacio, el aire y otros medios no conductores. El elemento que genera estas ondas se denomina antena, puede considerarse como un transductor y un adaptador de impedancia al medio de transmisión.

2.3.2.2 Front end de RF.

Este bloque cuenta con dispositivos electrónicos de estado sólido que adaptan el nivel de las señales de entrada para que sea adecuado en las siguientes etapas del SDR.

En el transmisor, se produce una amplificación de la señal entregada por las etapas de procesamiento hasta el nivel de potencia suficiente para su transmisión por el medio físico.

2.3.2.3 Oscilador local.

Genera las frecuencias apropiadas para convertir la frecuencia de RF en la frecuencia intermedia FI, mediante una mezcla no lineal que produce frecuencias imagen. Se selecciona la frecuencia deseada mediante filtros analógicos para su amplificación en los amplificadores

de frecuencia intermedia correspondientes, algunos llaman a este bloque mezclador/convertidor.

2.3.2.4 Bloque de frecuencia intermedia (FI).

En este bloque se realiza la selectividad y ganancia del receptor, la FI siempre tiene menor frecuencia que la RF debido a que es más fácil y menos costoso fabricar amplificadores estables para señales de baja frecuencia [2]. Por razones similares, también se procesa la señal para la transmisión a una frecuencia inferior para luego convertirla al valor final y amplificarla hasta el nivel permitido en la antena.

2.3.2.5 Conversión AD/DA.

Tomando en cuenta que la transmisión por el medio físico se realiza mediante señales analógicas, pero el procesamiento en el *transceiver* es de índole digital, se hace imprescindible realizar una conversión analógica/digital en el receptor y digital/analógica en el transmisor. A continuación se describen las partes más importantes de este bloque:

- **ADC**: el conversor analógico/digital (ADC) es un dispositivo que es capaz de ofrecer un valor binario de salida a partir de una entrada analógica de voltaje. El dispositivo que realiza el proceso contrario es el conversor digital/analógico (DAC). Esta definición involucra los siguientes procesos:
  - Muestreo: consiste en tomar muestras periódicas de la amplitud de la señal analógica. La velocidad en que se toman las muestras se llama frecuencia de muestreo.
  - Cuantificación: mide el nivel de voltaje de cada muestra y le asigna un valor numérico de salida. Cuando no coincide el valor de salida con el de entrada, se dice que existe ruido de cuantificación.
  - Codificación: la codificación consiste en traducir los valores obtenidos durante la cuantificación en código binario.
- **DDC:** El conversor *digital down converter* se encarga de convertir una señal digital de FI en una señal de banda base. La Figura 2.2 ilustra su composición.

  El DDC se compone de un mezclador digital, un oscilador local digital y un filtro digital pasabajos. El mezclador y el oscilador trasladan las muestras digitales de FI en banda base. El filtro limita el ancho de banda de la señal realizando la función de decimación de muestras a un rango menor de muestreo.

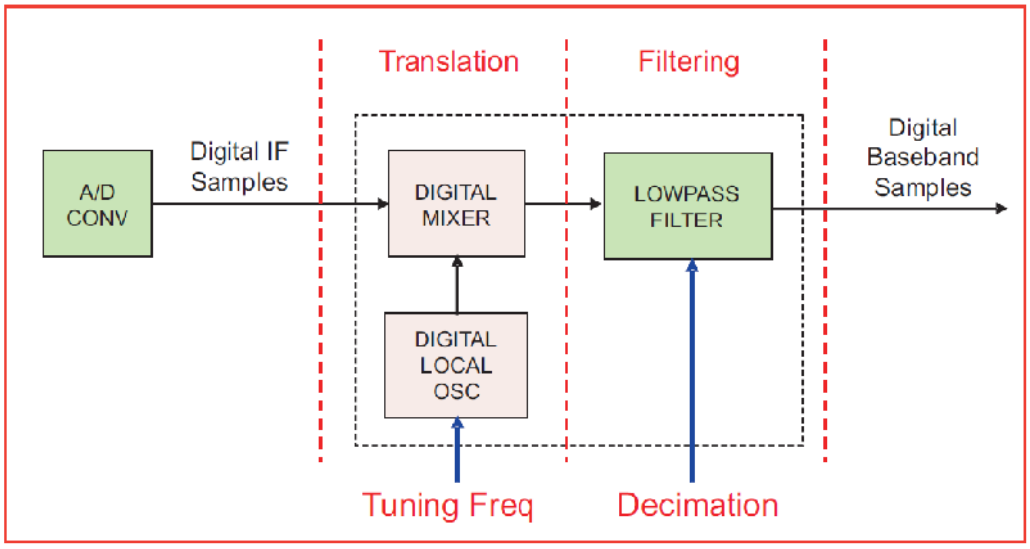

Figura 2.2. *Digital down converter*.
Fuente: Tomada de [1].

- **DUC**: El *digital up converter* es un conversor que traslada la señal de banda base en frecuencia digital intermedia IF. Esta señal es transformada en FI analógica por el convertidor digital analógico (DAC) y esta señal es a su vez convertida en señal RF por el transmisor. El DUC se compone como muestra la Figura 2.3.

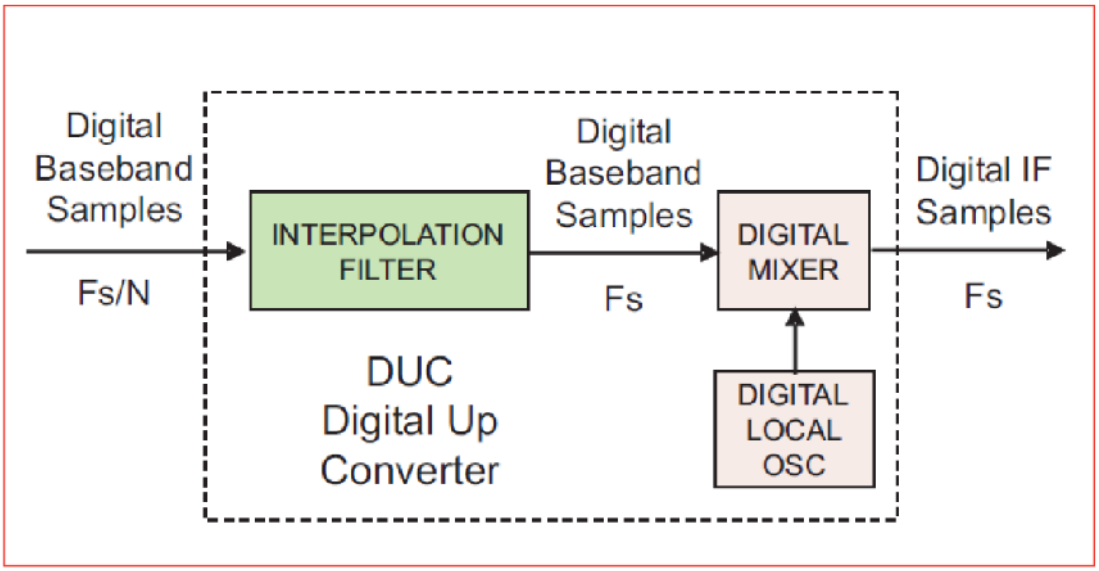

Figura 2.3. *Digital up converter*.
Fuente: Tomada de [1].

El mezclador y el oscilador local trasladan las muestras de banda base a frecuencia FI. El filtro de interpolación resuelve la diferencia entre la frecuencia de muestreo del oscilador (Fs) y la frecuencia de entrada de la señal en banda base (Fs/N).

2.3.2.6 Modulador/Demodulador.

Para que pueda transmitirse una información útil mediante la onda electromagnética que se propaga, es necesario imprimir de algún modo esta información sobre una señal portadora. Esto se logra modificando alguno de los parámetros que la definen de acuerdo con el valor de la información a transmitir. Este proceso se denomina modulación y el proceso inverso para recuperar la información es la demodulación, es por eso que algunos llaman a este bloque modem.

Estas funciones son actualmente muy complejas y totalmente digitales. Generalmente son realizadas por Procesadores de Propósito General (GPP), pero para que las tareas de modulación/demodulación puedan ser fácilmente programables se utilizan procesadores como DSPs o FPGAs.

Los parámetros que se modifican para que la onda transmita información útil son típicamente la frecuencia o la fase/amplitud de la señal, utilizando modulaciones de múltiples niveles denominadas “en cuadratura'' como: mQAM, mPSK, etc. [22].

2.4 Plataformas de *hardware* para SDR

El *hardware* existente dentro del subsistema digital es el encargado de proporcionar toda la flexibilidad y reconfigurabilidad al radio. Normalmente este *hardware* está constituido por dispositivos DSP, pero cada vez son más frecuentes las implementaciones que combinan DSP con FPGA y ASIC para el desarrollo de las diversas funciones que desempeña este subsistema, para las cuales cada tipo de dispositivo ofrece ventajas y desventajas significativas en su elección como plataforma *hardware* de implementación [17].

La configuración y reconfiguración del *hardware* presente en el subsistema digital se realiza utilizando *software*. Este *software* es desarrollado utilizando diversas metodologías y usando herramientas para escritura de código y simulación de sistemas para efectos de pruebas y validación de los componentes desarrollados [17].

Para realizar el procesamiento de las señales basándose en *software* existen diversas alternativas diferenciadas principalmente por dos parámetros como consumo de potencia y costo. Aunque el tamaño de los componentes también puede ser una variable importante a considerar en ciertas aplicaciones. Kenington presenta algunos de los factores más determinantes en el costo de las alternativas *hardware* para SDR [17]:

- El costo directo de los dispositivos en sí mismos.
- El costo asociado a los componentes adicionales y complementarios.

- Costos no recurrentes (asociados más con soluciones basadas en ASIC).
- Inversiones en herramientas y entrenamiento para el desarrollo.
- Costos relacionados con consumo de potencia, refrigeración y tecnologías de alimentación.
- El ciclo de vida del producto SDR y sus diferencias con el ciclo de vida de las aplicaciones/servicios sobre él implementadas.
- Grado de flexibilidad requerido.

De esta manera, los actuales requerimientos de flexibilidad en el procesamiento multimodo y multibanda imponen serios retos a los esquemas de implementación física de SDR, y las cuatro alternativas para ello se resumen en la Tabla 2.1 [17].

Tabla 2.1. Comparación de alternativas para *hardware* SDR.

	Consu-mo de potencia	**Tamaño**	**Costo**	**Campo actualiza-ble**	**Evolu-ción del Chip**	**Herramien-tas**
DSP de alta velocidad	Muy Alta	Moderado	Moderado	Alto	Fácil	Muchas
Múltiples ASIC	Muy Alta	Grande	Alto	Ninguno	Difícil	Muchas
Hardware Parametrizado	Alta	Bajo	Moderado	Medio	Moderado	Algunas
Lógica Reconfigurable (FPGA)	Moderada	Bajo	Bajo	Alto	Fácil	Algunas

Fuente: Tomada de [17].

En las siguientes secciones se presenta una breve descripción de las plataformas de *hardware* más utilizadas para SDR.

2.4.1 SRL QuickSilver QS1R.

Software Radio *Laboratory* LLC es el fabricante del receptor *SRL QuickSilver QS1R* mostrado en la Figura 2.4, cuyas principales características son: posee una resolución de 16bits tipo *Linear Technologies LTC2208*, un conversor analógico/digital (ADC) de 130MSPS y un FPGA *Altera EP3C25 Cyclone III FPGA*. La conexión con el PC se la realiza a través de la interfaz USB 2.0. El receptor QS1R cubre el rango de frecuencias que va desde 10KHz a 62.5MHz en configuración estándar y puede ser usado en aplicaciones hasta 500MHz [2].

2.4.2 WR-G31DDC 'EXCALIBUR'.

WiNRADiO WR-G31DDC '*EXCALIBUR'* mostrado en la Figura 2.5 es un dispositivo SDR de alto rendimiento, bajo costo, muestreo directo, receptor de onda-corta con un rango de frecuencias que va desde 9KHz a 50MHz. Incluye un analizador de espectros en tiempo real

Figura 2.4. Dispositivo SRL QuickSilver QS1R
Fuente: Tomado de [2].

de 50 MHz y 2 MHz de ancho de banda instantáneo disponible para grabar, demodular y el posterior procesamiento digital [3].

Figura 2.5. Dispositivo WR-G31DDC '*EXCALIBUR*'.
Fuente: Tomado de [3].

2.4.3 FiFi SDR.

Mostrado en la Figura 2.6 FiFi SDR es un económico y compacto receptor SDR, basado en el oscilador Si570 que puede ser conectado a la PC por la interface USB. Una característica del dispositivo es la tarjeta de sonido integrada USB, que permite la recepción de todas las transmisiones y bandas de radioaficionados en onda media y corta en todos los tipos de modulación, como la radio DRM. Desde el 2013, el equipo integra un dispositivo de audio USB con una velocidad de muestreo de 192KHz [4].

Figura 2.6. Dispositivo FiFi SDR.
Fuente: Tomado de [4].

2.4.4 BladeRF.

Otra plataforma de *hardware* para SDR es *bladeRF* mostrada en la Figura 2.7, que está diseñada para permitir a una comunidad de aficionados o profesionales explorar y experimentar con los aspectos multidisciplinarios de comunicación RF. Puede sintonizar frecuencias desde 300MHz hasta 3.8GHz, sin necesidad de tarjetas extras. Los controladores de código abierto actuales le permiten soportar el *software* GNU Radio, entre otros. BladeRF actúa como un modem de RF, una picocelda de GSM o LTE, receptor GPS, etc. [5].

El *firmware* del microcontrolador USB 3.0 (*Cypress FX3*) está disponible para ser modificado. Además posee una velocidad de 5Gbps, baja latencia y mayor entrega de potencia por un sólo cable, permitiendo un gran ancho de banda para radio en las computadoras modernas [5].

Figura 2.7. Dispositivo bladeRF.
Fuente: Tomada de [5].

2.4.5 EasySDR.

Mostrado en la Figura 2.8 easySDR es uno de los dispositivos de SDR más pequeños del mercado ya que dimensiones son apenas de 37x22x78mm y peso de 35g, diseñado esencialmente para operar en computadores personales. Este dispositivo se basa en un alto sintonizador multibanda de RF trabajando con entradas de frecuencia configurable entre 64MHz y 1700MHz. La señal amplificada deseada se obtiene por un amplificador de bajo ruido, sintonizado a frecuencia intermedia (IF) y luego muestreada a un conversor analógico/digital de 16bits con 48000muestras/segundo de frecuencia de muestreo. Los datos son muestreados y enviados a la PC a través de la interfaz USB y pueden ser decodificados por el *software* HDSDR [6].

Figura 2.8. Dispositivo easySDR.
Fuente: Tomada de [6].

2.4.6 HackRF.

Mostrado en la Figura 2.9 HackRF es un proyecto de *hardware* abierto para construir periféricos SDR.

HackRF opera en el rango de frecuencias que va desde 30MHz hasta 6GHz, pudiendo ser usado para transmitir o recibir señales de radio. Opera en el modo Half-duplex (es decir se puede transmitir o recibir, pero no puede hacer ambas cosas en el mismo instante de tiempo). El máximo ancho de banda de de HackRF es 20MHz, alrededor de 10 veces el ancho de banda de sintonizadores dungle de TV que son populares para SDR [7]. Este dispositivo puede ser alimentado a través de la interface USB y es lo suficientemente pequeño para llevarlo junto con el computador personal. En la Figura 2.9 se puede apreciar su estructura. HackRF beta está siendo utilizado en las plataformas Linux, OSX y Windows [7].

Figura 2.9. Dispositivo HackRF.
Fuente: Tomado de [7].

2.4.7 USRP N210.

Ettus Research es un proveedor innovador de SDR, incluyendo la familia original de productos USRP. La compañía fue fundada en el 2004 y se encuentra ubicada en Mountain View, California. A partir del 2010 *Ettus Research* en su totalidad propiedad de *National Instruments* [8].

El *hardware* USRP N210 mostrado en la Figura 2.10 permite diseñar e implementar sistemas de radio de *software* flexible siendo ideal para aplicaciones que requieren alto rendimiento en RF y gran ancho de banda. Estas aplicaciones incluyen prototipos de capa física, acceso dinámico de espectro y radio cognitiva, monitoreo del espectro, grabación, reproducción e incluso despliegue del dispositivo conectado a la red [8].

El USRP™ *hardware driver* es el controlador oficial para todos los productos de *Ettus Research* y soporta los sistemas operativos Linux, Mac OSX y Windows [8].

Algunas de sus principales características son las siguientes [8]:

- Se puede usar con GNU Radio, LabVIEW™ y Simulink™.
- Dual 100MS/s, 14-bit ADC.
- Interfaz Gigabit Ethernet para conectar hacia la PC.
- Spartan 3A-DSP 3400 FPGA.

Figura 2.10. Dispositivo de *hardware* USRP N210
Fuente: Tomada de [8].

2.5 Software para SDR

2.5.1 GNU Radio.

Como se aprecia en la Figura 2.11 GNU Radio es un *software* de desarrollo de herramientas de código libre y abierto que proporciona bloques de procesamiento de señales para implementar *software* radio. Este puede ser usado con *hardware* de RF externo de bajo costo para crear radios definidos mediante *software* o sin *hardware* utilizando el entorno de simulación. Es ampliamente utilizado en entornos de aficionados, académicos y comerciales para contribuir tanto en la investigación de comunicaciones inalámbricas y sistemas de radio del mundo real [23].

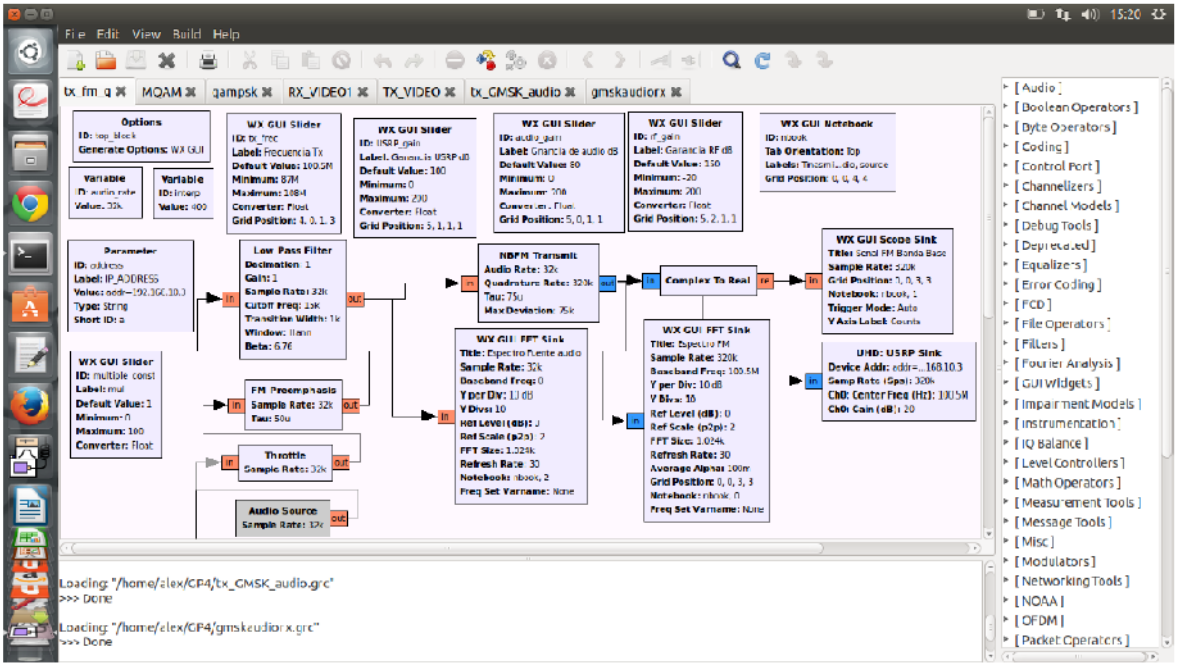

Figura 2.11. Entorno de programación del *software* GNU Radio.
Fuente: Imagen propia de los autores.

Las aplicaciones de GNU Radio son escritas a principio utilizando el lenguaje de programación Python, mientras que el suministro de herramientas críticas de procesamiento de señales que requieren alto rendimiento son implementados en C++ usando extensiones de procesamiento de punto flotante, cuando este está disponible. Así, el desarrollador es capaz de implementar, de manera simple, sistemas de radio de alto rendimiento funcionando a tiempo real aprovechando el ambiente de desarrollo de aplicaciones de manera inmediata [23].

Aunque no es una herramienta principalmente de simulación, GNU Radio complementa el desarrollo de algoritmos de procesamiento de señales a partir de datos previamente grabados o generados, evitando la necesidad de *hardware* de RF [23].

2.5.2 Software de desarrollo de sistemas NI LabVIEW.

Las plataformas SDR (USRP) y LabVIEW cuyo esquema se muestra en la Figura 2.12 ofrecen numerosas aplicaciones que van desde educación hasta investigación. Con soluciones de enseñanza listas para usar, la plataforma SDR soporta las plataformas de RF y telecomunicaciones a través de un aprendizaje práctico con señales del mundo real. La combinación de *hardware* y *software* brinda flexibilidad y funcionalidad, para ofrecer una plataforma de rápida generación de prototipos para diseño de capa física, grabación y reproducción, validación de algoritmos y más [9].

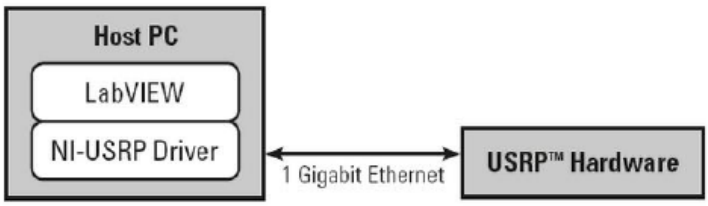

Figura 2.12. NI LabVIEW con la plataforma SDR.
Fuente: Tomado de [9].

Para que sea posible la conexión entre NI USRP y LabVIEW se usa el *driver* NI USRP *software suite DVD*, necesario para realizar las operaciones dentro del *software* y en el que vienen incluidas las siguientes herramientas [9]:

- NI-USRP *software driver*.
- LabVIEW *modulation toolkit*.
- LabVIEW *mathScript RT module*.
- LabVIEW *digital filter design toolkit*.

2.5.3 MATLAB®.

MATLAB® y Simulink® conectados al periférico USRP® de Ettus Research LLC™ proporciona un entorno de diseño y modelado. Con el paquete de ayuda, *Communications System Toolbox*™ y radio USRP®, se puede diseñar y verificar prácticas de los sistemas SDR empezando con la plataforma MATLAB 2011a y Simulink como se observa en la Figura 2.13. Este paquete no requiere del soporte del *software* de código abierto GNU Radio [10].

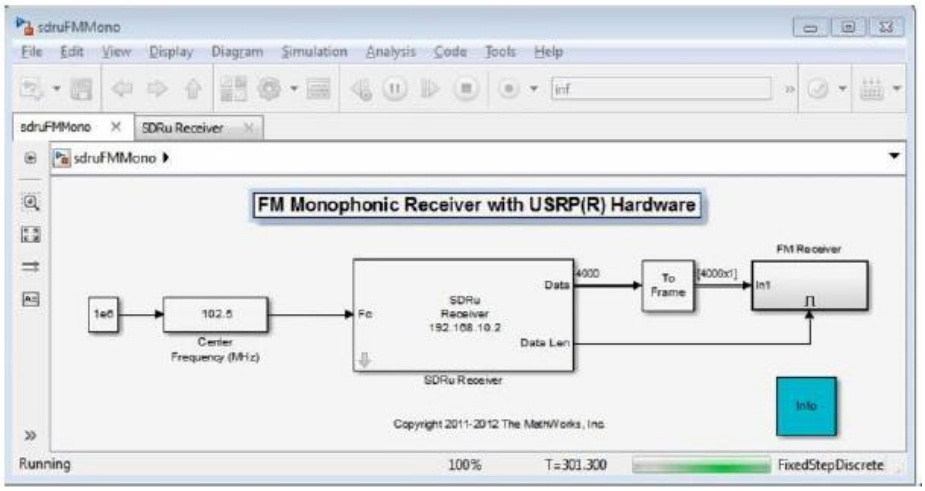

Figura 2.13. MATLAB y Simulink como plataforma *software* de SDR.
Fuente: Tomada de [10].

MATLAB y Simulink soportan los siguientes paquetes de Radio USRP® [10]:

- El uso de USRP® como un periférico independiente para datos de RF en tiempo real de E/S, incluyendo:
  - Funciones y objetos del sistema para conectar MATLAB a UHD™ basado en Radios USRP®.
  - Bloques para conectar Simulink a UHD™ basado en radios USRP®.
- Soporte para generación opcional de código HDL para personalizar el *hardware* USRP® N210, que requiere HDL *Coder*.

MATLAB y Simulink soporta paquetes para radio USRP® que han sido probados en los dispositivos USRP2® N210 usando tarjetas secundarias WBX, SBX, XCVR2450, TVRX, TVRX2, LFRX, LFTX, DBSRX, y DBSRX2. Puede también extenderse a otros radios basados en UHD™ y tarjetas secundarias desde *Ettus Research* LLC™ [10].

2.6 Modulaciones

La necesidad de modular viene dada por la imposibilidad de la propagación de la señal en su banda de frecuencias "base", o en superar las dificultades que representa esta propagación. En general, se pretenden conseguir los siguientes objetivos en el proceso de modulación:

- Posibilidad de multiplexación, es decir, de enviar varios canales de información de una manera conjunta por el mismo medio de transmisión.
- Facilitar la propagación de la señal por el canal de transmisión adaptándola a él. El ejemplo típico es la radiación de señales por ondas de radio que exige utilizar antenas de longitud aproximada $\lambda/2$, donde λ es la longitud de onda de la señal. Para señales de voz limitadas a 4kHz, la longitud de antena a utilizar sería de unos 75Km, totalmente desproporcionada. Entonces, para emitir señales de radio es necesaria una modulación previa para convertir la señal a frecuencias fácilmente radiables.
- Reducción del ruido e interferencia. Empleando el método de modulación adecuado se puede reducir el ruido e interferencias que sufre la señal durante su transmisión, con relación a la transmisión en banda base [24].

En todo proceso de modulación existen una serie de señales propias del proceso se llama moduladora a la señal que contiene toda la información que se quiere enviar. Existe también una señal encargada de “trasladar” al otro extremo de la comunicación esa información que contiene la moduladora. Esta señal que se encarga de llevar la información de la moduladora se denomina portadora. El resultado del proceso será una señal llamada portadora modulada. En general, la modulación va a consistir en la alteración sistemática de algún parámetro de la señal portadora a cargo de la señal moduladora, que es la que originalmente contiene la información [24].

Dependiendo de la naturaleza de la señal de información y de la señal portadora se pueden distinguir varios tipos de modulaciones como se muestra en la Tabla 2.2.

Tabla 2.2. Tipos de modulaciones mono portadoras.

	Moduladora Analógica	**Moduladora Digital**
Portadora Analógica	Modulación Analógica	Modulación Digital
Portadora Digital	Modulación de Pulsos	Códigos de Línea

Fuente: Tomada de [18].

2.6.1 Modulaciones analógicas.

La principal característica de este tipo de modulación reside en que tanto la moduladora como la portadora son analógicas. Dependiendo de cuál sea el parámetro de la señal portadora que

se module por la señal de información, se pueden distinguir algunos tipos de modulaciones analógicas.

2.6.1.1 Modulación de amplitud.

En este tipo de modulación, la amplitud de la portadora varía según la señal de información, de modo que la información de amplitud y frecuencia de ésta se "montan" sobre la portadora haciendo que su envolvente varíe de acuerdo a la señal moduladora o de información. Los diversos esquemas de modulación de amplitud se designan también como de envolvente variable [25] y comprenden los siguientes tipos:

- **AM con portadora completa y dos bandas laterales o AM completa:** la señal de amplitud modulada completa suele expresarse en la forma siguiente:

$$z(t) = V_p \cos(w_p)t + \frac{mV_p}{2}\left[\cos(w_p + w_m)t + \cos(w_p - w_m)t\right]. \quad (2.1)$$

En ella, V_p representa el voltaje de pico de la portadora y m, designado como índice de modulación, está dado por $m = V_m/V_p$ es el voltaje instantáneo de pico de la señal moduladora. El índice de modulación, m, puede tomar valores entre 0 y 1. El primero corresponde a la ausencia de modulación, en tanto que $m = 1$ corresponde al máximo nivel (100%) permisible de modulación [25].

En la Figura 2.14 se observa la señal de amplitud modulada completa con $m = 1$.

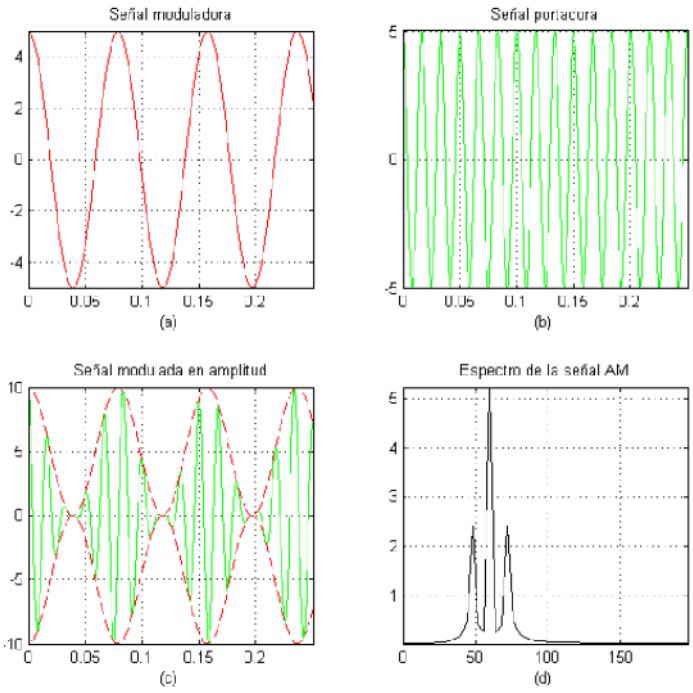

Figura 2.14. Modulación AM con $m = 1$.
Fuente: Imagen propia de los autores.

- **Banda lateral única (BLU o SSB *single side band*) sin portadora**: este método de modulación es utilizado extensamente en los sistemas de comunicaciones siendo el más eficiente de modulación de amplitud, tanto desde el punto de vista espectral como de potencia. Sin embargo el diseño del filtro es complejo. En principio, la potencia máxima necesaria para transmitir en banda lateral es sólo del 16% de la requerida para transmitir con modulación de AM completa [25].

  En aplicaciones en las cuales es importante el ahorro de potencia (como en el caso de sistemas móviles de comunicaciones), en que tanto el peso de los equipos como su consumo de potencia deben ser bajos, así como en sistemas en los que es importante utilizar anchos de banda reducidos, la banda lateral única es la mejor alternativa [25].

- **AM con banda lateral vestigial (AM-VSB v*estigial side band*)**: este tipo de modulación puede considerarse como intermedio entre la modulación de AM completa y la de banda lateral única. Se emplea únicamente en la transmisión analógica de televisión terrestre y por cable y en transmisión digital de televisión en el sistema estadounidense, en que se designa como VSB [25].

2.6.1.2 Modulación angular.

La modulación angular es un tipo de modulación no lineal cuya idea principal es que la señal de información se encuentre en la fase de la portadora, por lo tanto la amplitud de la señal modulada permanece constante, a este tipo de modulaciones se las conoce como modulaciones de envolvente constante [18].

Dentro de las modulaciones angulares se pueden distinguir:

2.6.1.2.1 Modulación de fase (PM).

Es un tipo de modulación angular que se caracteriza porque la fase de la portadora de amplitud constante varía de acuerdo con la señal modulante.

La expresión matemática de la señal portadora, está dada por:

$$y(t) = V_p \cos[w_p + KV_m \cos(w_m)]. \quad (2)$$

Donde $y(t)$ es la señal modulada, V_p la amplitud de la portadora, w_p la frecuencia de la portadora (rad/s), K sensibilidad a la desviación, V_m amplitud de la moduladora, y w_m la frecuencia de la moduladora (rad/s).

La desviación máxima de frecuencia se efectúa durante los cruces de la señal moduladora por cero; es decir, la desviación de frecuencia es proporcional a la pendiente de la primera

derivada de la señal moduladora. Tanto para la modulación de fase como de la frecuencia, la rapidez con que cambia la frecuencia es igual a la frecuencia de la señal moduladora [22].

En la Figura 2.15 podemos observar una señal modulada en fase.

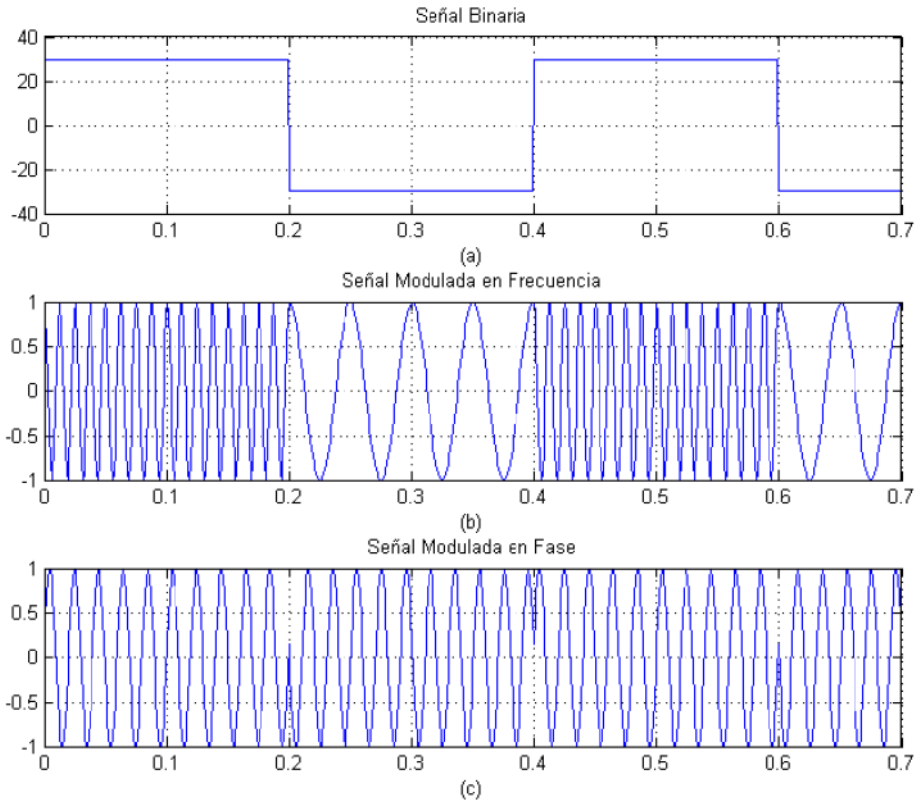

Figura 2.15. Modulación de fase.
Fuente: Imagen propia de los autores.

2.6.1.2.1 Modulación de frecuencia (FM).

En este caso la señal modulada mantendrá fija su amplitud y el parámetro de la señal portadora que variará es la frecuencia, y lo hace de acuerdo a como varíe la amplitud de la señal moduladora [11].

La expresión matemática de la señal portadora, está dada por:

$$V_p(t) = V_p \sin(2\pi f_p t). \quad (2.3)$$

Donde V_p es el valor pico de la señal portadora y f_p es la frecuencia de la señal portadora.

Mientras que la expresión matemática de la señal moduladora está dada por:

$$V_m(t) = V_m \sin(2\pi f_m t). \quad (2.4)$$

De acuerdo a lo dicho anteriormente, la frecuencia f de la señal modulada variará alrededor de la frecuencia de la señal portadora de acuerdo a la siguiente expresión:

$$f = f_p + \Delta f \sin(2\pi f_m t). \tag{2.5}$$

Por lo tanto la expresión matemática de la señal modulada resulta:

$$V_p(t) = V_p \sin\left[2\pi\left(f_p + \Delta f \sin(2\pi f_m t)\right)t\right]. \tag{2.6}$$

Δf se denomina desviación de frecuencia y es el máximo cambio de frecuencia que puede experimentar la frecuencia de la señal portadora. A la variación total de frecuencia desde la más baja hasta la más alta, se la conoce como oscilación de portadora. De esta forma, una señal moduladora que tiene picos positivos y negativos, tal como una señal senoidal pura, provocara una oscilación de portadora igual a 2 veces la desviación de frecuencia [11]

Una señal modulada en frecuencia puede expresarse mediante la siguiente expresión:

$$V(t) = V_p \sin\left(2\pi f_p t + \frac{\Delta f}{\Delta m}\cos(2\pi f_m t)\right). \tag{2.7}$$

Se denomina índice de modulación a:

$$m = \frac{\Delta f}{\Delta m}. \tag{2.8}$$

Se denomina porcentaje de modulación a la razón entre la desviación de frecuencia efectiva respecto de la desviación de frecuencia máxima permisible.

$$PMod = \frac{\Delta f \ efectiva}{\Delta m \ máxima} * 100. \tag{2.9}$$

Al analizar el espectro de frecuencias de una señal modulada en frecuencia, observamos que se tienen infinitas frecuencias laterales, espaciadas en f_m, alrededor de la frecuencia de la señal portadora f_p. Sin embargo, la mayor parte de las frecuencias laterales tienen poca amplitud (las funciones de Bessel explican esta característica), lo que indica que no contienen cantidades significativas de potencia.

El análisis de Fourier indica que el número de frecuencias laterales que contienen cantidades significativas de potencia, depende del índice de modulación de la señal modulada, y por lo tanto el ancho de banda efectivo también dependerá de dicho índice.

Schwartz [26] desarrolló como se muestra en la Figura 2.16 para determinar el ancho de banda necesario para transmitir una señal de frecuencia modulada cuando se conoce el índice de modulación [11].

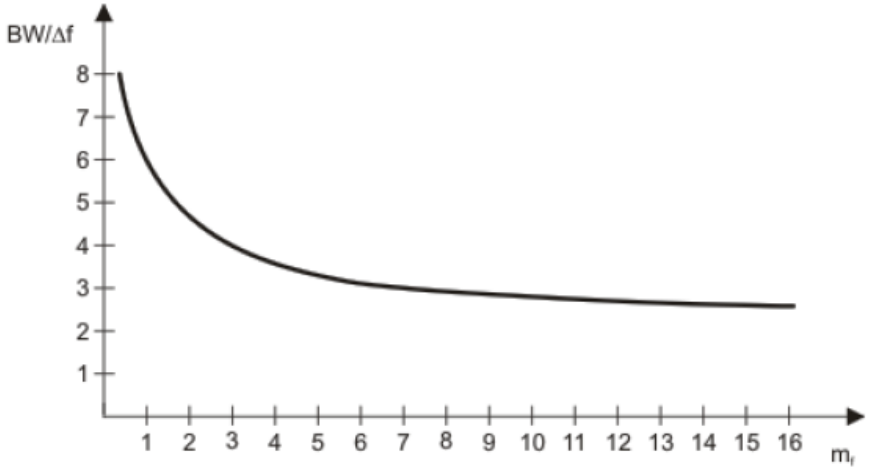

Figura 2.16. Ancho de banda FM necesario en relación a m_f.
Fuente: Imagen propia de los autores.

En la construcción de la gráfica se ha empleado el criterio práctico que establece que una señal de cualquier frecuencia componente, con una magnitud (tensión) menor de 1% del valor de la magnitud de la portadora sin modular, se considera demasiado pequeña como para ser significativa [11].

Al examinar la curva obtenida por Schwartz, se aprecia que para altos valores de m_f, la curva tiende a la asíntota horizontal, mientras que para valores bajos de m_f tiende a la asíntota vertical. Un estudio matemático detallado indica que el ancho de banda necesario para transmitir una señal FM para la cual $m_f < \frac{\pi}{2}$, depende principalmente de la frecuencia de la señal moduladora y es totalmente independiente de la desviación de frecuencia. Un análisis más completo demostraría que el ancho de banda necesario para transmitir una señal de FM, en la cual $m_f < \frac{\pi}{2}$, es igual a dos veces la frecuencia de la señal moduladora [11].

De igual manera que en AM y a diferencia de lo que ocurre para FM con $m_f > \frac{\pi}{2}$, por cada frecuencia moduladora aparecen dos frecuencias laterales, una inferior y otra superior, a cada lado de la frecuencia de la señal portadora y separadas en f_m de la frecuencia de la portadora cuando $m_f < \frac{\pi}{2}$ se la denomina FM de banda angosta, mientras que las señales de FM donde $m_f > \frac{\pi}{2}$, se las denomina FM de banda ancha.

Los espectros de frecuencia de AM y de FM de banda angosta, aunque pudieran parecer iguales, por medio del análisis de Fourier se demuestra que las relaciones de magnitud y fase en AM y FM son totalmente diferentes.

En FM el contenido de potencia de la señal portadora disminuye conforme aumenta m_f, con lo que se logra poner la máxima potencia en donde está la información, es decir en las bandas laterales [18].

2.6.2 Modulaciones digitales.

La principal característica de este tipo de modulación reside en que a diferencia de las modulaciones conocidas como analógicas, la fuente de información ahora es digital. La señal de información será por lo tanto un flujo de bits. Dependiendo del número de bits que se agrupen se formará una modulación de M niveles, donde $M = 2^k$ y k los bits agrupados. Por lo tanto, las señales moduladas, estarán contenidas en un conjunto finito conocido como el alfabeto de señales o símbolos.

Un concepto que cobra gran importancia en este tipo de modulaciones es el concepto de constelación, la constelación es la representación del alfabeto de señales en un sistema de L dimensiones. Esto se debe a que el conjunto de señales forman un espacio vectorial, por ende se podrá buscar una base orto normal que represente dicho espacio [18].

En los sistemas actuales la señal se separa en un conjunto de dos señales componentes que son independientes, que se conocen como: componente I (*in phase*, en fase) y componente Q (*quadrature*, cuadratura). Ambas componentes son ortogonales y no interfieren una con otra [12].

En las comunicaciones digitales, la modulación es a menudo expresada en términos de las componentes I y Q. Las cuáles son las componentes rectangulares del diagrama polar como se muestra en la Figura 2.17.

Las razones más comunes para utilizar la modulación digital para transmitir señales analógicas son: proveen mayor eficiencia en el ancho de banda y mayor calidad en transmisión. Bajo condiciones ideales la transmisión analógica es de mejor calidad que la digital, sin embargo cuando las condiciones de propagación se deterioran la calidad de la transmisión digital permanece mientras que la calidad de transmisión analógica empeora rápidamente [12].

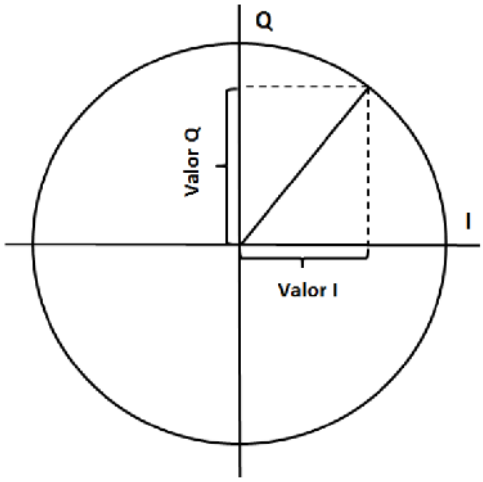

Figura 2.17. Diagrama de sistemas de modulación I/Q.
Fuente: Tomada de [12].

2.6.2.1 Modulación (PSK).

Phase Shift Keying (PSK) es una técnica de modulación digital que consiste en asignar una fase distinta a cada uno de los símbolos de la fuente, manteniendo constante la amplitud y la frecuencia y su representación es:

$$S_{PSK}(t) = \cos\left[w_{pt} + \frac{g_L(t)S\phi}{2}\right]. \tag{2.10}$$

Dónde $S\phi$ es la separación entre fases adyacentes, la misma que es igual a $S\phi = \frac{2\pi}{M}$ y M es el número de niveles de la modulación.

2.6.2.1.1 Modulación por desplazamiento de fase binaria (BPSK).

En esta técnica de modulación son posibles dos fases de salida para una sola frecuencia de portadora. Una fase de salida representa un 1 lógico y la otra un 0 lógico. Conforme la señal digital de entrada cambia de estado, la fase de la portadora de salida se desplaza entre dos ángulos que están desfasados 180° [27] como se muestra en la Figura 2.18.

Entonces se representa como:

$$S_{PSK}(t) = \cos\frac{g_L(t)S\phi}{2}\cos w_{pt} - \sin\frac{g_L(t)S\phi}{2}\sin w_{pt}\ , \tag{2.11}$$

$$S_{PSK}(t) = \cos\phi L\ \cos w_{pt} - \sin\phi L\ \ \sin w_{pt}\ , \tag{2.12}$$

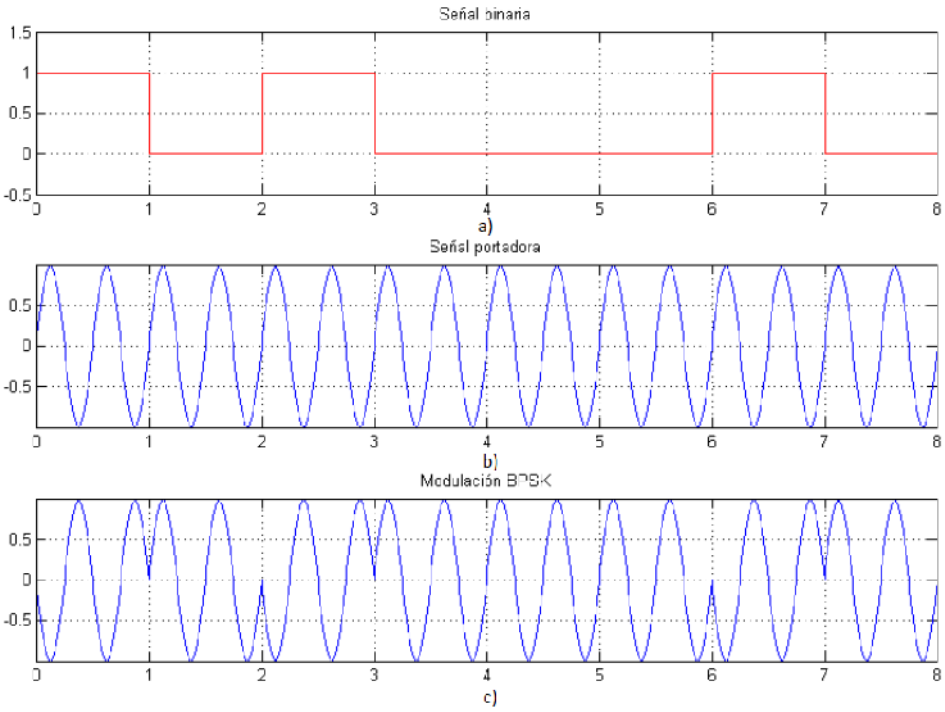

Figura 2.18. Modulación BPSK a) señal binaria, b) señal portadora, c) modulación con BPSK
Fuente: Imagen propia de los autores.

donde

$$\phi L = \frac{g_L(t) S\phi}{2}, \tag{2.13}$$

$$M_I(t) = \cos \phi L \qquad M_Q(t) = -\sin \phi L\,. \tag{2.14}$$

Finalmente la ecuación quedaría:

$$S_{PSK}(t) = M_I(t) \cos w_{pt} + M_Q(t) \sin w_{pt}\,. \tag{2.15}$$

En la Tabla 2.3 se resume las características de BPSK.

Tabla 2.3 Características de modulación BPSK.

BPSK	$M_I(t)$	$M_Q(t)$	$S_{PSK}(t)$
0	1	0	$\cos w_{pt}$
1	0	1	$-\sin w_{pt}$

Fuente: Tabla propia de los autores.

2.6.2.1.2 Modulación (QPSK).

La técnica de modulación por desplazamiento de fase cuaternaria (QPSK) o en cuadratura, es otra forma de modulación digital de modulación angular de amplitud constante. QPSK es una técnica de codificación M-ario, en donde el número de niveles $M = 4$ para una sola

frecuencia portadora. Existen cuatro posibles condiciones: 00, 01, 10 y 11, separados 90° o $\pi/2$ que se combinan en grupos de 2 bits llamados dibits. Cada código dibit genera una de las cuatro fases [12].

$$S_{QPSK}(t) = \cos\left[w_{pt} + \frac{g_L(t)S\phi}{2}\right]. \tag{2.16}$$

Donde $S\phi$ es la separación entre fases adyacentes, la misma que es igual a $S\phi = \frac{\pi 2}{M}$ y M es el número de niveles de la modulación que para QPSK son 4.

Entonces tenemos:

$$S_{PSK}(t) = \cos\frac{g_L(t)S\phi}{4}\cos w_{pt} - \sin\frac{g_L(t)S\phi}{4}\sin w_{pt}\ . \tag{2.17}$$

Y como en el caso de la modulación BPSK, se procedió a sintetizar la ecuación y esta será la misma para QPSK, pero cambiando los valores de los componentes en fase y cuadratura, en la Tabla 2.4 se resumen algunas características.

Tabla 2.4. Características de modulación QPSK.

QPSK	$M_I(t)$	$M_Q(t)$
01	$\frac{1}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}$
00	$-\frac{1}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}$
10	$-\frac{1}{\sqrt{2}}$	$\frac{1}{\sqrt{2}}$
11	$\frac{1}{\sqrt{2}}$	$\frac{1}{\sqrt{2}}$

Fuente: Tabla propia de los autores.

2.6.2.2 Modulación QAM.

Quadrature amplitude modulation es una técnica de modulación digital en la que la información va a ser modulada tanto en amplitud como en fase. QAM va a ser un tipo de modulación M-aria en la que para grupos de k bits, podemos obtener $M = 2^k$ salidas diferentes

La modulación cuenta con una constelación con un entramado cuadrado de puntos de señales. La forma general de una modulación M-QAM se representa para un intervalo de $0 \leq t \leq T_s$ y $i = 4,8,16 \ldots M$ por [12].

$$S_{QAM}(t) = \sqrt{\frac{2E_{min}}{T_s}}\ a_i \cos w_{pt} - \sqrt{\frac{2E_{min}}{T_s}}\ b_i \sin w_{pt}\ . \tag{2.18}$$

Donde E_{min} es la energía de la señal con una amplitud mínima y a_i, b_i son un par de coordenadas independientes de acuerdo a la localización de la señal especifica [12].

En la Figura 2.19 se muestra el diagrama de constelación de una señal modulada en QAM.

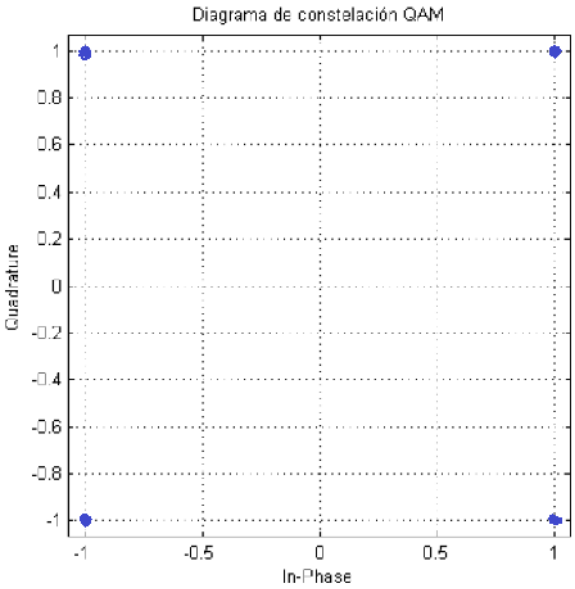

Figura 2.19. Diagrama de constelación QAM.
Fuente: Imagen propia de los autores.

2.6.2.3 Modulación GMSK.

Gaussian Minimun Shift Keying es un esquema de modulación continua en fase. Se trata de una técnica que consigue suavizar las transiciones de fase entre estados de la señal, consiguiendo por lo tanto reducir los requerimientos de ancho de banda. Con GMSK, los bits de entrada representados de forma rectangular (+1;-1) son transformados en pulsos Gaussianos mediante un filtro Gaussiano, para posteriormente ser suavizados por un modulador de frecuencia [28]. En la figura 2.20, es posible apreciar como la fase se suaviza debido a la introducción del filtro Gaussiano, al hacer pasar la secuencia de datos 0100 por el modulador.

En la mayoría de los casos, la duración del pulso Gaussiano supera la de un bit, dando lugar a lo que se conoce como interferencia inter-simbólica (ISI). El grado de esta superposición es determinado por el producto del ancho de banda del filtro Gaussiano y la duración de un bit. Este producto se conoce normalmente como BT.

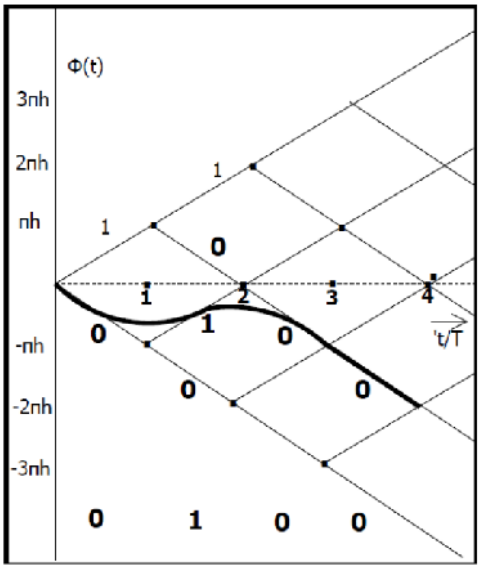

Figura 2.20. Diagrama de fase para GMSK.
Fuente: Tomada de [13].

Cuanto menor sea el valor de BT mayor será el solapamiento entre pulsos Gaussianos. La portadora resultante es una señal continua en fase, lo cual es importante porque las señales con transiciones suaves entre fases requieren menor ancho de banda para ser transmitidas. Básicamente, en GMSK, los lóbulos laterales del espectro se reducen al pasar la señal codificada en NRZ (Codificación digital polar sin retorno a cero) a través de un filtro conformador de pulso Gaussiano. Por otra parte, este suavizado de la señal hace que el receptor tenga que realizar un trabajo mayor en la demodulación de la señal, ya que las transiciones entre bits no están bien definidas [28].

La señal final GMSK se puede representar a través de una expresión matemática. Esta función está expresada en:

$$S_t = \sqrt{2E_b t}\ \cos(2\pi f_c t + \theta(t) + Z_0). \tag{2.19}$$

Donde $\mathrm{E_b}$ es la energía de señal por bit y $\mathrm{Z_0}$ es una constante de fase aleatoria que puede ser asumida como cero ($\mathrm{Z_0} = 0$).

La fase de la señal modulada está dada por:

$$\theta(t) = \Sigma m_i \pi h \int_{-\bowtie}^{t-iT} g(u)du. \tag{2.20}$$

Donde m_i es la señal de datos NRZ (*non return zero*), la cual tiene como valores 1 y -1. El índice de modulación $h = 0.5$ resulta en el máximo cambio de fase de $\pi/2$ radianes por cada intervalo de datos.

CAPÍTULO III

MATERIALES Y MÉTODOS

En este capítulo se detallan los equipos y elementos tanto de *hardware* como de *software* empleados en el presente trabajo de investigación.

3.1 Universal Software Radio Peripheral (USRP)

Se trata de un equipo SDR diseñado por la empresa *ETTUS Research* como propuesta de *hardware* libre, donde los microprocesadores convencionales pueden actuar como dispositivos de radio bajo un gran ancho de banda, convirtiéndose en una plataforma flexible de bajo costo que permite implementar y diseñar potentes sistemas de radiocomunicaciones con aplicaciones en tiempo real [29]. En esencia, sirve como procesador digital de banda base y conversor de frecuencia intermedia FI en un sistema de radiocomunicación.

3.1.1 Hardware del USRP N210.

Los sistemas USRP están integrados por una tarjeta principal y una o varias tarjetas secundarias para transmisión o recepción que se diferencian entre sí por sus rangos de frecuencia, potencias de salida y ancho de banda. En el caso del USRP N210, su arquitectura emplea una FPGA, Spartan 3A DSP 3400, con un conversor A/D dual de 100MS/s con 14 bits y un conversor D/A dual de 400MS/s con 16 bits. Utiliza memoria SRAM de alta velocidad de 1 MB y conectividad Gigabit Ethernet que permite enviar al computador hasta 50MS/s [30].

Figura 3.1. USRP N210.
Fuente: Imagen propia de los autores.

El equipo incluye además 2 cables de RF con conectores SMA, una fuente de poder independiente, un cable Ethernet y componentes para el ensamble del equipo. Es necesario que el computador al cual se desea conectar el USRP posea una tarjeta PCI Gigabit Ethernet, que permite la conexión con la interfaz de red del USRP.

En la Figura 3.2 podemos observar su arquitectura de *hardware*.

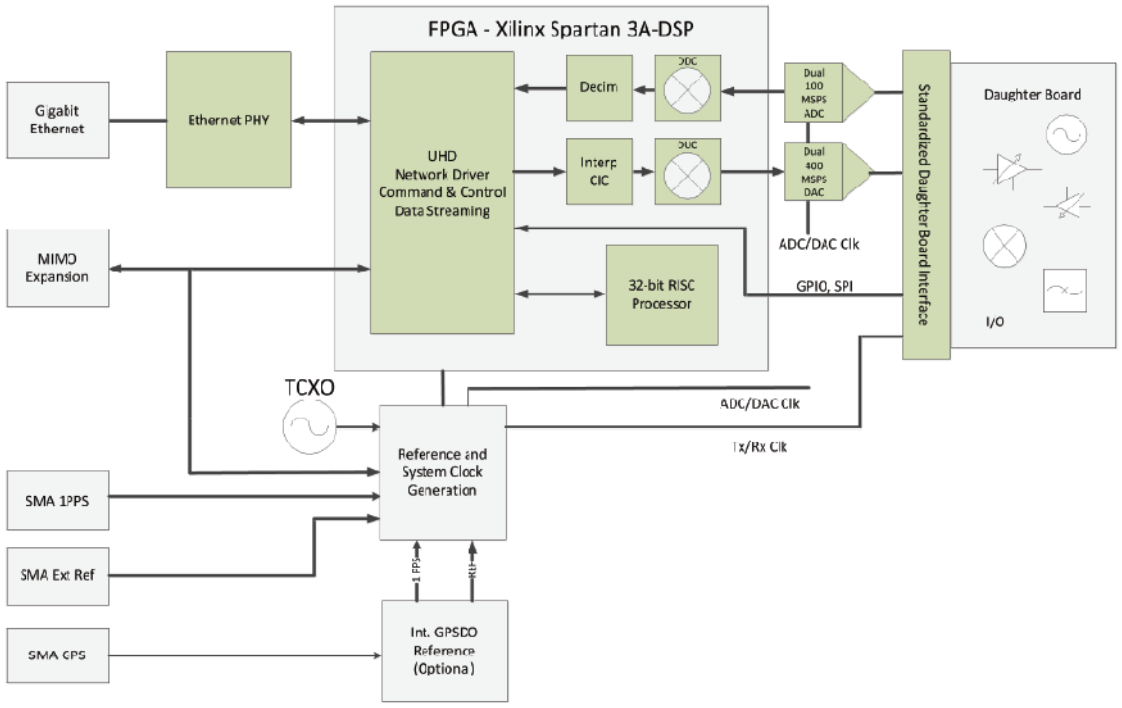

Figura 3.2. Arquitectura *hardware* USRP N210.
Fuente: Tomada de [8].

3.1.1.1 Tarjeta principal.

Esta placa se encarga de comunicar la señal generada vía *software* desde el *host* hacia el módulo de RF, el cuál realizará lo necesario a la señal para disponer de ésta a la frecuencia requerida para la aplicación a desarrollar. La Figura 3.3 muestra esquemáticamente las funciones a realizar por la placa principal.

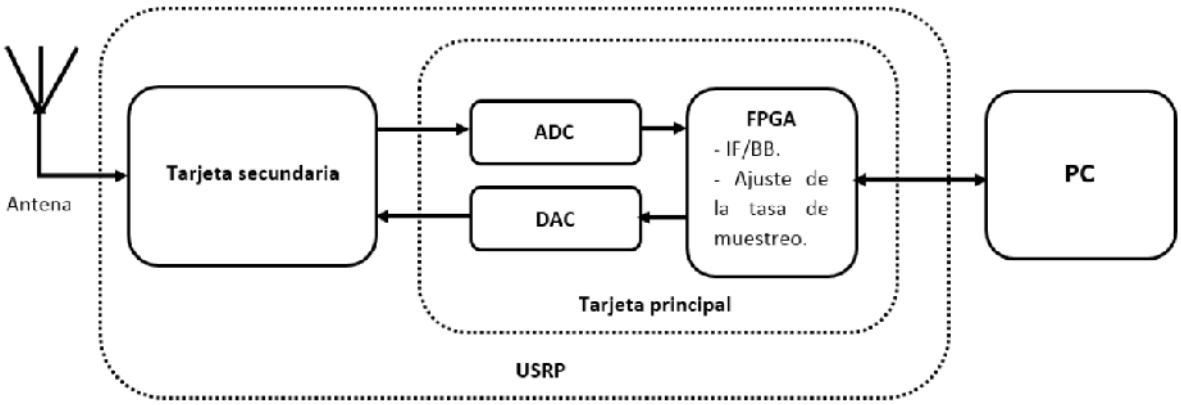

Figura 3.3. Funciones de la tarjeta principal.
Fuente: Tomada de [18].

Su función comienza a la salida del PC y acaba cuando la señal atraviesa el DAC en el caso del transmisor, mientras que cuando se emplea como receptor su función empieza cuando la señal sale de la tarjeta secundaria y termina cuando la señal es conducida hacia el PC, encargándose de la conversión de la señal desde el dominio digital al analógico o del analógico al digital dependiendo de cuál sea el caso y adecuando la tasa de muestreo de la señal para su transmisión.

En el caso de que se desee transmitir una señal, esta será generada vía *software* y será enviada mediante Gigabit Ethernet hacia el USRP entrelazando las componentes de fase y cuadratura de la señal, una vez la señal llega al puerto Ethernet del dispositivo es conducida hacia la FPGA, esta es la encargada de desentrelazar las componentes de la señal y de realizar tanto operaciones sobre el ancho de banda que se requiera mediante el uso de filtros interpoladores como desplazamientos en frecuencia de la señal haciendo uso de los DUC. Finalmente la señal a la salida de la FPGA es conducida hacia el DAC de doble canal (uno para cada componente), el cual convertirá las componentes al dominio analógico para su transmisión y las enviará hacia la tarjeta secundaria [18].

Para el caso del receptor, llegan al ADC las componentes de la señal entregada por la placa secundaria, siendo convertidas al dominio digital y entregadas a la FPGA, quien será la encargada de desplazar estas en frecuencia mediante el DDC, de realizar operaciones sobre el ancho de banda la señal mediante filtros diezmadores y de entrelazar las componentes I y Q de la señal para su transmisión a través de Ethernet.

El USRP N210 (tarjeta principal + tarjeta secundaria) está diseñado para que el DUC lleve a cabo el traslado de frecuencia desde BB/IF y posteriormente la tarjeta secundaria llevará a cabo el traslado IF/RF. Sin embargo, muchas tarjetas secundarias están diseñadas para operar en conversión directa eliminando así la necesidad de hacer uso del DUC, por lo que solo se utilizará éste cuando la tarjeta secundaria no pueda sintetizar la frecuencia requerida por el usuario llevando a cabo un ajuste más fino. En el caso del receptor sucede lo mismo que en el transmisor [18].

Los filtros diezmadores o bien interpoladores se utilizan para acoplar las distintas tasas binarias entre el enlace Gigabit Ethernet y el requerido por la aplicación.

En cuanto a la señal de reloj a utilizar, esta puede ser bien externa (SMA 1PPS, SMA Ext Ref, SMA GPS) o bien interna. La señal de reloj, por motivos de optimizar la sincronización del dispositivo, será utilizada tanto en la FPGA, en los como conversores ADC, DAC y en la tarjeta secundaria

3.1.1.2 Tarjeta secundaria.

Son las encargadas de cumplir gran parte de las funciones del transmisor y/o del receptor de radiofrecuencia clásico. La Figura 3.4 muestra las funciones a realizar por la placa secundaria.

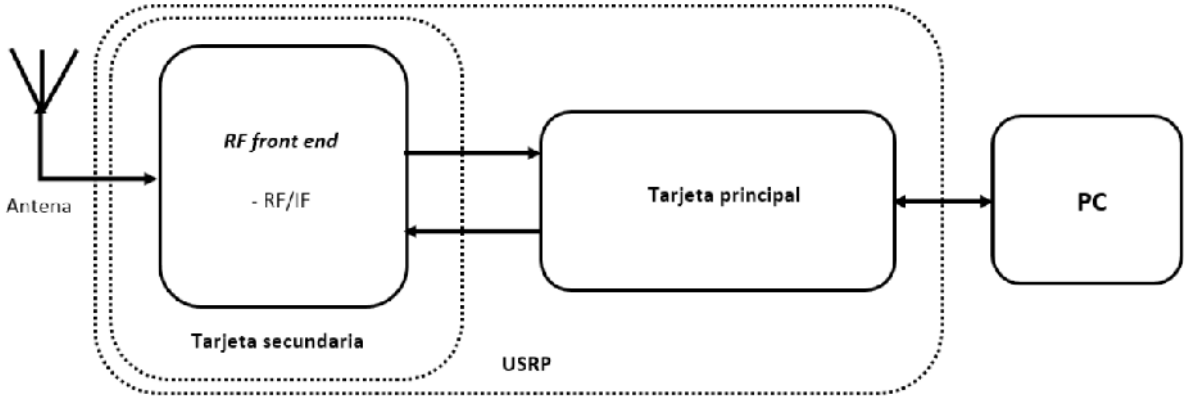

Figura 3.4. Funciones de la tarjeta secundaria.
Fuente: Tomada de [18].

Su función empieza a la salida del DAC y termina cuando es conducida la señal a transmitir hasta el conector SMA, en el caso del transmisor. Por el contrario, en el caso del receptor, empieza cuando la señal llega al conector SMA y finaliza cuando ésta es conducida hacia el ADC, por lo tanto este tipo de placas se conecta físicamente a la tarjeta principal.

En la placa principal existen dos ranuras etiquetadas como TX y RX, cada ranura tiene acceso a un DAC (para el transmisor) y a un ADC (para el receptor), éstos son de doble canal, permitiendo así la conversión de manera independiente para la componente en fase y en cuadratura de la señal para las placas que lo permitan o para dos canales independientes.

Cada placa secundaria cuenta con una *electrically eraseable programmable read-only memory* (EEPROM), la cual aparte de servir como identificador para el sistema, permite establecer al *host* la configuración adecuada para su uso correcto (por ejemplo los distintos rangos de ganancia permitidos) [18].

El USRP N210 cuenta con un PGA (*programable gain amplifier* - amplificador de potencia programable) antes de los ADCs para amplificar la señal de entrada y utilizar el rango completo en caso de que la señal sea débil. En la parte de transmisión el USRP tiene dos DACs de alta velocidad a 16 bits por muestra y una tasa de 400 mega-muestras por segundo (400MSPS), contando de igual forma con un PGA después de los DACs que proporciona hasta 10mW de ganancia [31].

Estos canales de entrada y salida son conectados a la FPGA 3A-DSP 3400, en la cual se conecta un chip Gigabit Ethernet que sirve como interfaz de conexión al computador con un ancho de banda de 50 MHz usando una cuantización de 8 bits [18].

La FPGA realiza operaciones matemáticas de alto ancho de banda y reduce la tasa de datos para que puedan ser enviados a través de la interfaz Gigabit-Ethernet hasta el computador. En el N210, el procesamiento con alta frecuencia de muestreo se realiza en la FPGA, mientras el procesamiento con baja frecuencia de muestreo se realiza en el computador.

La configuración básica de la FPGA incluye dos DDCs completos, pero también es posible la implementación de 4 DDCs sin filtros de media banda. Esto permite tener 1, 2 o 4 canales de recepción separados. Las salidas de los ADCs van conectadas a las entradas de los DDC. Estos mezclan, filtran y diezman las señales de entrada en la FPGA. Se utilizan en la recepción, esencialmente por dos razones:

- Para convertir la señal en banda de frecuencia FI a una señal en banda base.
- Para diezmar la señal, logrando que la tasa de datos pueda ser adaptada a la interfaz Gigabit Ethernet y que sea acorde a la capacidad de procesamiento del computador.

En la transmisión se realiza el proceso inverso, donde es necesario convertir una señal banda base a una señal de frecuencia intermedia y enviarla a través de los DACs. Este proceso lo realizan los DUC. En la transmisión se usan filtros interpoladores CIC *(cascaded intengrator-comb*) que interpolan las muestras antes de trasladar la señal digital a la frecuencia intermedia por el DUC [18].

Los DDC y DUC combinados con altas tasas de muestreo simplifican en gran medida los requerimientos de filtrado analógico.

Adicionalmente se debe tener una tarjeta con soporte de RF y dos antenas que constituyan el *Front-End* del USRP. En este trabajo se utiliza la tarjeta transmisora y receptora WBX, la misma que es un transceptor de banda ancha con osciladores independientes para el transmisor y el receptor permitiendo así operar en modo *full-duplex* con señales complejas. Por defecto, esta placa opera convirtiendo la señal de RF directamente a BB.

El transmisor presenta una potencia máxima de salida de 100mW y un ancho de banda de 40MHz tanto para el transmisor como para el receptor. Dicha placa está diseñada también para soportar MIMO *(multiple input multiple output)*. El rango de operación de la placa es de 50MHz-2200MHz presentando ganancias variables tanto en la cadena del transmisor como en la del receptor, como se puede apreciar en la Tabla 3.1. En el caso del transmisor la

ganancia variable va desde 0dB hasta los 25dB mientras que en el receptor va desde 0 dB hasta los 31.5 dB. Su figura de ruido va desde 5dB hasta 10dB [18].

Mostrada en la Figura 3.5 la placa WBX tiene dos conectores SMA, uno de ellos se puede configurar bien como transmisor o bien como receptor (TX/RX), mientras que el otro puerto solo puede actuar como receptor (RX2) [14].

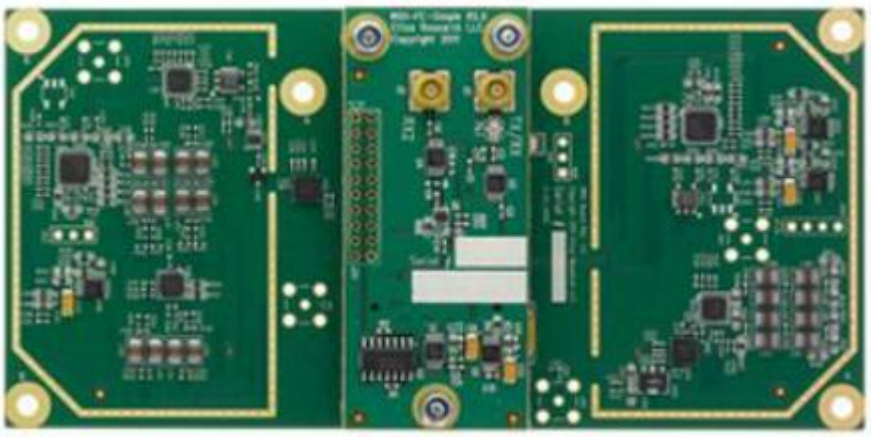

Figura 3.5. Tarjeta secundaria WBX.
Fuente: Tomada de [14].

Tabla 3.1. Características de tarjetas RF secundarias WBX y FRX900.

Tarjetas RF	WBX	RFX900
Rango de Frecuencia	50MHz a 2.2GHz	750 MHz a 1050MHz
Potencia de transmisión	100mW (20dBm)	200mW (23dBm)

Fuente: Tomada de [14].

El módulo de transmisión y recepción utiliza dos antenas VERT900, la misma que se puede observar en la figura 3.6 y poseen las características listadas en la Tabla 3.2. Su ganancia no supera los 3 dBi por lo que su alcance máximo será 200 mts, si se despliegan en un espacio libre.

Figura 3.6. Antena VERT900.
Fuente: Tomada de [8].

Tabla 3.2. Características de la antena VERT900.

Parámetros	Características
Ganancia	3dBi
Frecuencia	824MHz a 960MHz, 1710MHz a 1990MHz
Banda	Cuatribanda celular /PCS
Compatibilidad	Tarjetas secundarias WBX, RFX900, RFX1800

Fuente: Tomada de [8].

3.1.1.3 Panel frontal.

En la parte frontal del equipo N210 se han colocado 6 leds enumerados y de color verde, que indican un determinado estado del dispositivo como se observa en la Figura 3.7.

Figura 3.7. Panel frontal del equipo USRP N200/210.
Fuente: Imagen de los autores.

Según las letras que se han asignado a los leds pueden indicar los siguientes estados si están encendidos:

- **Led A:** el equipo está transmitiendo Tx.
- **Led B:** indica si está conectado el cable tipo MIMO.
- **Led C:** recepción.
- **Led D:** el *firmware* ha sido cargado.
- **Led E:** reloj de referencia.
- **Led F: CPLD** (*complex programmable logic device*) está activo.

En el panel también se encuentran los puertos tanto de la interfaz Gigabit Ethernet que serán conectados con el computador. También el puerto de expansión MIMO utilizado para realizar una conexión en paralelo con otros USRPs si es necesario para sincronizarlos con el mismo reloj.

Asimismo, tenemos los puertos para conectar las antenas de transmisión y recepción. Estos puertos están etiquetados como “RF1” para transmisión y “RF2” para recepción. Y por último están los puertos “REF clock” y “PPS in” que sirven para conectar un reloj de referencia en el primero o un multivibrador [32] en el segundo, ambos con la finalidad de realizar la sincronización necesaria para las aplicaciones que se desarrollen. La Tabla 3.3 indica los parámetros permitidos de reloj o multivibrador.

Tabla 3.3. Valores de entrada permitidos en el panel frontal.

Puerto	**Tolerancia de entrada**
Ref *clock*	Max 10MHz Reference *clock* con 0 - 15dBm nivel de potencia.
PPS in	3.3 a 5Vpp (voltaje pico-pico).

Fuente: Tomada de [32].

3.1.1.4 GPSDO.

El dispositivo *global positioning system disciplined oscillator* (GPSDO), permite una sincronización precisa de muestras para transmisión Tx y recepción Rx en el USRP, generando una frecuencia de reloj de 10MHz en su oscilador de salida con una precisión de 0.01PPM (Partes por Millón), acercándose a la frecuencia de 13MHz con 0.02PPM requerido en la mayoría de las implementaciones GSM tanto para la generación de frecuencia como para la sincronización del reloj [33].

El USRP N210 tiene incorporado en su tarjeta principal un oscilador de cristal de temperatura compensada (TCXO) con precisión de 2.5PPM, muy baja para trabajar como reloj principal en la red. Por lo que se recomienda usar la señal del GPSDO como entrada de reloj de referencia REF en la tarjeta principal del USRP. Los pasos para la conexión del GPSDO con el equipo USRP se pueden encontrar en la guía desarrollada por el fabricante [34].

3.1.2 UHD.

UHD *(USRP hardware driver)* se trata del *software* controlador o *driver*, que se debe instalar en el ordenador para que pueda interactuar con el radio USRP, desarrollado por *Ettus Research* para el desarrollo de aplicaciones con los equipos de la familia USRP. Puede trabajar en los sistemas operativos Linux, Windows, y Mac.

Posibilita una Interfaz de Programación de Aplicaciones (API) para la investigación de nuevos servicios que se pueden adaptar a los radios basados en SDR. Los usuarios de UHD pueden realizar la comunicación del *hardware* URSP bajo plataformas de desarrollo de *software* como: GNU Radio, LabVIEW, MATLAB, OpenBTS.

3.2 GNU Radio

El control y adquisición de datos desde el USRP se inició como un proyecto denominado *software* de código abierto (OSS *open source software*) conocido como GNU Radio, fundado por Eric Blossom [15] en 1998 con el objetivo de desarrollar un marco de trabajo para *software* radio. Se trata de una herramienta de *software* libre y de código abierto, constituida por un conjunto de archivos y librerías que proporcionan bloques de procesamiento de señales, permitiendo así el diseño y la simulación de sistemas basados en *software* radio [15].

Dentro de GNU Radio existen dos niveles que un programador puede observar. Bloques de código de bajo nivel que son escritos en C++ para eficiencia, y consiste de componentes de procesamiento de señal pequeños como podemos apreciar en la Figura 3.8 [15]. Los códigos de alto nivel están escritos en Python que principalmente su conexión es de varios bloques de procesamiento de señales juntos en un solo grafo dirigido. Como Python es un lenguaje interpretado no requiere tiempo adicional de compilación durante desarrollo o experimentación; y esto puede ser usado en beneficio del desarrollador para el despliegue rápido de aplicaciones RAD (*rapid application deployment*) [15].

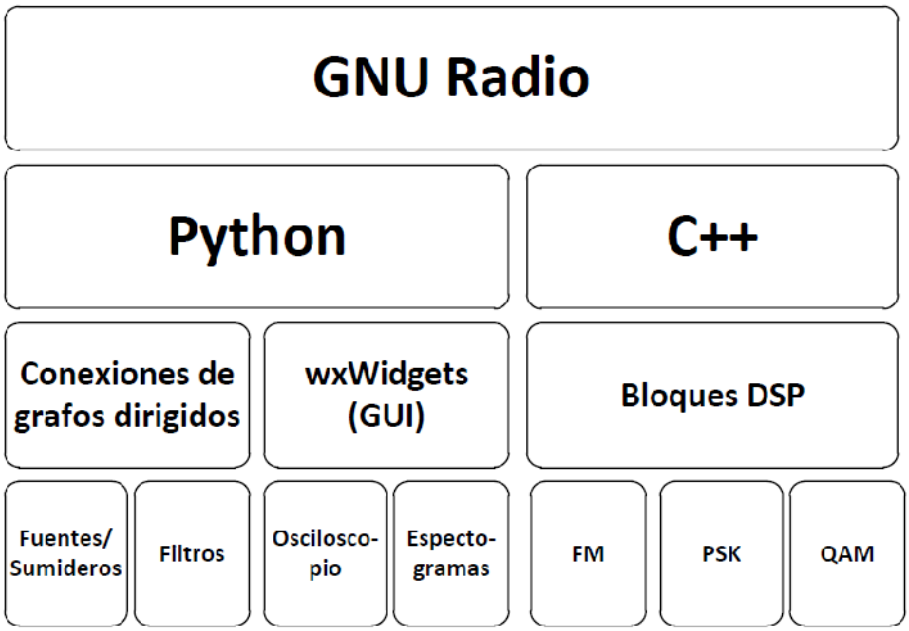

Figura 3.8. Arquitectura de GNU Radio.
Fuente: Tomada de [15].

Aunque GNU Radio incluye diversas y variadas funciones para realizar proyectos SDR, también permite al usuario incluir procesos no desarrollados o contemplados previamente en el proyecto mediante una serie de lineamientos bien establecidos. En la Figura 3.9 se observa la manera como GNU Radio, siendo una biblioteca de lenguaje de programación Python,

puede interactuar con bibliotecas de lenguaje C++ por medio de la herramienta de programación SWIG (*simplified wrapper and interface generator* - envoltura simplificada y generador de interfaz). Este programa funciona como un intérprete de lenguaje C++ a lenguaje Python, además a través de la herramienta CMAKE (*cross platform MAKE* - compilador multiplataforma), permite construir, probar y empaquetar códigos de programación. GNU Radio puede integrar y compilar bibliotecas para C++ no definidas previamente [16].

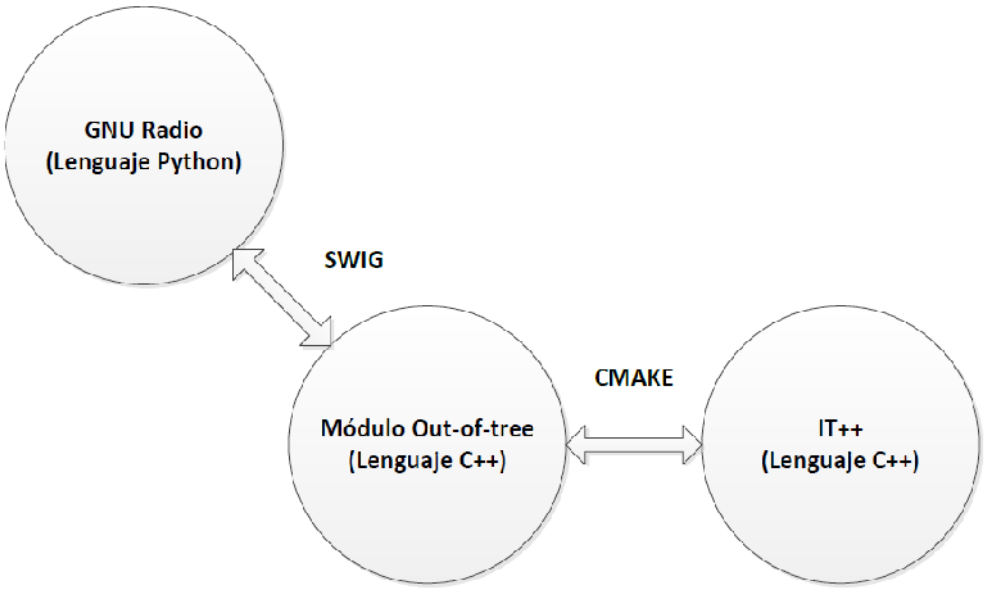

Figura 3.9. Integración de GNU Radio con bibliotecas externas.
Fuente: Tomado de [16].

Una forma de desarrollar sistemas de radio definido mediante *software* en GNU Radio es por medio de la interfaz gráfica de usuario GNU Radio *Companion* (GRC), que permite utilizar de una forma visual las funciones y bibliotecas de GNU Radio [16].

Esta herramienta gráfica permite al usuario la implementación de sistemas de procesamiento de señales para comunicaciones sin requerir conocimientos previos en Python ya que interpreta los bloques y los convierte a código de lenguaje de programación Python. Esta interfaz no es un compilador, solo es un intérprete [16].

Con lo mencionado anteriormente se puede realizar una clasificación de los tipos de bloques que se usa en GNU Radio:

- **Fuentes:** como ficheros de texto, imágenes, fuentes de audio, *hardware* radio, micrófono, etc.
- **Bloques de procesado de señal:** como filtros, amplificadores, operadores lógicos, operadores matemáticos, moduladores, demoduladores.

- **Sumideros:** visualizadores gráficos de la información tanto transmitida y receptada, como la FFT, espectro, constelación, etc.

Los bloques de GNU Radio se caracterizan por procesar los datos de manera continua desde su entrada hacia su salida. Idealmente, los bloques desempeñan únicamente una función para así hacer GNU Radio más flexible. Siendo caracterizados por su número de puertos de entrada y de salida así como del tipo de dato que manejan [16].

GNU Radio etiqueta los tipos de datos de acuerdo al tamaño de bits o precisión que requiera el sistema a desarrollar, siendo desde el más básico *bytes*, pasando por *shorts*, *ints*, *floats* hasta llegar a *complex*. Aunque para GNU Radio, un flujo de datos regular es sólo un vector cuya longitud es el número total de información procesada [15].

Las características de los tipos de datos son [15]:

- **Byte:** se conforma de un *byte* de datos, esto es, 8 bits por elemento.
- **Short:** consiste en un entero conformado por 2 *bytes*.
- **Int:** este tipo de datos se conforma por un entero de 4 *bytes*.
- **Float:** permite el punto flotante por medio de 4 *bytes*.
- **Complex:** es un arreglo de 8 *bytes*, que se compone de dos arreglos tipos *float*.

Los bloques de procesamiento de datos se conforman de cuatro tipos de archivos principales [15]:

- **Archivos xml:** en ellos se definen los parámetros del bloque como el tipo de dato a utilizar (*float, int, short, complex*), la biblioteca a la que pertenece, etc.
- **Archivos h:** son las bibliotecas de los bloques que se implementan.
- **Archivos cc:** donde se escribe el código de programación del proceso que realizará el bloque, el lenguaje de programación es C++.
- **Archivos i:** también conocidos como archivos *swig*, permiten a la herramienta SWIG obtener los parámetros para la comunicación entre los archivos escritos de los bloques de C++ y la interfaz de Python.

Ya que se ha realizado una síntesis del funcionamiento de GNU Radio y se conoce su arquitectura de *software*, lo que resta es presentar de qué manera se agrupan las librerías y archivos que están presentes dentro de la plataforma. Las mismas que son agrupadas en módulos dependiendo de la función que desempeñen, entre los principales tenemos los presentados en la Tabla 3.4.

Tabla 3.4 Módulos de GNU Radio.

GNU Radio Project	
gr	Módulo principal de GNU Radio, esta se necesitará prácticamente en todos los casos puesto que contiene bloques básicos como gran parte de las fuentes, gran parte de los sumideros así como funciones fundamentales como adición, sustracción, etc.
digital	Este módulo contiene las librerías y archivos que se encargan de llevar a cabo las modulaciones y demodulaciones digitales.
audio	Este módulo proporciona control sobre la tarjeta de sonido, permite enviar o recibir señales de audio a través de la tarjeta de sonido.
blocks	Este módulo contiene bloques de procesado comúnmente utilizados en los *flow graphs.*
blks2	Este módulo contiene bloques adicionales escritos en Python.
trellis	Este módulo provee los archivos necesarios para poder realizar codificaciones convolucionales.
analog	Es en este módulo donde se ubican los archivos relativos a las modulaciones analógicas.
wavelet	Este módulo proporciona bloques de procesado para las trasformadas *wavelet*.
fft	Este módulo proporciona bloques de procesado para la *Fast Fourier Transform* (FFT).
window	Este módulo contiene rutinas para el diseño de ventanas.
optfir	Este módulo contiene rutinas para el diseño óptimo de filtros de respuesta al impulso finita (FIR).
filter	Este módulo proporciona bloques de procesado para operaciones de filtrado.
qtgui	Módulo que proporciona sumideros gráficos basados en QT.
wxgui	Módulo que proporciona una GUI basada en Wx.
grc	Módulo necesario para poder utilizar la interfaz gráfica GNU Radio-*Companion*.
video-sdl	Este módulo proporciona control para permitir enviar o recibir señales de video.
vocoder	Este módulo incluye varios bloques de procesado los cuales implementan *vocoders.*
uhd	Este es el módulo que sirve de interfaz a la librería UHD para poder transmitir o recibir datos de los USRP.
Out of Tree	
osmosdr	Este módulo proporciona el soporte necesario para el uso del *hardware* OsmoSDR.
baz	Este módulo añade nuevas funcionalidades al proyecto GNU Radio, como por ejemplo soporte para el *hardware* RTL-2832.

Fuente: Tomada de [18].

Los módulos presentados en la Tabla 3.4 son gestionados a través de carpetas que se encargan de agrupar las librerías y archivos mencionados anteriormente. La estructura típica de un módulo de GNU Radio se presenta de la siguiente forma:

- **apps:** contiene las aplicaciones de prueba y ejemplos.
- **cmake:** contiene los diferentes archivos de configuración y no se modifican.
- **docs:** contiene los archivos de documentación del bloque que se generan de manera automática con Doxygen.
- **grc:** contiene los archivos (xml) de los bloques que se implementarán en GRC.
- **include:** contiene los archivos fuente de las librerías (h) de los bloques desarrollados.
- **lib:** contiene los archivos fuente (cc) de los bloques desarrollados.
- **Phyton:** contiene los *scripts* escritos en Python.
- **Swig:** contiene los archivos *swig* (i) con la configuración del intérprete de C++ y Phyton.

Cada carpeta, incluyendo la raíz, contiene archivos "CmakeLists.txt" con los datos de configuración que utiliza el programa CMAKE para compilar los bloques de procesamiento de señales [16].

Para facilitar el desarrollo de la estructura de módulos de proyectos propios existen archivos de ordenes (*script*) tales como gr-modtool.py, que es un archivo escrito en lenguaje de programación Python, que por medio de una interfaz sencilla permite al usuario final crear, modificar o eliminar módulos y bloques de los proyectos personalizados [16].

Para el correcto funcionamiento del *software* GNU Radio se procedió a realizar los pasos y comandos necesarios para su instalación en Ubuntu 13.04. Dicha información se encuentra en el anexo A.

3.2.1 GNU Radio *Companion.*

Como se explicó previamente, una aplicación de GNU Radio consiste en la interconexión de bloques (fuente - bloques de procesado - sumidero) [18]. GNU Radio *Companion* surge como alternativa a la programación directa en Python de la aplicación, se trata de una interfaz que permite el diseño de sistemas mediante programación visual. Esta herramienta es muy similar a Simulink de Matlab o Labview que son un entorno de programación gráfico.

Para realizar un esquemático en GNU Radio-Companion basta con arrastrar los bloques que se necesita hacia el entorno de programación e interconectarlos como se puede apreciar en la Figura 3.10, en el orden preciso y dependiendo de las características del sistema que vamos a implementar la cantidad de bloques variará.

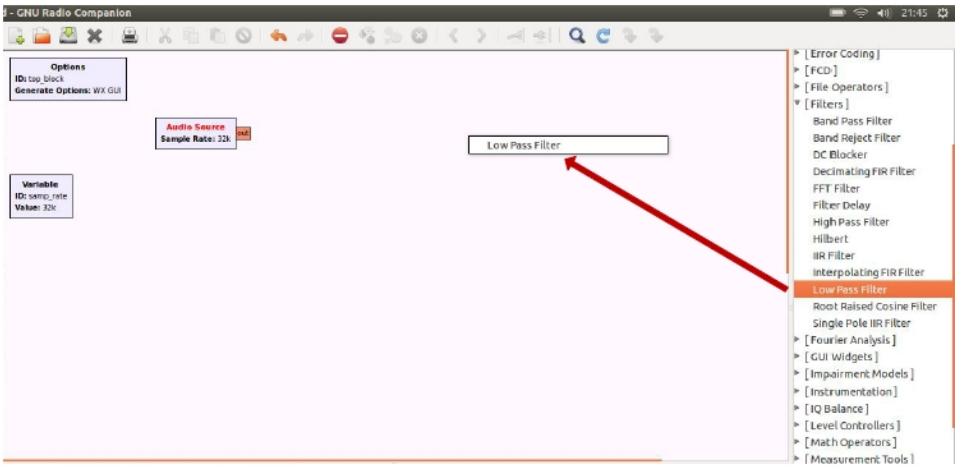

Figura 3.10. Entorno de programación de GNU Radio *Companion*.
Fuente: Imagen propia de los autores.

3.3 GStreamer

GStreamer es un *framework* para la creación de aplicaciones de sistemas multimedia. El diseño fundamental proviene de *video pipeline* en la Universidad de Oregon, así como algunas ideas de DirectShow [35].

La estructura de desarrollo de Gstreamer hace posible la escritura de cualquier tipo de aplicaciones de transmisión multimedia haciendo viable su implementación de manera sencilla, maneja audio, video o ambos a la vez como se puede observar en la Figura 3.11. El diseño de *pipeline* está hecho para tener una pequeña sobrecarga por encima de lo que inducen los filtros aplicados [35]. Esto ayudará al programador a la reproducción de audio, *streaming*, edición y grabación de audio o video capturado por cualquier cámara digital. El concepto de *pipeline* es la parte central para la edición del video, reproductores multimedia, etc., GStreamer es una herramienta multiplataforma que puede trabajar en: Linux/Unix, OpenMAX-IL (via gst-omx), Windows: DirectShow, Mac OS X: QuickTime [36].

En la Figura 3.11 se puede apreciar una captura de video con GStreamer desde el computador.

GStreamer proporciona varias herramientas, entre las que tenemos las siguientes [35]:

- Un API (*Application Programming Interface* - Interfaz de programación de aplicaciones) para aplicaciones multimedia.
- Arquitectura *plugin*.
- Arquitectura *pipeline*.
- Un mecanismo para el manejo de la comunicación.
- Un mecanismo para la sincronización.

- Más de 250 *plugins* proporcionando más de 1000 elementos.
- Un conjunto de herramientas.

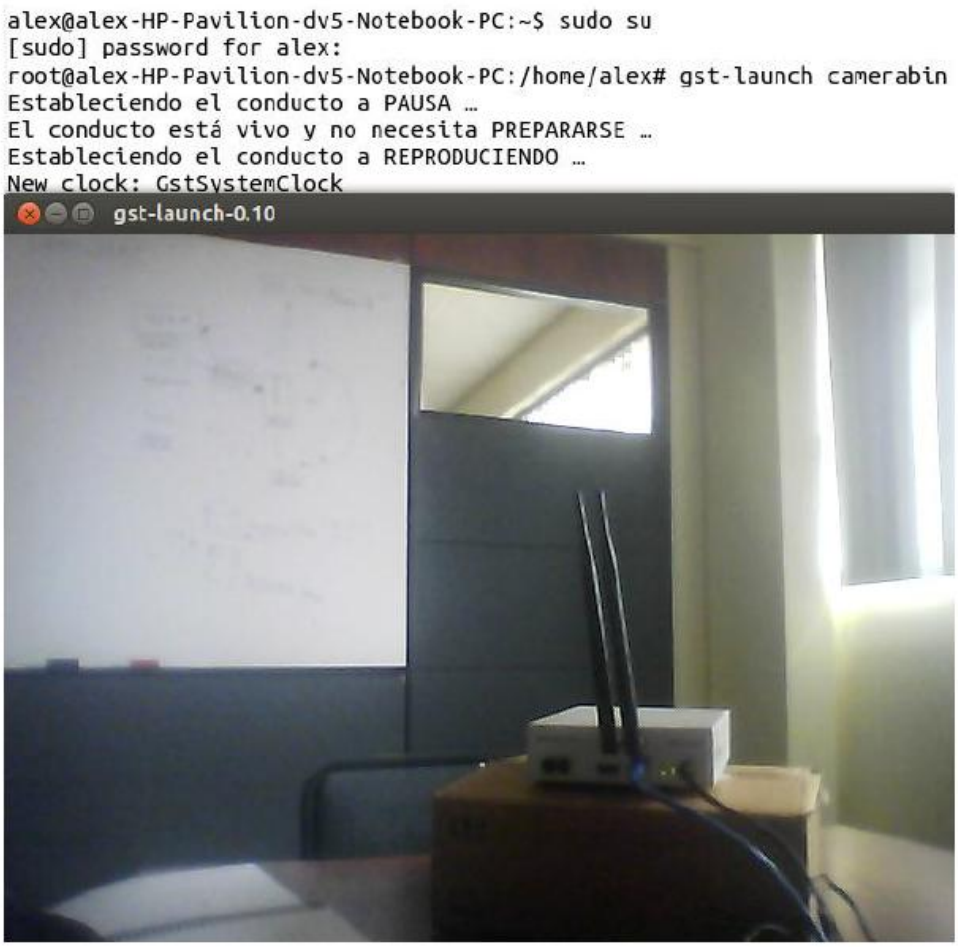

Figura 3.11. Captura de video con GStreamer desde la *webcam* del computador.
Fuente: Imagen propia de los autores.

La instalación de GStreamer en Ubuntu 13.04 se la detalla en el anexo F.

3.4 Mplayer

MPlayer es un reproductor multimedia libre. Reproduce la mayoría de los archivos MPEG, VOB, AVI, OGG/OGM, VIVO, ASF/WMA/WMV, QT/MOV/MP4, FLI, RM, NuppelVideo, YUV4MPEG, FILM, RoQ, PVA, soportados por algunos códecs nativos XAnim y DLL Win32. Además puede reproducir VideoCD, SVCD, DVD, 3ivx y DivX 3/4/5. También trae la opción para subtítulos, soportando 14 formatos diferentes (MicroDVD, SubRip, SubViewer, Sami, VPlayer, RT, SSA, AQTitle, JACOsub, VOBsub, CC, OGM, PJS y MPsub).

Junto al paquete de descarga de MPlayer, se puede encontrar la aplicación MEncoder, una herramienta esencial para el proceso de codificación de video o audio. Además trae por

defecto un GUI (Interfaz gráfica de usuario) hecho en GTK, gmplayer, aunque existen también algunos otros GUI's más potentes como por ejemplo KMPlayer, el cual está hecho en Qt [37].

El reproductor puede correr en la mayoría de las plataformas, incluyendo Linux como se aprecia en la Figura 3.12, derivados de Unix, Mac OS X y también Windows.

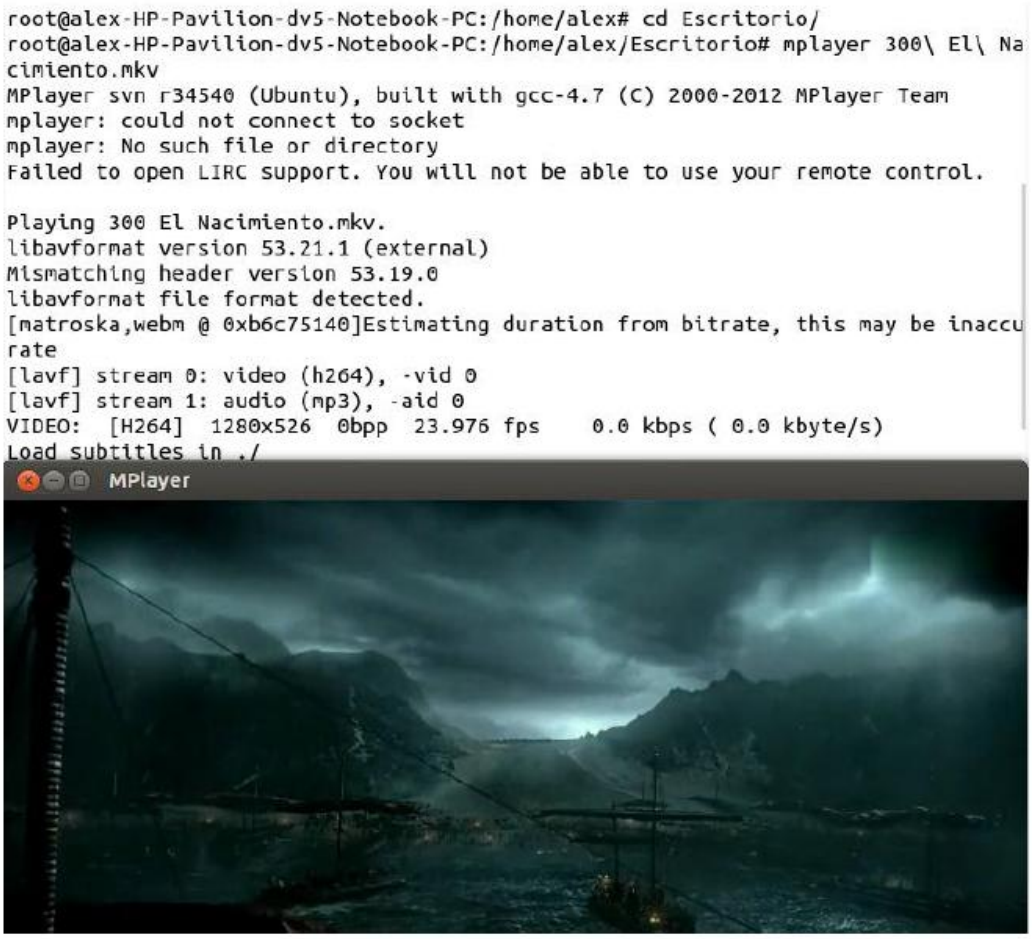

Figura 3.12. Reproductor Mplayer.
Fuente: Imagen propia de los autores.

3.4.1 Lista de *codecs* más importantes.

- Video MPEG1 (VCD) y MPEG2 (SVCD/DVD/DVB).
- MPEG4, DivX, OpenDivX (DivX4), DivX 5.02, XviD, y otras variantes de MPEG4.
- Windows Media Video v7 (WMV1), v8 (WMV2) y v9 (WMV3) usado en archivos wmv.
- RealVideo 1.0, 2.0 (G2), 3.0 (RP8), 4.0 (RP9).
- Sorenson v1/v3 (SVQ1/SVQ3), Cinepak, RPZA y otros *codecs* QuickTime comunes.
- Cinepak e Intel Indeo *codecs* (3.1, 3.2, 4.1, 5.0).
- VIVO 1.0, 2.0, I263 y otras variantes h263(+).
- FLI/FLC.
- Decodificador nativo para HuffYUV.

- Varios formatos simples y antiguos del tipo RLE-like.
- MPEG layer 1, 2, y 3 (MP3) audio.
- AC3/A52 (Dolby Digital) audio (*software* o S/PDIF).
- WMA (DivX Audio) v1, v2 (*codec* nativo).
- WMA 9 (WMAv3), Voxware audio, ACELP.net etc (usando x86 DLLs).
- RealAudio: COOK, SIPRO, ATRAC3, DNET (usando RP's *plugins*).
- QuickTime: Qclp, Q-Design QDMC/QDM2, MACE 3/6 (usando QT's DLLs).
- Ogg Vorbis audio *codec* (lib nativa).
- VIVO audio (g723, Vivo Siren).
- Alaw/ulaw, (ms)gsm, pcm, *adpcm y otros formatos de audio simples y antiguos.
- Flash Video (FLV).

La instalación de Mplayer en Ubuntu 13.04 se la detalla en el anexo E.

CAPÍTULO IV

FUNCIONAMIENTO Y PRUEBAS

En este capítulo se mostrarán los módulos desarrollados en el presente trabajo de tesis empleando la herramienta de *software* GNU Radio *Companion*, que funcionando en conjunto con los dispositivos SDR (USRP N210) permitió cumplir con los objetivos planteados.

Para la implementación de los sistemas de comunicaciones se ha empleado un esquema transmisor-receptor; utilizando los siguientes equipos:

- Dos computadores portátiles con sistema operativo Ubuntu 13.04. Una de ellas consta con un procesador Intel Core I5 y 6 GB de RAM; y la otra con un procesador AMD Tourion II Dual Core de 4 GB de RAM. Cualquiera de ellas puede funcionar como transmisor y como receptor.
- Dos equipos SDR USRP N210 cuyas características se detallan en el capítulo 3.
- Para la transmisión de video se empleará la *webcam* propia del computador.

4.1 Conexión del equipo USRP N210

Como se indica en la Figura 4.1, para la conexión física entre el equipo y el computador es necesario un cable Ethernet categoría 5e. Es importante mencionar que el computador deberá tener una interfaz Gigabit Ethernet para ser compatible y permitir la comunicación con el dispositivo USRP N210.

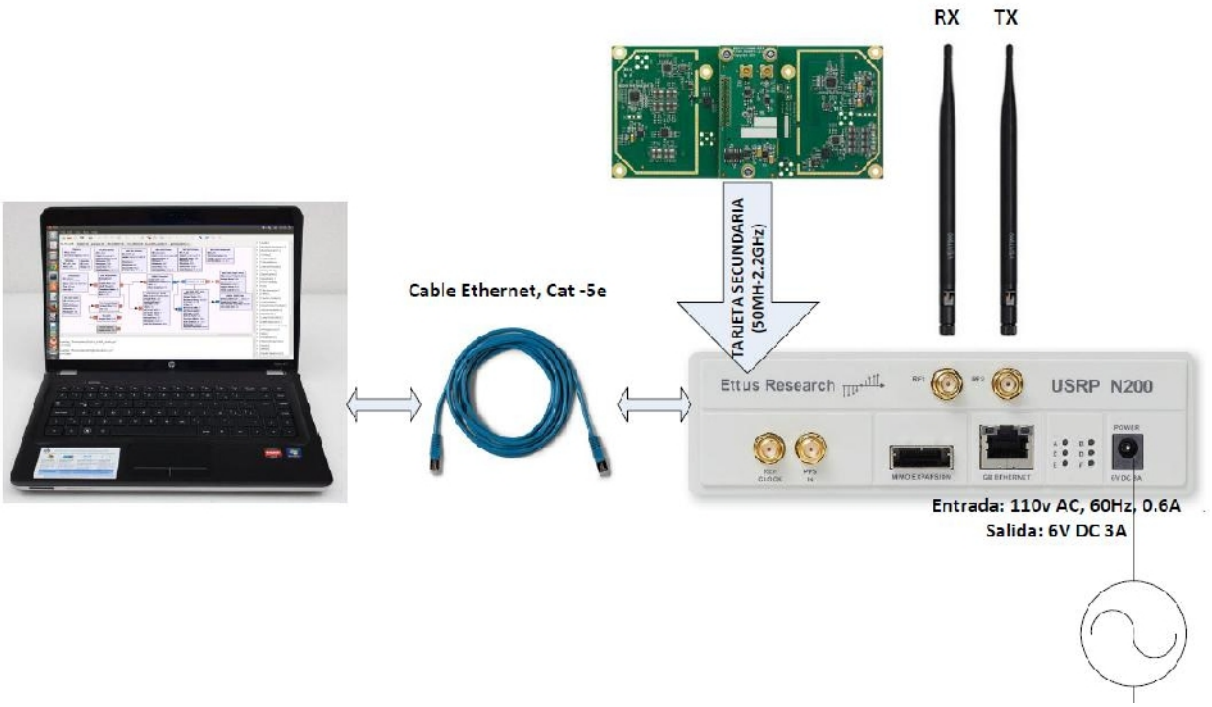

Figura 4.1. Diagrama de conexión del equipo con el computador.
Fuente: Imagen propia de los autores.

Ahora es necesario asegurarse que el *firmware* y la imagen de la FPGA estén cargados correctamente en el dispositivo, para ello deberá estar encendido el led (diodo emisor de luz) indicador correspondiente a la letra “D” en el panel frontal, caso contrario se procederá a cargar manualmente el *firmware* y la imagen del dispositivo, como se explica en el anexo C.

En la Figura 4.2 se muestra el esquema de trabajo transmisor-receptor de los equipos SDR correctamente conectados.

Figura 4.2. Esquema de trabajo emisor-receptor
Fuente: Imagen propia de los autores.

Seguidamente se procede a hacer una prueba de comunicación entre el dispositivo y el computador mediante el comando PING (*packet internet groper*). Por defecto, el USRP N210 tiene asignada la dirección IP: **192.168.10.2** y para diferenciar el transmisor del receptor se ha colocado también la dirección IP: **192.168.10.3**. Si el PING es exitoso como se observa en la Figura 4.3, podemos comenzar con la programación de los sistemas de comunicaciones.

```
max@Max-Peralta:~$ sudo su
[sudo] password for max:
root@Max-Peralta:/home/max# ifconfig eth0 192.168.10.1
root@Max-Peralta:/home/max# ping 192.168.10.3
PING 192.168.10.3 (192.168.10.3) 56(84) bytes of data.
64 bytes from 192.168.10.3: icmp_req=1 ttl=32 time=1.12 ms
64 bytes from 192.168.10.3: icmp_req=2 ttl=32 time=1.07 ms
64 bytes from 192.168.10.3: icmp_req=3 ttl=32 time=1.06 ms
64 bytes from 192.168.10.3: icmp_req=4 ttl=32 time=1.04 ms
```

Figura 4.3. Respuesta del comando PING.
Fuente: Imagen propia de los autores.

A continuación se explicará el diseño de cada uno de los sistemas de comunicaciones propuestos: comenzando con un analizador de espectros, un transmisor y receptor FM, un transmisor FM con múltiples portadoras y múltiples fuentes de audio; también se implementará un sistema de comunicaciones digital con diferentes modulaciones y finalmente un sistema de comunicación experimental para transmisión digital de video.

4.1.1 Caracterización de la conexión con el computador.

Antes de comenzar con la programación de los sistemas de comunicaciones para su implementación en conjunto con los equipos SDR, es necesario establecer límites de diseño en la programación de los mismos. Es decir, valores como: las amplitudes de las señales que se envían hacia el DAC que tiene un rango de -1V a +1V [18], valores que estén fuera de este rango saturarán el DAC. La ganancia de TX o RX, dependen de las tarjetas de RF utilizadas que serán de 0dB a 25dB en el transmisor y de 0dB a 31.5dB en el receptor.

También es de suma importancia considerar la tasa de muestreo (*sample rate)*. En el caso del USRP N210 la máxima tasa soportada es de 50 millones de muestras por segundo (MSps) o de 25MSps según el número de bits utilizados para la representación de cada muestra (16 bits o 32 bits) respectivamente.

En base a estos factores se calcula la velocidad de transmisión que deberá ser soportada por la conexión GbE. Para ello se emplea la fórmula:

$$V(bps) = N_{bits}\left(\frac{bits}{Sample}\right) * sample\ rate\ \left(\frac{samples}{second}\right) \leq 1 \cdot 10^9\,. \qquad (4.1)$$

La máxima tasa de muestreo estará fijado a 25MSps, que generará una velocidad de transmisión (caso de 32 bits por muestra) de:

$$V(bps) = \left[16\left(\frac{bits}{I\ Sample}\right) + 16\left(\frac{bits}{Q\ Sample}\right)\right] * 25 \cdot 10^6\left(\frac{samples}{second}\right) = 800 \cdot 10^6 \leq 1 \cdot 10^9. \qquad (4.2)$$

Esta velocidad es soportada por el estándar GbE al ser inferior al Gbps. Este cálculo se ha realizado para el caso de un enlace en modo de operación *half-duplex.*

De acuerdo con la tasa de Nyquist, la frecuencia de muestreo *fs* debe ser el doble del ancho de banda de la señal a muestrear (parte positiva del espectro) para evitar el *alias*. Esto significa que la señal modulada (banda base) como máximo tendrá un ancho de banda *fs*/2 (parte positiva del espectro). Por otra parte, a la hora de calcular el MBW (*modulation bandwith*) de la señal, hay que tener en cuenta el número de muestras utilizadas por símbolo [18] de la siguiente manera:

$$f_{bb(Hz)=} \frac{f_s}{2 * N\left(\frac{samples}{symbol}\right)}. \quad (4.3)$$

Por lo tanto, el máximo ancho de banda de la señal (el mínimo número de muestras por símbolo es 2) será *fs*/4. Esto se traduce en un máximo ancho de banda de 6.25MHz para el caso de 32 bits y 12.5MHz para el caso de 16 bits (el ancho de banda mencionado describe el espectro unilateral).

El ancho de banda de la señal está íntimamente ligado a la velocidad de transmisión:

$$R_b(bps) = 2f_{bb} * log_2 M. \quad (4.4)$$

Donde ***Rb*** es la tasa de bits y ***M*** el tamaño del alfabeto. Sustituyendo el parámetro f_{bb} por el obtenido en la ecuación (4.3) se obtiene:

$$R_b(bps) = \frac{f_s}{N\left(\frac{samples}{symbol}\right)} * log_2 M. \quad (4.5)$$

Por último, si se quiere obtener la velocidad de símbolo, la ecuación (4.5) quedará como:

$$R_s(baudios) = \frac{f_s}{N\left(\frac{samples}{symbol}\right)}. \quad (4.6)$$

4.2 Sistemas de comunicaciones analógicos

4.2.1 Analizador de espectros.

En este apartado se procederá al desarrollo de un analizador de espectros que funcionará en una banda de frecuencias desde 50MHz a 2.2GHz debido a las limitaciones de la tarjeta secundaria WBX.

En la Figura 4.4 se muestra el esquema de bloques del analizador de espectros.

Como podemos observar su configuración es sencilla, solo es necesario añadir dos bloques principales que son: **UHD: USRP Source** que es el encargado de recibir las señales desde el USRP a través de la interfaz Gigabit Ethernet para su procesamiento en el computador y el bloque **FFT Sink** que nos permite visualizar el espectro de las señales al aplicar la FFT a las mismas. Los demás bloques que se observan en la figura sirven para definir valores necesarios para el procesamiento de las señales como: frecuencia de muestreo, la ganancia del receptor y el rango de frecuencias que serán soportadas por el equipo.

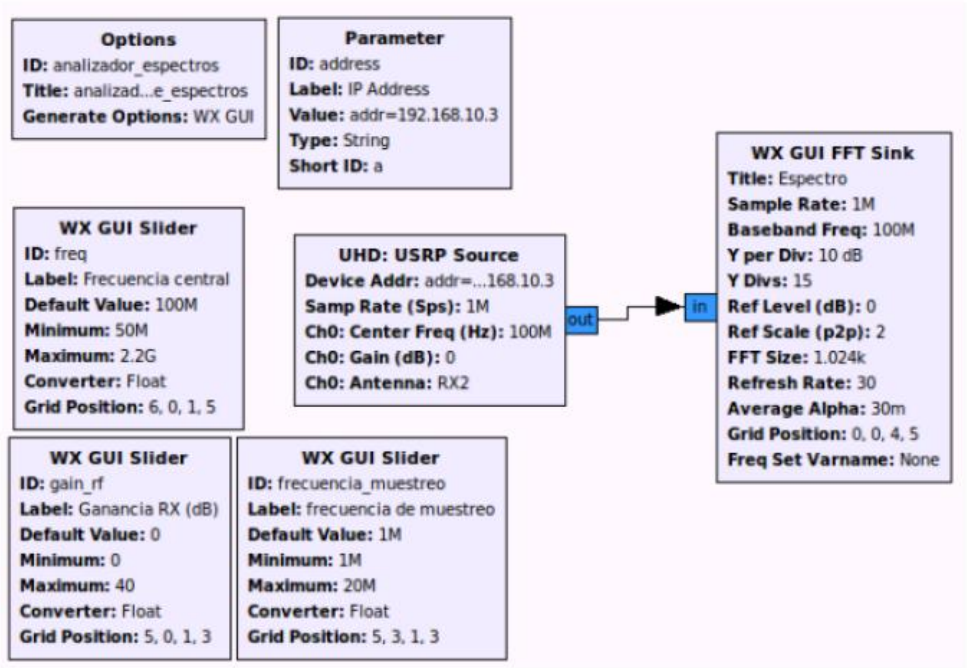

Figura 4.4. Esquema del analizador de espectros.
Fuente: Imagen propia de los autores.

La configuración del bloque **UHD: USRP Source** se muestra en la Figura 4.5.

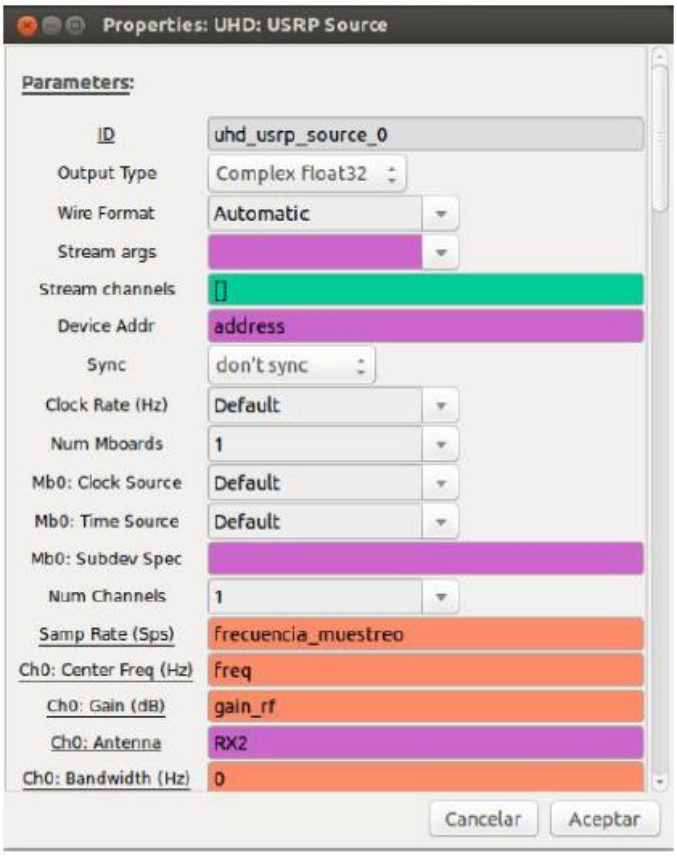

Figura 4.5. Configuración del bloque **UHD: USRP Source**.
Fuente: Imagen propia de los autores.

En este bloque se configuran parámetros como: la dirección IP del USRP, la tasa de muestreo, la frecuencia central, la ganancia del dispositivo, el tipo de antena empleada, entre otros.

En la Figura 4.6 se presenta el resultado obtenido, pudiéndose observar el espectro de varias emisoras del dial FM.

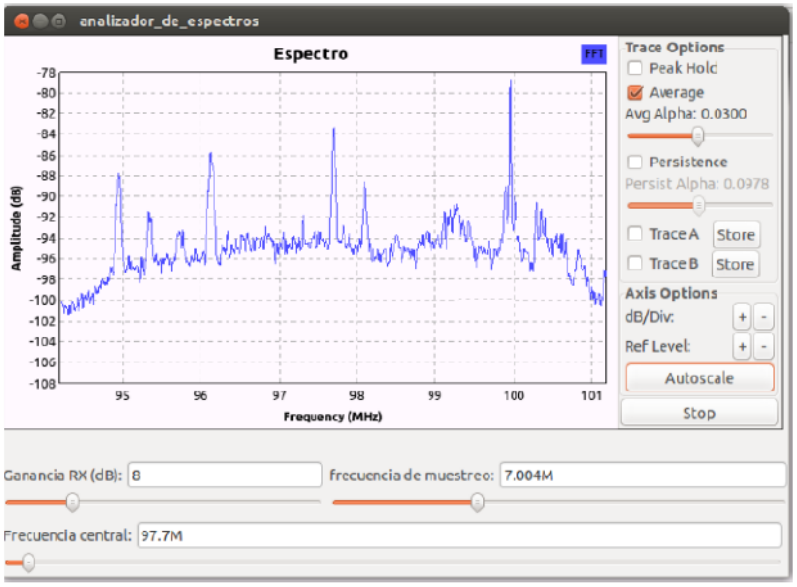

Figura 4.6. Espectro de señales FM.
Fuente: Imagen propia de los autores.

4.2.2 Sistema de transmisión FM.

Para la implementación de un transmisor FM se han empleado los bloques y configuraciones mostradas en la figura 4.7.

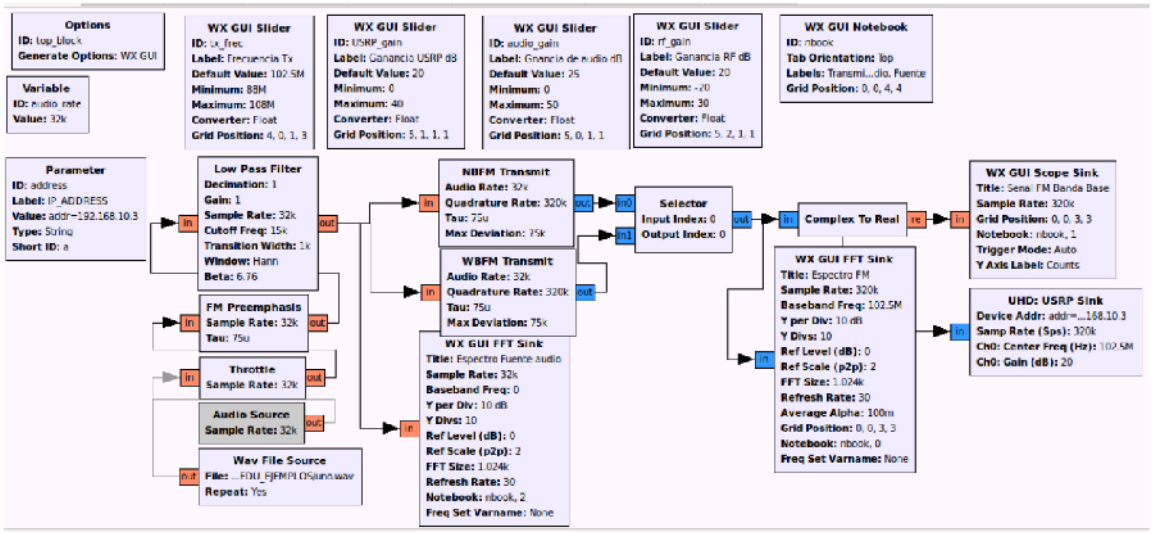

Figura 4.7. Esquema del transmisor FM.
Fuente: Imagen propia de los autores.

La fuente de audio es el bloque **Wav File Source**, mediante el cual se carga un archivo de audio en formato .wav. Es importante mencionar, que también se pueden emplear otras fuentes para generar la señal moduladora de este sistema, por ejemplo generadores de señales y el micrófono del computador o uno externo.

Esta señal moduladora pasa por una etapa de filtrado, la cual se detalla en el anexo D. La configuración de una transmisión en banda angosta o en banda ancha se realiza mediante los bloques: **NBFM** (*Narrow Band FM Transmit)* en caso de banda angosta y el bloque **WBFM** (*Wide Band FM Transmit*) en caso de banda ancha. En este punto se tiene una señal modulada en FM y el traslado a una frecuencia adecuada para la transmisión se realiza en el bloque **UHD: USRP Sink** en el cual se establece la frecuencia de la portadora y se especifica la tasa de muestras que recibe este sumidero desde el bloque modulador que será diez veces la tasa de audio de la fuente que se va a transmitir.

La configuración del bloque **UHD: USRP Sink** se muestra en la Figura 4.8.

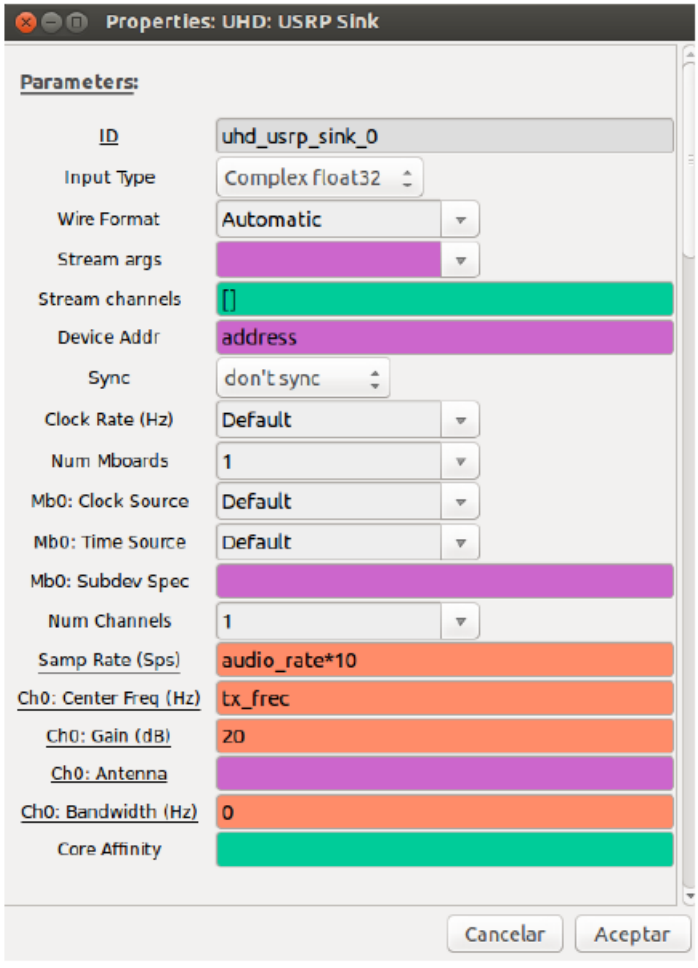

Figura 4.8. Configuración del bloque **UHD: USRP Sink**.
Fuente: Imagen propia de los autores.

En la Figura 4.9 se muestra el espectro de la señal FM transmitida.

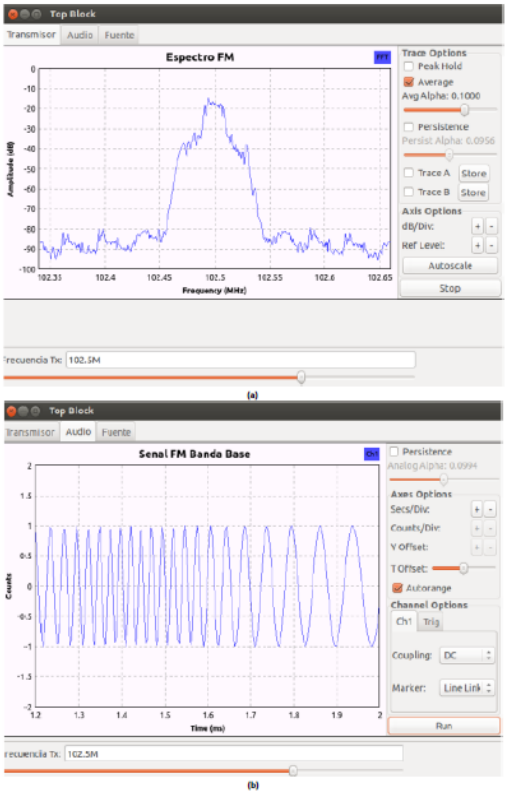

Figura 4.9. Espectro FM transmitido (a) dominio frecuencial (b) dominio temporal.
Fuente: Imagen propia de los autores.

4.2.3 Sistema de recepción FM.

Para la implementación de un receptor FM se han empleado los bloques y configuraciones mostradas en la Figura 4.10.

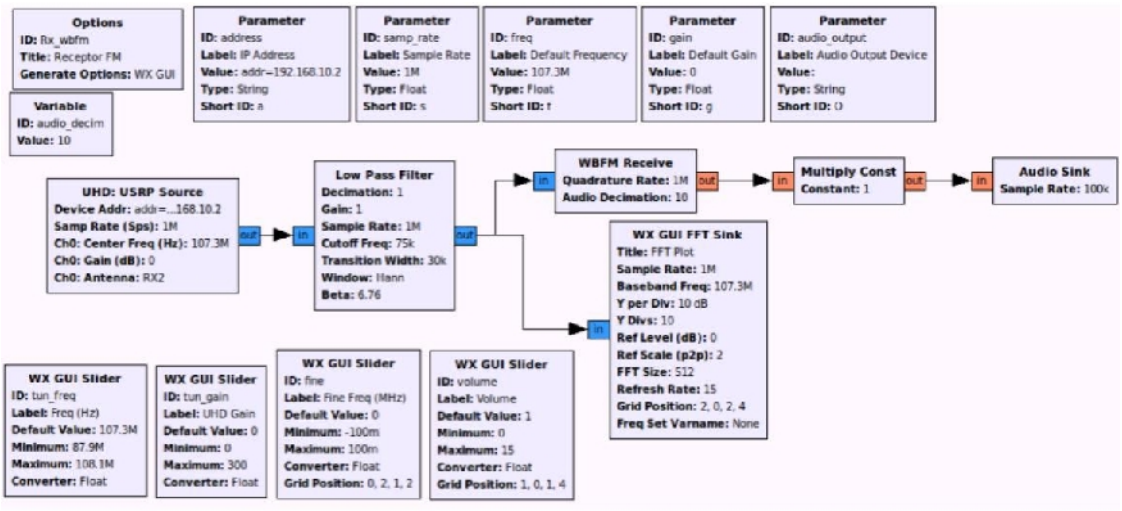

Figura 4.10. Esquema del receptor FM.
Fuente: Imagen propia de los autores.

Para este receptor las señales son recibidas desde del USRP a través de la interfaz Gigabit Ethernet por medio del bloque **UHD: USRP Source**. A continuación estas señales pasan por un proceso de filtrado para su posterior demodulación llevada a cabo por el bloque **WBFM Receive**; finalmente la señal FM demodulada pasa por el bloque **Multiply Const** que servirá para controlar el volumen de la señal que se reproducirá en los altavoces; función que cumple el bloque **Audio Sink**.

El resultado obtenido se muestra en la figura 4.11.

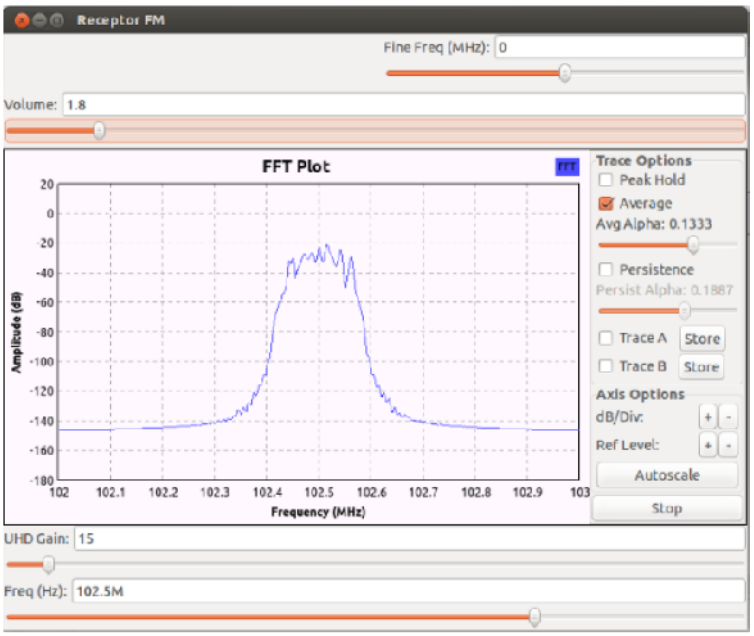

Figura 4.11. Espectro de la señal FM recibida en la frecuencia 102.5MHz.
Fuente: Imagen propia de los autores.

4.2.4 Sistema de transmisión FM con múltiples portadoras y múltiples fuentes de audio.

Para la implementación de este sistema se han empleado los bloques y configuraciones mostradas en la Figura 4.12.

En este caso existen tres fuentes de audio del tipo .wav y se mantiene el esquema del transmisor FM normal para la portadora central. Luego, para adicionar las nuevas portadoras se emplean los bloques **Signal Source** mediante los cuales se modularán las otras dos fuentes de audio; para ello basta con multiplicar estas portadoras por las señales previamente moduladas.

La configuración de los bloques **Signal Source** se muestra en la Figura 4.13.

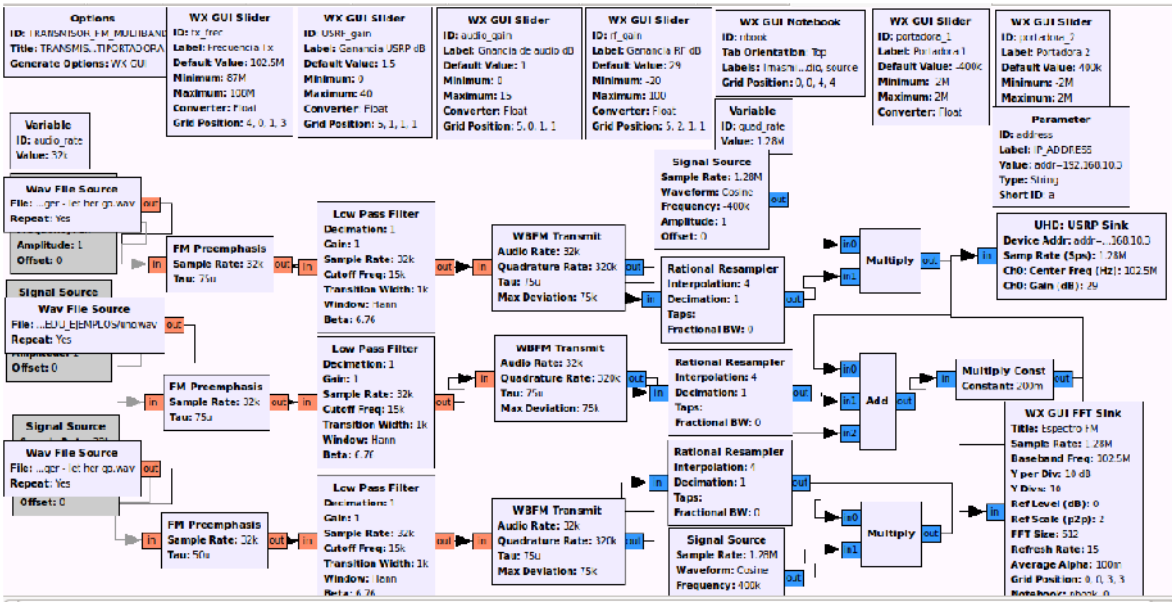

Figura 4.12. Esquema transmisor FM múltiples portadoras y múltiples fuentes de audio.
Fuente: Imagen propia de los autores.

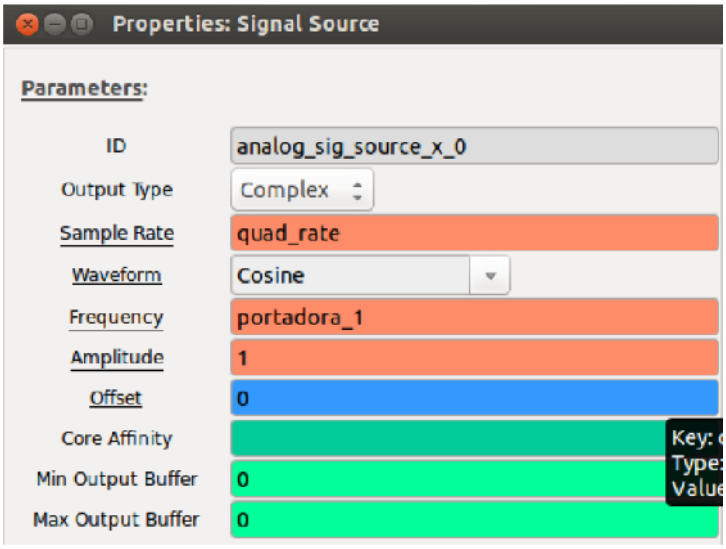

Figura 4.13. Configuración del **Signal Source**.
Fuente: Imagen propia de los autores.

Mediante un sumador se integran las tres señales moduladas y finalmente se procede al traslado de las mismas a las frecuencias adecuadas para su transmisión mediante el bloque **UHD: USRP Sink**.

En la Figura 4.14 se muestra el resultado obtenido donde se está transmitiendo las tres señales de audio en portadoras diferentes separas 400KHz de la frecuencia central que es 102.5MHz.

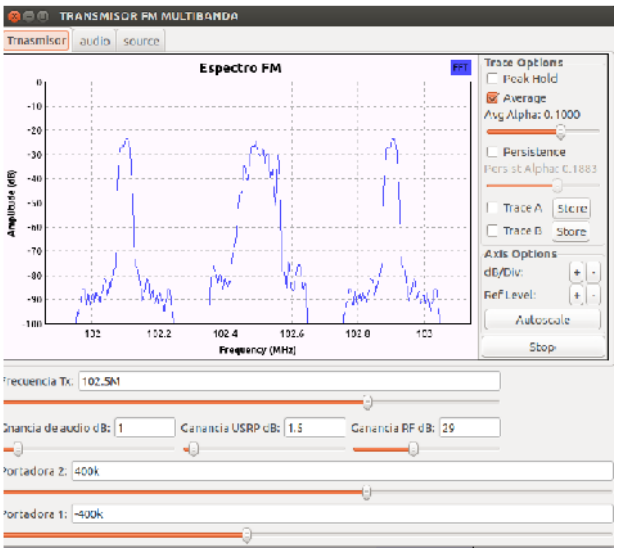

Figura 4.14. Espectro transmisor FM con múltiples portadoras y múltiples fuentes de audio.
Fuente: Imagen propia de los autores.

4.3 Sistemas de comunicaciones digitales

En esta sección se procederá al desarrollo de un sistema de comunicación digital; el mismo constará de un transmisor y un receptor. Para este sistema se podrán seleccionar tres tipos de modulaciones digitales: M-QAM, M-PSK, GMSK. Los detalles sobre estas modulaciones digitales se encuentran especificados en el capítulo 2.

4.3.1 Transmisor digital.

En la Figura 4.15 se muestra el diseño llevado a cabo para implementar el transmisor digital.

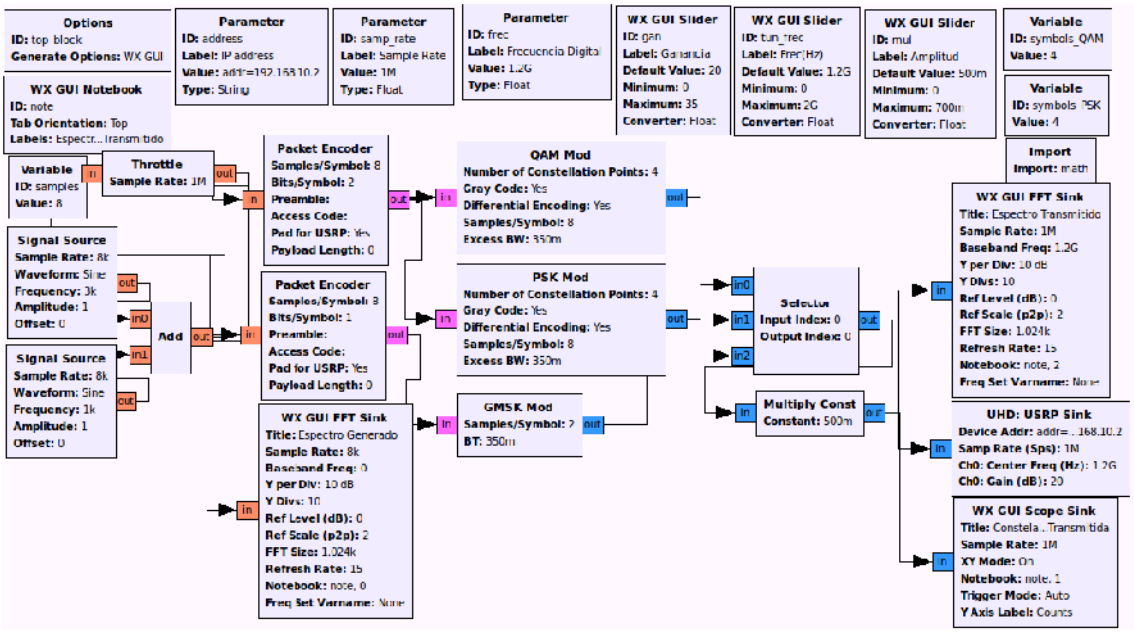

Figura 4.15. Esquemático del transmisor digital.
Fuente: Imagen propia de los autores.

Como se observa en la Figura 4.15 se utilizará como fuente de información dos tonos senoidales generados por el bloque **Signal Source**. Cabe mencionar que se puede emplear otros tipos de fuentes como: **Audio Source** que permite tomar el audio generado por el micrófono del computador o el bloque **Wav File Source** que permite cargar archivos de audio desde el computador en formato .wav.

La configuración del bloque **Signal Source** se muestra en la figura 4.16 donde se configura la frecuencia y tipo de señal que se va a transmitir. Para este sistema se empleó una suma de dos tonos (señales senoidales) a 1KHz y 3KHz.

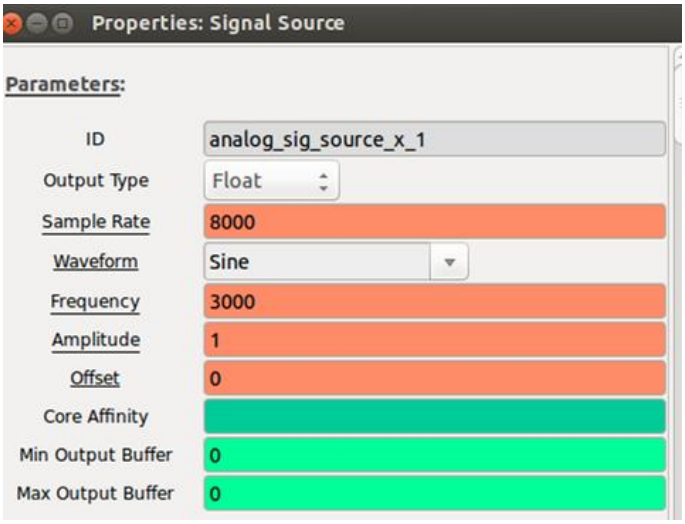

Figura 4.16. Configuración del bloque **Signal Source**.
Fuente: Imagen propia de los autores.

El bloque **Throttle** tiene la función de limitar el uso de la CPU; el bloque **Packet Encoder** tiene la función de transformar las muestras tipo *float* en tipo *byte* por medio de un empaquetado de datos. Luego se procede a modular estas señales, mediante el selector se escoge la modulación deseada y a través del bloque **Scope Sink** podemos observar el diagrama de constelación obtenido.

En la Figura 4.17 se muestran los parámetros de configuración del modulador QAM; en este caso, se implementará una modulación 4-QAM y se ha seleccionado el parámetro *Excess BW* que es coeficiente de *roll off* del filtro de coseno alzado con un valor de 0.35 con la finalidad de reducir la interferencia intersímbolo.

En la Figura 4.18 se muestran los parámetros de configuración del modulador PSK; en este caso, se implementará una modulación 4-PSK.

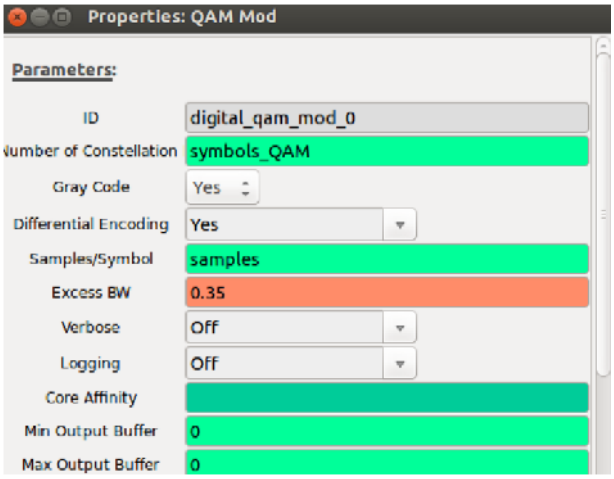

Figura 4.17. Configuración del bloque **QAM Mod**.
Fuente: Imagen propia de los autores.

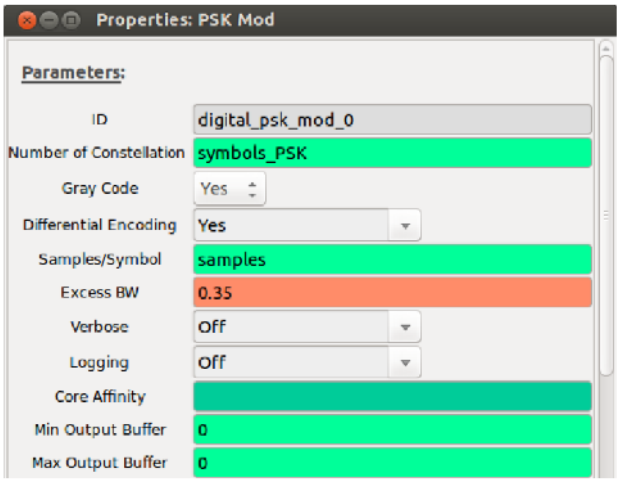

Figura 4.18. Configuración del bloque **PSK Mod**.
Fuente: Imagen propia de los autores.

En la Figura 4.19 se muestran los parámetros de configuración del modulador GMSK.

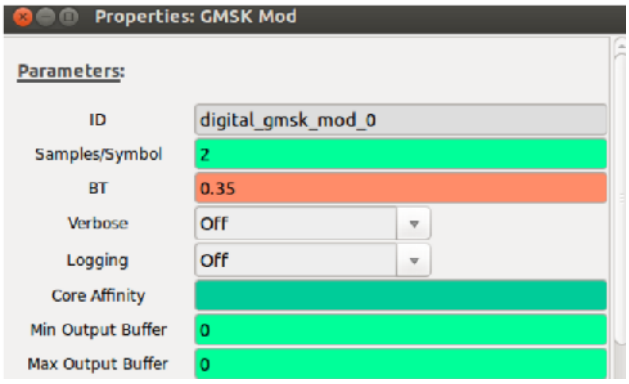

Figura 4.19. Configuración del bloque **GMSK Mod**.
Fuente: Imagen propia de los autores.

El bloque **Multiply Const** tiene la función de adecuar la señal a niveles soportados por la tarjeta RF y finalmente se procede al traslado en frecuencia mediante el bloque **USRP Sink**.

4.3.2 Receptor digital.

Una vez desarrollado el transmisor digital, en esta sección se desarrollará el proceso de recepción y demodulación de la señal que se emitió en el transmisor. El esquemático del receptor se muestra en la Figura 4.20.

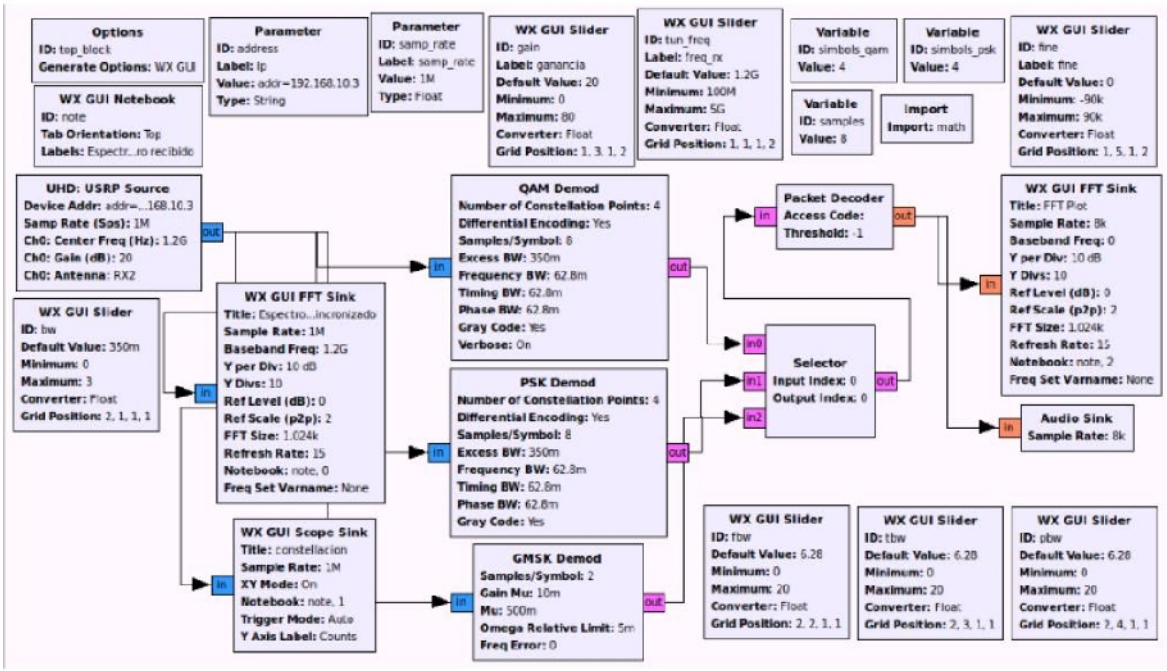

Figura 4.20. Esquemático del receptor digital.
Fuente: Imagen propia de los autores.

Mediante el bloque **UHD: USRP Source** se receptan las señales emitidas por el transmisor digital. Seguidamente se colocó el bloque **Throttle** que tiene la función de limitar el uso de la CPU; luego de esto se procede a demodular la señal escogiendo la misma a través del selector.

Los bloques demoduladores presentados a continuación mostrarán la configuración de los diferentes sistemas empleados para el receptor digital. En la Figura 4.21 se muestran los parámetros de configuración del demodulador 4-QAM.

En la Figura 4.22 se aprecia la configuración de bloque demodulador de PSK, con parámetros para 4-PSK.

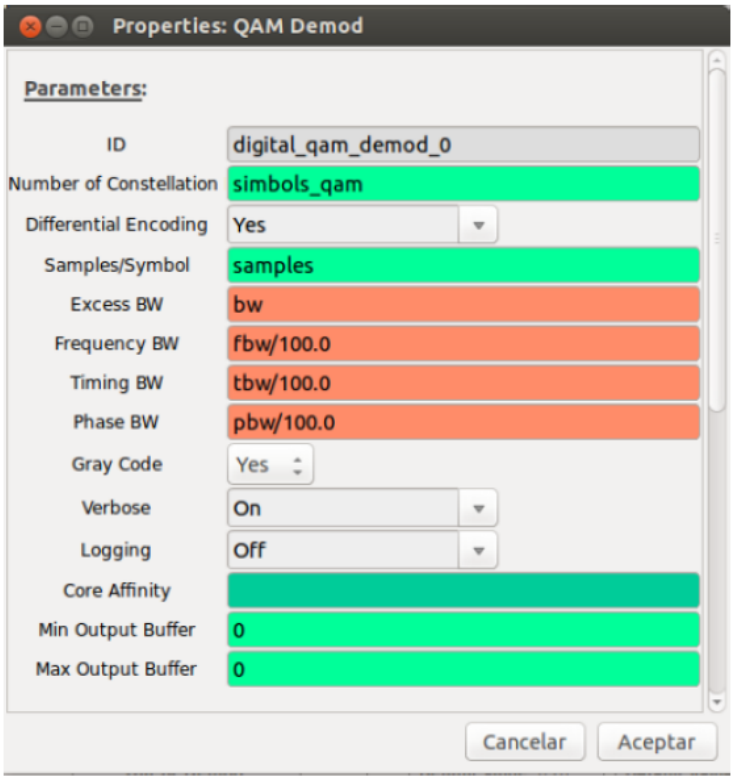

Figura 4.21. Configuración del bloque **QAM Demod**.
Fuente: Imagen propia de los autores.

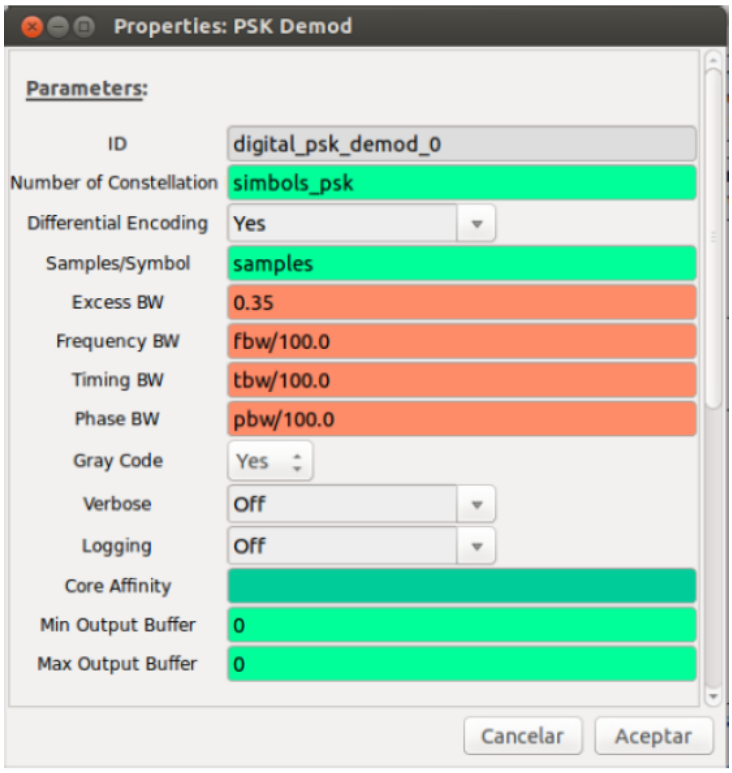

Figura 4.22. Configuración del bloque **PSK Demod**.
Fuente: Imagen propia de los autores.

En la Figura 4.23 se aprecia la configuración del bloque demodulador de GMSK.

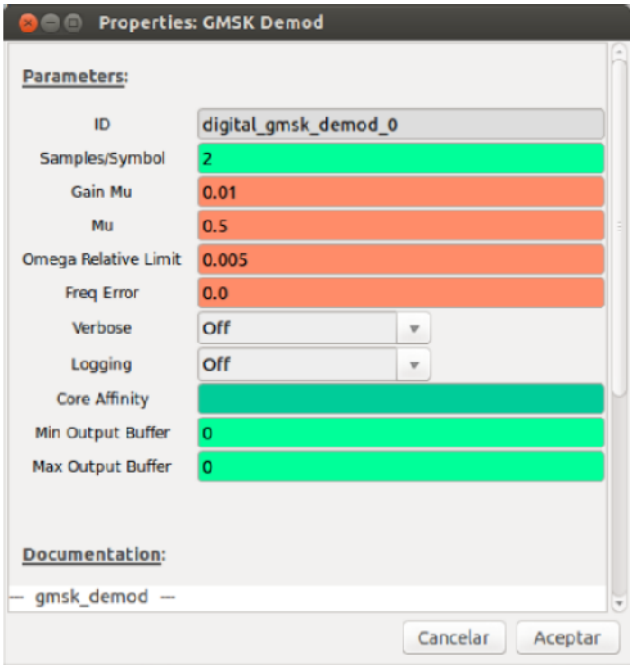

Figura 4.23. Configuración del bloque **GMSK Demod**.
Fuente: Imagen propia de los autores.

4.3.3 Resultados obtenidos.

En este sistema de transmisión y recepción digital se ha implementado 3 sumideros gráficos mediante los cuales se puede visualizar los espectros de los tonos generados, los diagramas de constelación y el espectro de transmisión y recepción.

4.3.3.1 Sistema 4-QAM.

En la Figura 4.24 se muestra el espectro generado por dos tonos de frecuencia a 1KHz y 3KHz respectivamente.

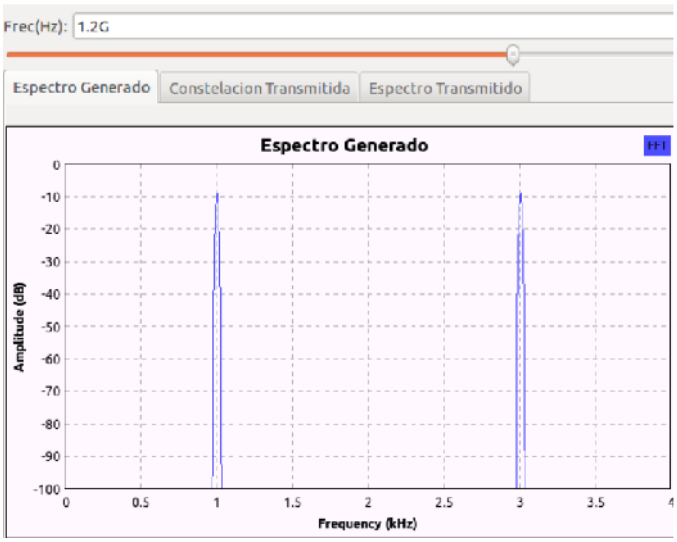

Figura 4.24. Espectro de tonos de frecuencia 1 y 3KHz.
Fuente: Imagen propia de los autores.

En la Figura 4.25 se observa el diagrama de constelación transmitido.

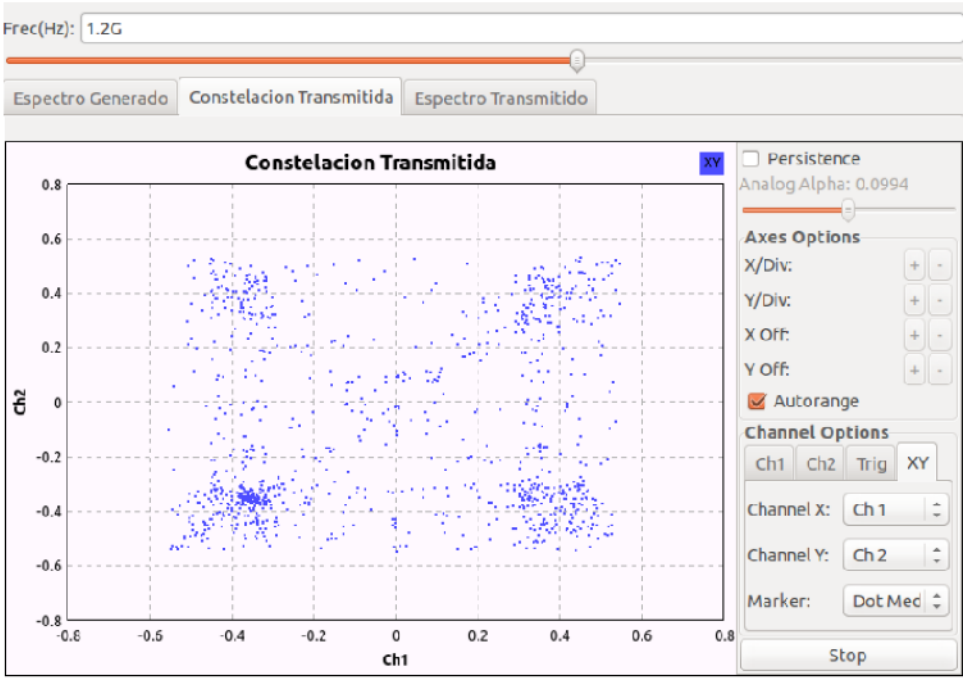

Figura 4.25. Diagrama de constelación transmitida para 4-QAM.
Fuente: Imagen propia de los autores.

En la Figura 4.26 se observa el espectro de la señal transmitida.

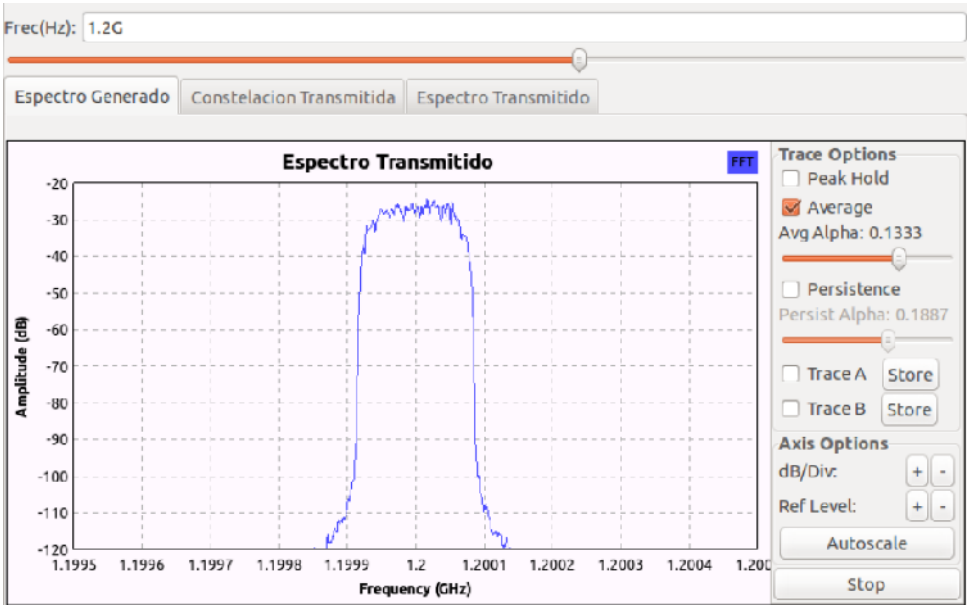

Figura 4.26. Espectro transmitido para 4-QAM en la frecuencia 1.2GHz.
Fuente: Imagen propia de los autores.

En la Figura 4.27 se observa el espectro de la señal recibida.

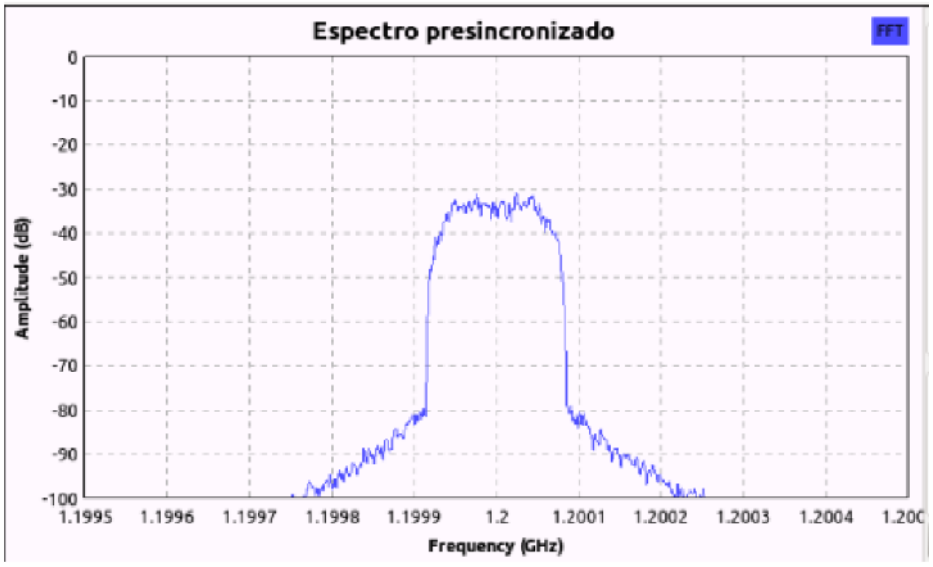

Figura 4.27. Espectro recibido de modulación 4-QAM en la frecuencia 1.2GHz.
Fuente: Imagen propia de los autores.

La constelación 4-QAM presentará los efectos descritos en el anexo B, como se puede observar en la Figura 4.28. Resumiendo estos efectos son: ruido que causa que los puntos de la constelación recibidos se dispersen en relación al diagrama de constelación ideal, atenuación que hace que los puntos de constelación se muevan hacia el centro, problemas con la sincronización de fase que causan que los diagramas de constelación giren sobre sí mismos y multitrayecto que afecta directamente a la amplitud y a la fase de la señal recibida, esto causa que la constelación presente un desplazamiento de fase.

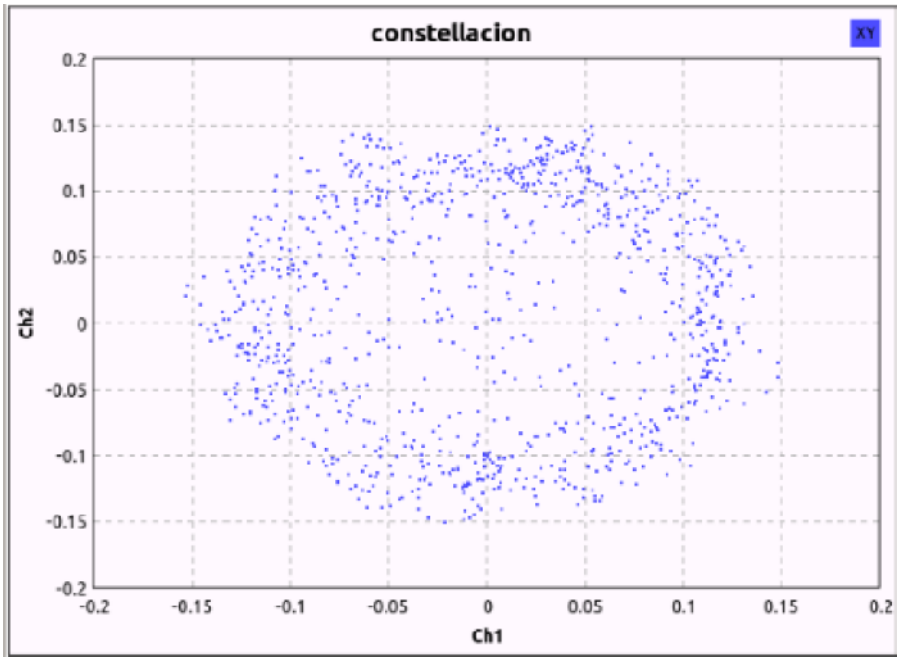

Figura 4.28. Constelación recibida de modulación 4-QAM en la frecuencia 1.2GHz.
Fuente: Imagen propia de los autores.

Finalmente, luego de realizar el proceso de compensación y demodulación se obtiene la imagen de la Figura 4.29.

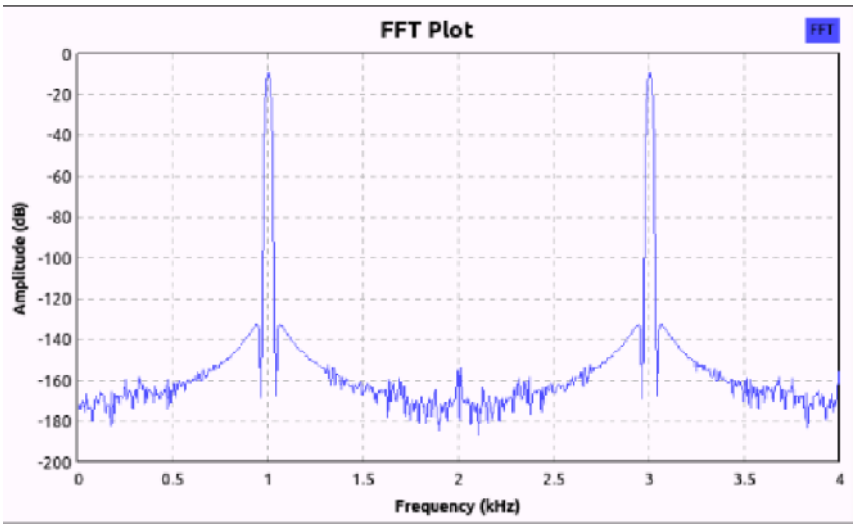

Figura 4.29. Espectro de los tonos recibidos y demodulados.
Fuente: Imagen propia de los autores.

Como podemos observar en la Figura 4.29 el proceso de recepción y demodulación ha sido realizado correctamente.

4.3.3.2 Sistema 4-PSK.

En la Figura 4.30 se muestra el espectro generado por dos tonos de frecuencia 1KHz y 3KHz.

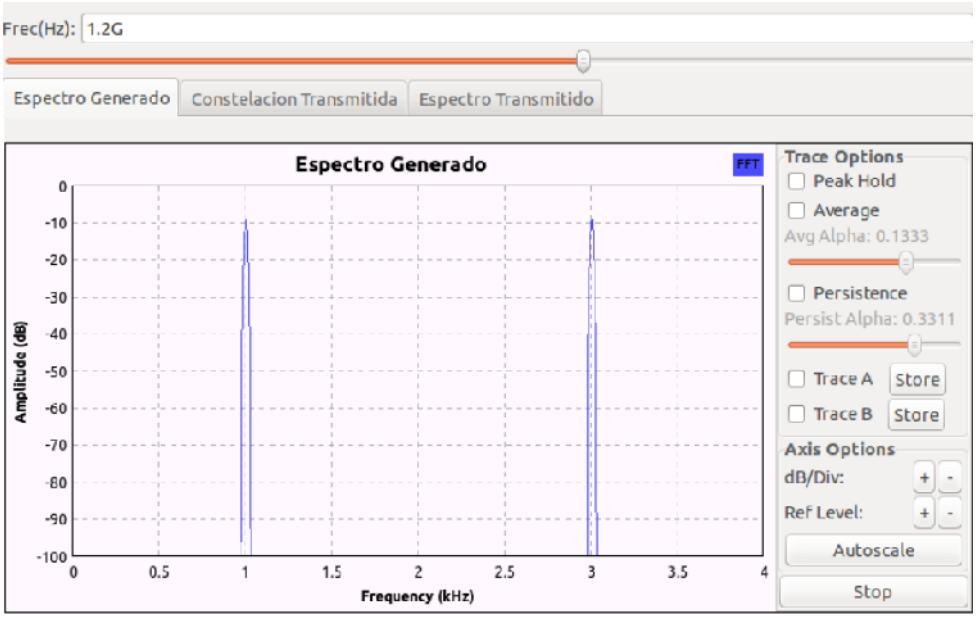

Figura 4.30. Espectro de tonos de frecuencia 1KHz y 3KHz.
Fuente: Imagen propia de los autores.

En la Figura 4.31 se observa el diagrama de constelación transmitido.

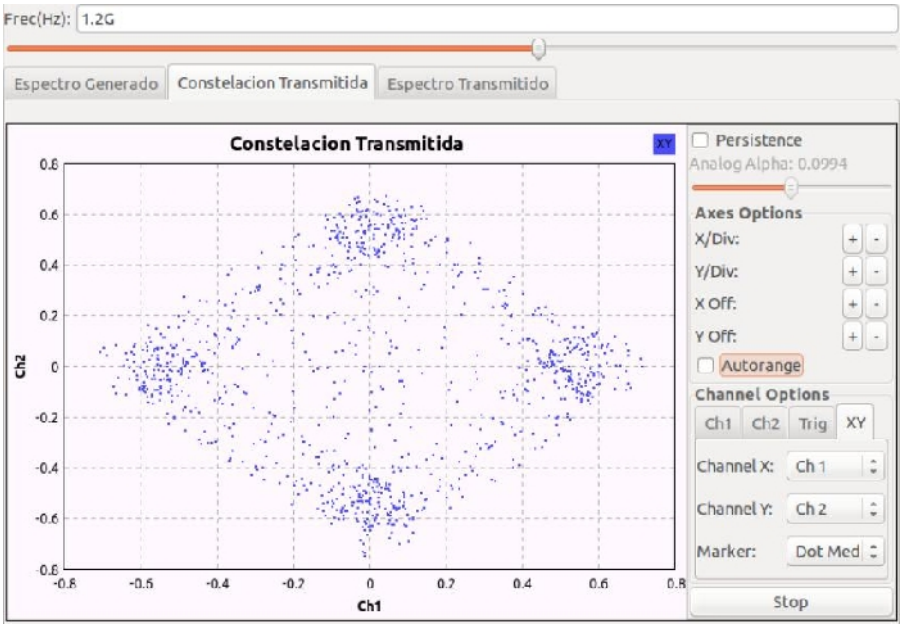

Figura 4.31. Diagrama de constelación transmitido para 4-PSK.
Fuente: Imagen propia de los autores.

En la Figura 4.32 se observa el espectro de la señal transmitida.

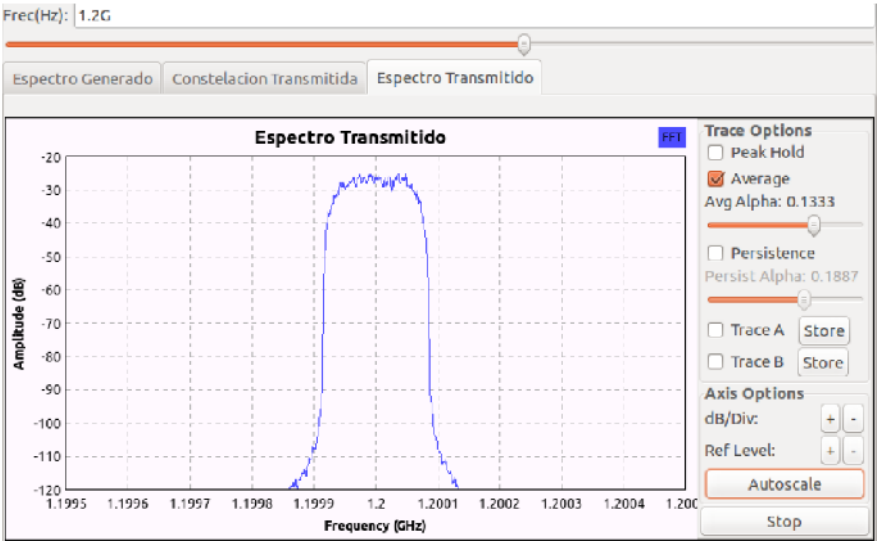

Figura 4.32. Espectro transmitido para 4-PSK en la frecuencia 1.2GHz.
Fuente: Imagen propia de los autores.

En la Figura 4.33 se observa el espectro de la señal recibida.

La constelación 4-PSK presentará los efectos descritos en el anexo B, como se puede observar en la figura 4.34.

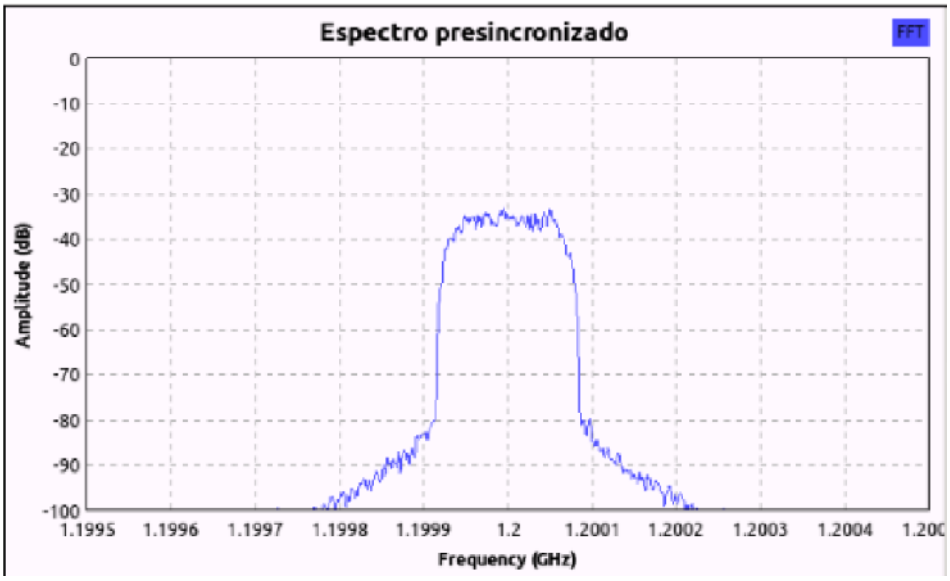

Figura 4.33. Espectro recibido de 4-PSK en la frecuencia 1.2GHz.
Fuente: Imagen propia de los autores.

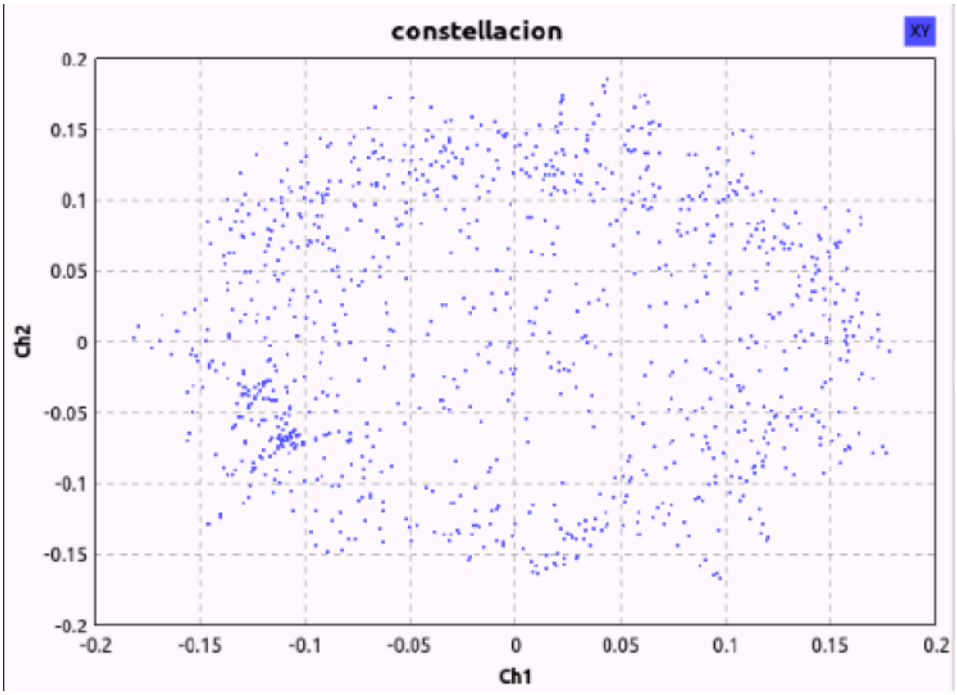

Figura 4.34. Constelación recibida de modulación 4-PSK en la frecuencia 1.2GHz.
Fuente: Imagen propia de los autores.

Finalmente, luego de realizar el proceso de compensación y demodulación se obtiene el espectro mostrado en la Figura 4.35.

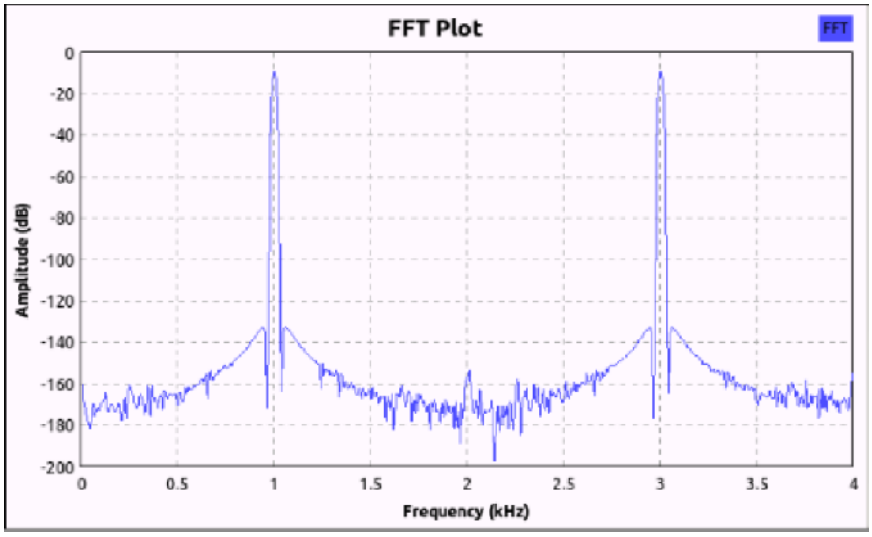

Figura 4.35. Espectro de audio recibido y demodulado.
Fuente: Imagen propia de los autores.

4.3.3.3 Sistema GMSK.

En la Figura 4.36 se muestra el espectro generado por dos tonos de frecuencia 1KHz y 3KHz.

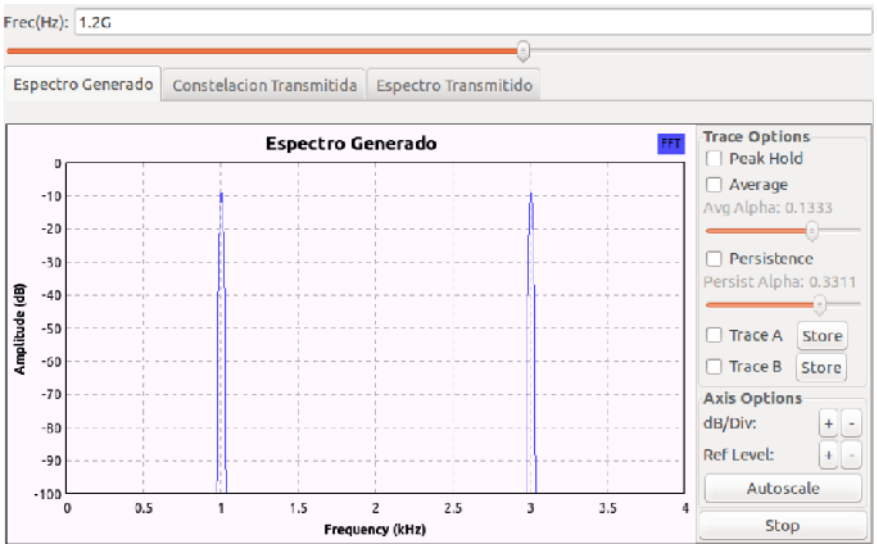

Figura 4.36. Espectro de tonos de frecuencia 1KHz y 3KHz.
Fuente: Imagen propia de los autores.

En la Figura 4.37 se observa el diagrama de constelación transmitido.

En la Figura 4.38 se observa el espectro de la señal transmitida.

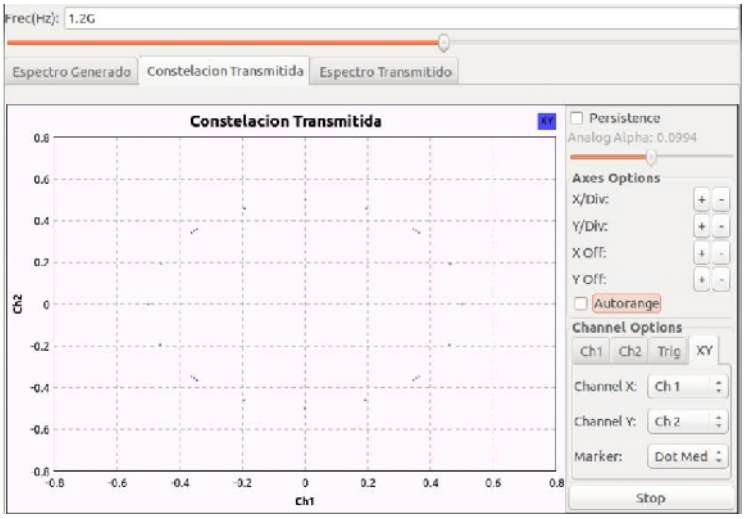

Figura 4.37. Diagrama de constelación transmitido para GMSK.
Fuente: Imagen propia de los autores.

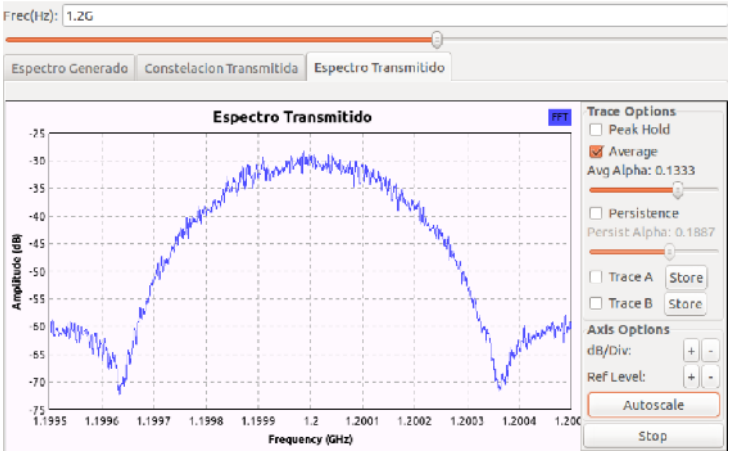

Figura 4.38. Espectro transmitido para GMSK en la frecuencia 1.2GHz.
Fuente: Imagen propia de los autores.

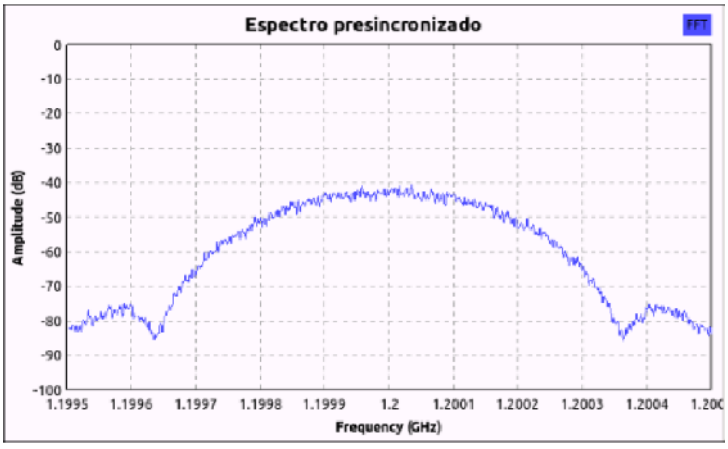

Figura 4.39. Espectro recibido de GMSK en la frecuencia 1.2GHz.
Fuente: Imagen propia de los autores.

La constelación GMSK recibida presentará los efectos descritos en el anexo B, como se puede observar en la figura 4.40.

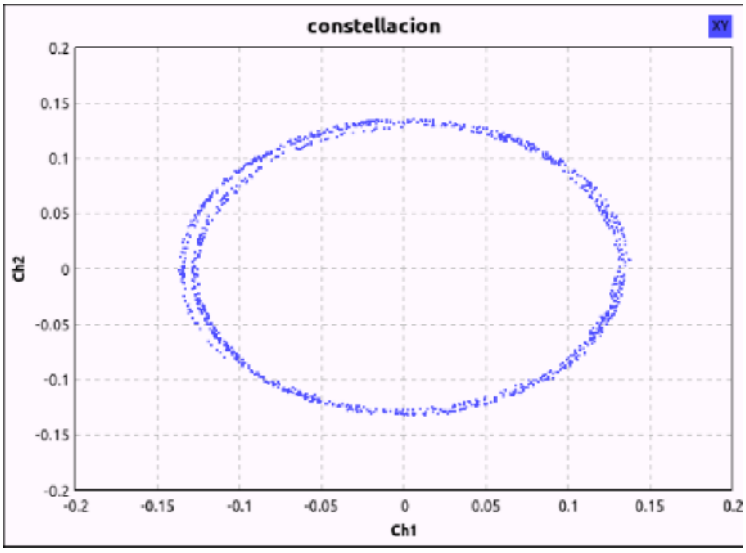

Figura 4.40. Diagrama de constelación recibido para GMSK.
Fuente: Imagen propia de los autores.

Finalmente, luego de realizar el proceso de compensación y demodulación se obtiene el espectro mostrado en la Figura 4.41.

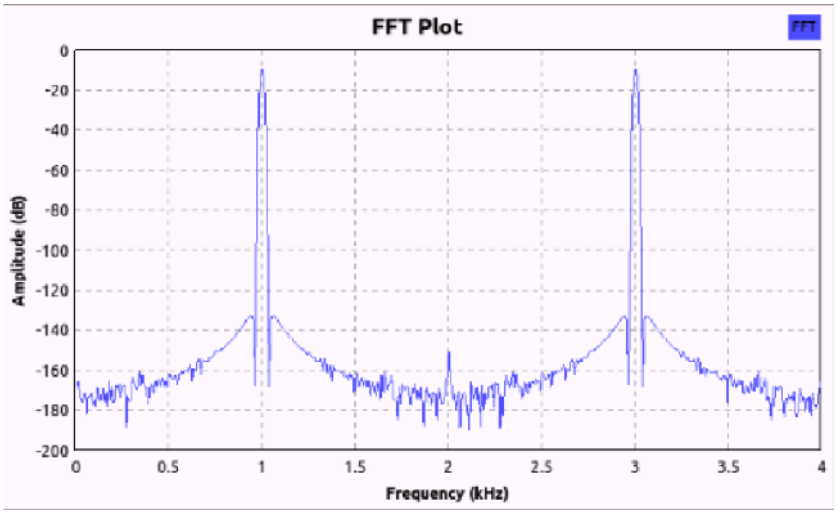

Figura 4.41. Espectro de audio recibido y demodulado.
Fuente: Imagen propia de los autores.

Como podemos observar en la Figura 4.41 el proceso de recepción y demodulación ha sido realizado correctamente.

4.4 Sistema de comunicación experimental para la transmisión y recepción digital de video

Se implementó un sistema de transmisión y recepción de video en base al sistema de transmisión-recepción digital y con la ayuda de la herramienta de *software Gstreamer*, que se utilizó para capturar y codificar el video de la *webcam* del computador en formato *mpeg-4*; esta señal de información fue procesada y transmitida usando la modulación GMSK a través del USRP. Además se empleó la herramienta de *software Mplayer* para la parte de recepción para decodificar y reproducir la señal de video recibida.

4.4.1 Transmisor digital de video.

Primeramente, se debe crear un archivo con extensión *.ts* (*transport stream*) en el computador que funcionará en conjunto con un USRP N210 como transmisor. Este archivo establecerá un conducto entre *Gstreamer* y GNU Radio *Companion* para el flujo continuo de video desde la *webcam* del computador.

La creación de este archivo se realiza en la terminal de Ubuntu mediante el comando:

```
# mkfifo txfifo.ts
```

Para iniciar el flujo de video se emplea el siguiente comando:

```
# gst-launch v4l2src device=/dev/video0 ! video/x-raw-yuv, width=640, height=480 ! timeoverlay halign=right valign=bottom shaded-background=true ! textoverlay text="Video de Prueba 30fps 640*480 UTPL" halign=left valign=bottom shaded-background=true ! ffmpegcolorspace ! x264enc bitrate=420 ! mpegtsmux ! filesink location=txfifo.ts
```

Lista 4.1 Comando para iniciar flujo de video desde la *webcam.*

Mediante este comando se configura la fuente de video que es **video0** correspondiente a la *webcam* del computador, la resolución del video a transmitir, la cantidad de fotogramas por segundo, el *encoder* de video y su tasa de bits.

Como podemos observar mediante este comando se alista el conducto para recibir un flujo de video con una resolución de 640*480 a 30 fotogramas por segundo como se muestra en la Figura 4.42.

En la Figura 4.43 se muestra el esquema del transmisor de video diseñado en GNU Radio *Companion*.

```
root@alex-HP-Pavilion-dv5-Notebook-PC:/home/alex# cd GP4/
root@alex-HP-Pavilion-dv5-Notebook-PC:/home/alex/GP4# gst-launch v4l2src device
/dev/video0 ! video/x-raw-yuv,width=640,height=480 !    timeoverlay halign=righ
 valign=bottom shaded-background=true !   textoverlay text="Video de Prueba 30f
s 640*480 UTPL" halign=left valign=bottom shaded-background=true !  ffmpegcolor
pace ! x264enc bitrate=420 ! mpegtsmux ! filesink location=txfifo.ts
Estableciendo el conducto a PAUSA …
```

Figura 4.42. Inicialización del conducto **txfifo.ts** para el flujo de video.
Fuente: Imagen propia de los autores.

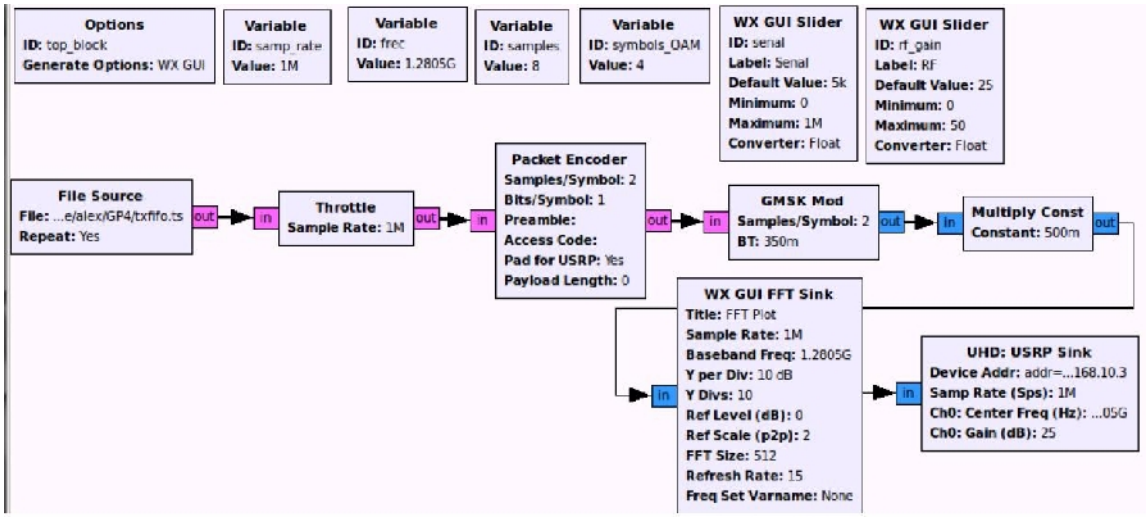

Figura 4.43. Esquema del transmisor digital de video.
Fuente: Imagen propia de los autores.

Como podemos observar en la Figura 4.43 en el bloque **File Source** se selecciona el archivo correspondiente al conducto creado para el flujo de video. La configuración de este bloque se muestra en la Figura 4.44.

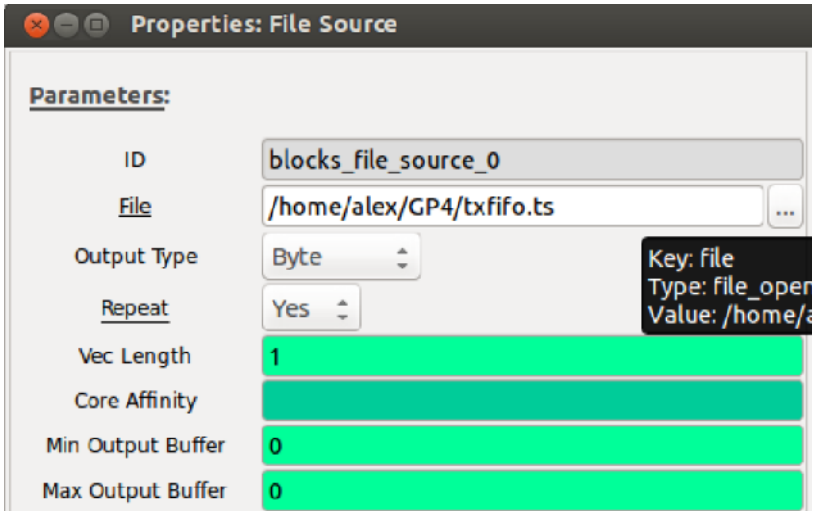

Figura 4.44. Configuración del bloque **File Source**.
Fuente: Imagen propia de los autores.

Al ejecutar el módulo del transmisor el conducto cambiará a modo *REPRODUCIENDO* como se aprecia en la Figura 4.45 y comenzará el *streaming* de video generado desde la *webcam* y transmitido en modulación GMSK a través del USRP, como se muestra en la Figura 4.46.

```
root@alex-HP-Pavilion-dv5-Notebook-PC:/home/alex/GP4# gst-launch v4l2src device=
/dev/video0 ! video/x-raw-yuv,width=640,height=480 !    timeoverlay halign=right
 valign=bottom shaded-background=true !   textoverlay text="Video de Prueba 30fp
s 640*480 UTPL" halign=left valign=bottom shaded-background=true !  ffmpegcolors
pace ! x264enc bitrate=420 ! mpegtsmux ! filesink location=txfifo.ts
Estableciendo el conducto a PAUSA …
El conducto está vivo y no necesita PREPARARSE …
Estableciendo el conducto a REPRODUCIENDO …
New clock: GstSystemClock
```

Figura 4.45. Conducto en modo *REPRODUCIENDO*.
Fuente: Imagen propia de los autores.

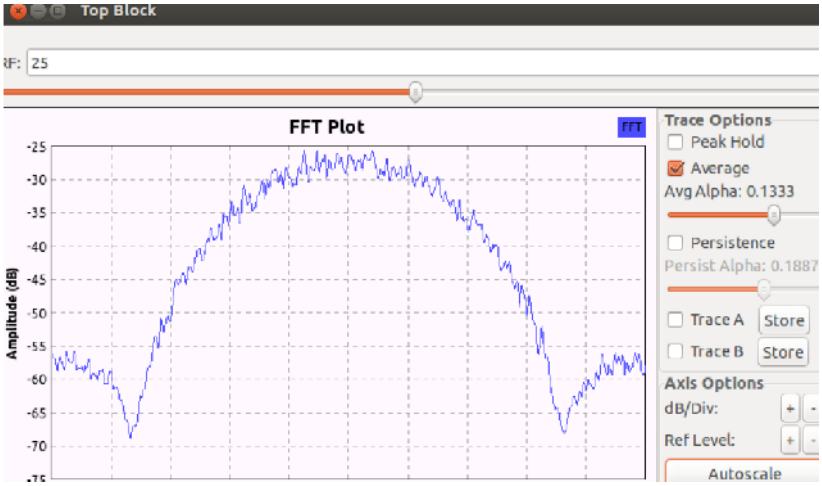

Figura 4.46. Transmisión de video con modulación GMSK en la frecuencia de 1.2GHz.
Fuente: Imagen propia de los autores.

4.4.2 Receptor digital de video.

Para la recepción del flujo de video es necesario crear otro archivo en formato *.ts* en el computador que funcionará en conjunto con un USRP N210 como receptor que en este caso cumplirá la función de conducto entre GNU Radio *Companion* y *Mplayer***.**

La creación de este archivo se realiza en la terminal de Ubuntu mediante el comando: # mkfifo rxfifo.ts

En la Figura 4.47 se muestra el esquemático del receptor digital de video en GNU Radio *Companion.*

En el bloque **File Sink** se selecciona el archivo correspondiente al conducto creado para el flujo de video.

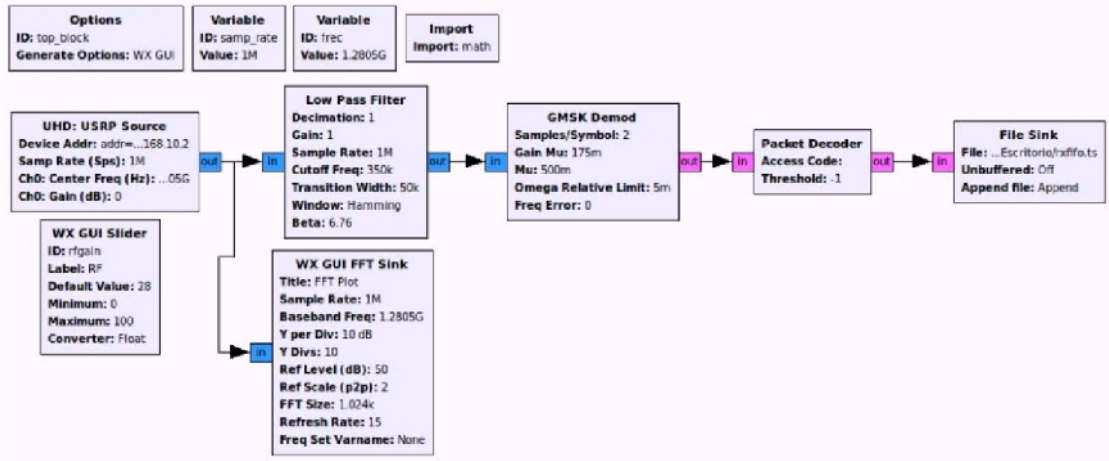

Figura 4.47. Receptor digital de video en GNU Radio *Companion*.
Fuente: Imagen propia de los autores.

Seguidamente se prepara este conducto para la recepción del mismo mediante el comando:
mplayer rxfifo.ts

El conducto listo se puede observar en la Figura 4.48.

```
root@Max-Peralta:/home/max/Escritorio# mplayer rxfifo.ts
MPlayer svn r34540 (Ubuntu), built with gcc-4.7 (C) 2000-2012 MPlayer Team
mplayer: could not connect to socket
mplayer: No such file or directory
Failed to open LIRC support. You will not be able to use your remote control.

Playing rxfifo.ts.
```

Figura 4.48. Preparación del conducto rxfifo.ts.
Fuente: Imagen propia de los autores.

Al correr el módulo del receptor el conducto detectará el formato del flujo de datos entrante que es *.ts* como se aprecia en la Figura 4.49. Luego de esto se debe esperar unos segundos para sincronización y almacenamiento de datos en los *buffers* del computador y si la demodulación fue realizada correctamente en GNU Radio el programa reconocerá el video recibido en formato *mpeg-4* y comenzará la reproducción del mismo.

```
TS file format detected.
Cannot seek backward in linear streams!
Seek failed
```

Figura 4.49. Detección del formato *.ts* en el receptor.
Fuente: Imagen propia de los autores.

En la Figura 4.50 se muestra el espectro de la señal recibida en la frecuencia 1. 2805GHz.

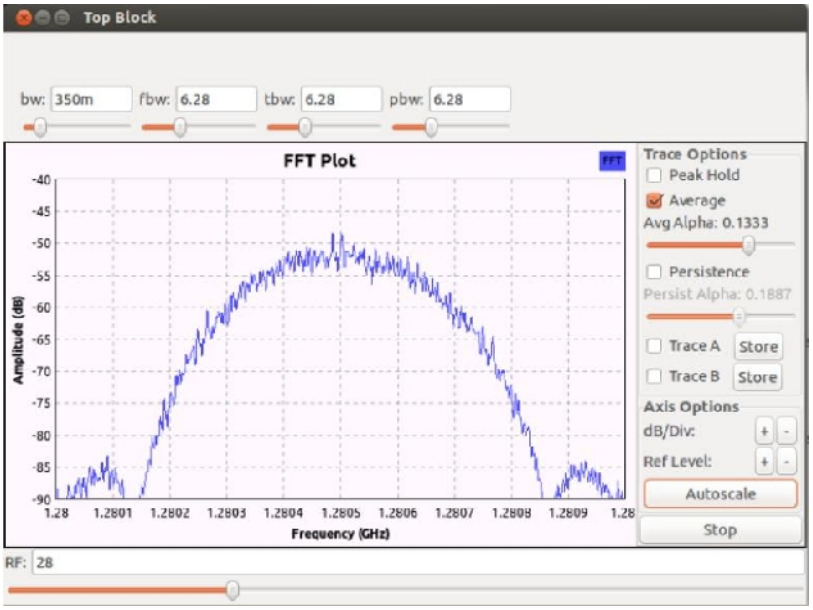

Figura 4.50. Espectro recibido del transmisor digital de video en la frecuencia 1.2805GHz.
Fuente: Imagen propia de los autores.

Finalmente en la Figura 4.51 se observa la señal de video demodulada y decodificada.

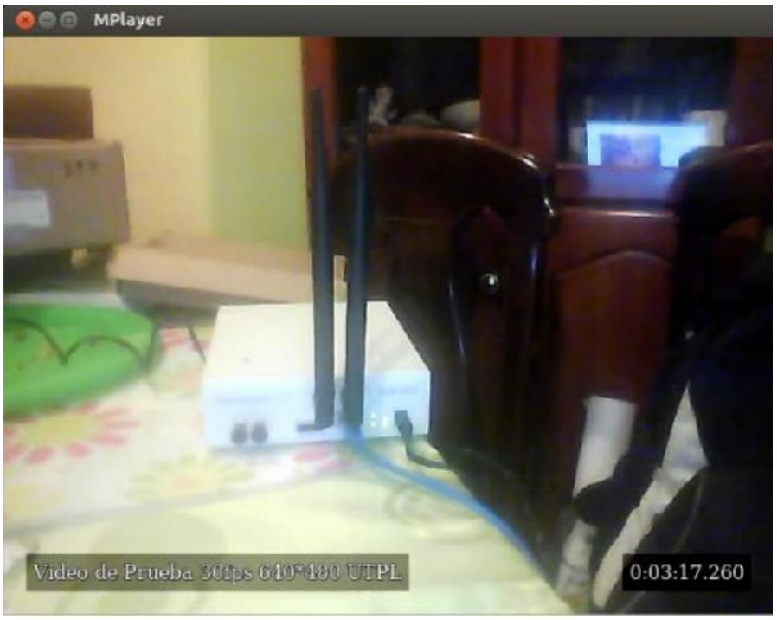

Figura 4.51. Señal de video demodulada y decodificada.
Fuente: Imagen propia de los autores.

Si se requiere una aplicación en la que se desea transmitir video con audio, es posible hacerlo realizando una simple modificación en el comando para generar el flujo de video; añadiendo una instrucción para integrar audio en formato AAC (*advanced audio coding*), como se puede observar en la lista 4.2.

```
# gst-launch v4l2src device=/dev/video0 ! video/x-raw-yuv, width=640, height=480 ! timeoverlay halign=right valign=bottom shaded-background=true ! textoverlay text="Video de Prueba con Audio 30fps 640*480 UTPL" halign=left valign=bottom shaded-background=true ! ffmpegcolorspace ! x264enc bitrate=420 ! muxout. pulsesrc ! queue ! audioconvert ! voaacenc bitrate=8000 ! aacparse ! muxout. mpegtsmux name=muxout ! queue ! filesink location=txfifo.ts
```

Lista 4.2 Comando para iniciar flujo de video con audio desde la *webcam.*

El resto del sistema es el mismo tanto en el transmisor como en el receptor, los resultados obtenidos se muestran en la Figura 4.52.

```
VIDEO H264(pid=64) AUDIO AAC(pid=65) NO SUBS (yet)!  PROGRAM N. 1
Cannot seek backward in linear streams!
Seek failed
FPS seems to be: 2000000000.000000
Load subtitles in ./
Failed to open VDPAU backend libvdpau_nvidia.so: cannot open shared object file:
 No such file or directory
[vdpau] Error when calling vdp_device_create_x11: 1
==========================================================================
Opening video decoder: [ffmpeg] FFmpeg's libavcodec codec family
libavcodec version 53.35.0 (external)
Mismatching header version 53.32.2
Selected video codec: [ffh264] vfm: ffmpeg (FFmpeg H.264)
==========================================================================
==========================================================================
Opening audio decoder: [ffmpeg] FFmpeg/libavcodec audio decoders
AUDIO: 44100 Hz, 1 ch, s16le, 0.0 kbit/0.00% (ratio: 0->88200)
Selected audio codec: [ffaac] afm: ffmpeg (FFmpeg AAC (MPEG-2/MPEG-4 Audio))
==========================================================================
AO: [pulse] 44100Hz 1ch s16le (2 bytes per sample)
Starting playback...
```

Figura 4.52. Señal decodificada de video con audio.
Fuente: Imagen propia de los autores.

Obteniendo así la señal de video con audio correctamente demodulada y decodificada como se aprecia en la Figura 4.53.

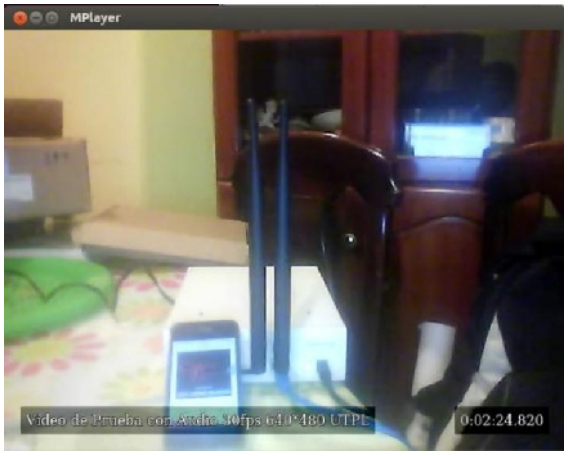

Figura 4.53. Reproducción de video con audio.
Fuente: Imagen propia de los autores.

4.5 Cálculo de la tasa de error de bit (BER)

Una vez implementados los sistemas de comunicaciones digitales; se han realizado pruebas del porcentaje de error obtenido en transmisiones de imágenes con diferentes niveles de SNR (relación señal-ruido). Para ello se ha empleado una modulación GMSK y se ha transmitido una imagen en escala de grises con la finalidad de poder representarla en formato uint8 y que cada pixel tenga valores representados en 8 bits. Debido a que en GNU Radio *Companion* no es posible modificar directamente el valor de SNR, se lo puede realizar indirectamente mediante un bloque de modelamiento de canal (**Channel model**) agregando ruido *gaussiano* con amplitud variable a la señal modulada.

Para la obtención del porcentaje de error se ha realizado un programa en Matlab en el cual se compara la imagen transmitida con la recibida, se obtiene la imagen de error entre las dos imágenes restándolas en valor absoluto. Finalmente se procede a calcular el porcentaje de error de bit sobre esta imagen de error dividiendo la cantidad de bits errados para la cantidad total de bits de la imagen original y obteniendo el porcentaje.

En la figura 4.54 se muestra el espectro de la transmisión GMSK (en 1.2GHz) con un valor de ruido en el bloque **Channel model** de 10mV.

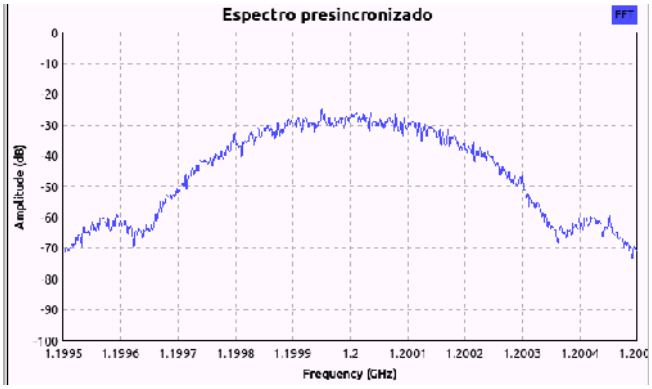

Figura 4.54. Espectro GMSK con nivel de ruido 10mV en el bloque **Channel model**.
Fuente: Imagen propia de los autores.

En la figura 4.55 se observa la imagen original transmitida y la imagen recibida.

a) b)

Figura 4.55. a) imagen transmitida, b) imagen recibida.
Fuente: Tomada de [38].

En la figura 4.56 se muestra el espectro de la transmisión GMSK (en 1.2GHz) con un valor de ruido en el bloque **Channel model** de 80mV.

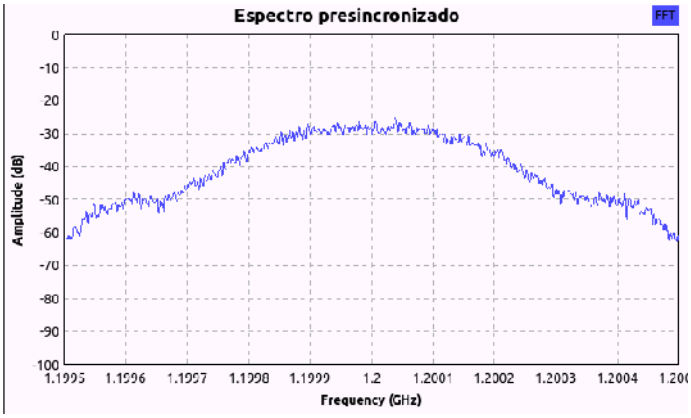

Figura 4.56. Espectro GMSK con nivel de ruido 80mV en el bloque **Channel model**.
Fuente: Imagen propia de los autores.

En la figura 4.57 se observa la imagen original transmitida y la imagen recibida.

a) b)

Figura 4.57. a) imagen transmitida, b) imagen recibida.
Fuente: Tomada de [38].

En la tabla 4.1 se presenta un resumen de los resultados obtenidos en las pruebas.

Tabla 4.5. Resultados obtenidos para el porcentaje de error de bit.

Porcentaje de error de bit	
Voltaje de ruido (mV)	BER (%)
10mV	0.0262%
20mV	0.0262%
30mV	0.9900%
40mV	3.1971%
50mV	6.7550%
60mV	5.5653%
70mV	10.6394%
80mV	13.9024%

Fuente: Tabla propia de los autores.

Como se puede observar el porcentaje de error aumenta conforme aumenta el nivel de ruido; sin embargo esta estimación deja de ser válida a niveles más altos de ruido debido a que en el receptor el bloque **Packet Decoder** detiene su salida de datos si se producen un número determinado de errores en el código de acceso como se analizó en el anexo B.

CONCLUSIONES

- La tecnología SDR está definida por dos partes que son hardware y software; la primera es la encargada de transformar las señales de radiofrecuencia a banda base o frecuencia intermedia (IF) por medio de la tarjeta secundaria; y la segunda parte es la encargada del procesamiento digital de señales en banda base que se realiza en el computador, logrando así implementar sistemas de comunicaciones tanto analógicos como digitales con hardware reducido.

- El equipo SDR USRP N210 opera en un amplio rango de frecuencias (50 - 6000MHz); dependiendo de las características de la tarjeta secundaria empleada. Hemos empleado la tarjeta secundaria WBX que opera en un rango de frecuencias de 50 a 2200MHz, permitiéndonos desarrollar sistemas de comunicaciones analógicos y digitales en las bandas VHF y UHF.

- Se demuestra que mediante la tecnología SDR es posible implementar sistemas de comunicaciones analógicos y digitales de bajo costo comparado con los sistemas tradicionales y con un tiempo reducido de despliegue ya que solo es necesario la conexión del USRP con el computador, esto se logra al sustituir algunos componentes de *hardware* por rutinas de *software*.

- Se debe desarrollar los sistemas de comunicaciones en un entorno de simulación a través de fuentes y sumideros virtuales en GNU Radio *Companion* con la finalidad de probar su correcto funcionamiento y luego de esto proceder a su migración a plataformas SDR; es decir, haciendo uso de dispositivos de *hardware*.

- Al migrar los sistemas de comunicaciones a un entorno real se presentarán efectos indeseados en el canal de telecomunicaciones como: ruido, distorsión, desplazamiento de fase, frecuencia, etc.; siendo importante simular estos efectos aisladamente para conocer su influencia sobre las señales de RF y poder reducirlos a niveles que no afecten el correcto funcionamiento de los sistemas.

- En la programación de los sistemas de comunicaciones es necesario establecer parámetros de diseño como la tasa de muestreo, los índices y tipos de modulación, la cantidad de bits y muestras por símbolo de las diferentes modulaciones a emplearse; en el caso de la tasa de muestreo es posible utilizar operaciones de decimación e interpolado con la finalidad de acoplar la señal a las diferentes etapas de procesamiento.

RECOMENDACIONES

- Se recomienda realizar actualizaciones y mejoras del sistema operativo mediante los comandos *update* y *upgrade* antes de la instalación de programas o *drivers* que se requieran para la implementación de los sistemas.

- Es importante abrir el entorno gráfico GNU Radio *Companion* mediante en modo de usuario privilegiado de tal manera que contemos con todos los permisos para la utilización del mismo.

- Como alternativa a la programación en el lenguaje Python se puede emplear el entono gráfico de GNU Radio denominado GNU Radio *Companion*, en el cual se realiza una programación por bloques con funcionalidades específicas.

- Para obtener un mayor alcance en los sistemas de comunicaciones se recomienda emplear antenas que operen en bandas determinadas de acuerdo a cada aplicación.

- Se emplearon herramientas software adicionales como **GStreamer** y **MPlayer** para la obtención, codificación - decodificación y reproducción de la señal de video en formato **mpeg-4**.

REFERENCIAS

[1] H. Arslan, *Cognitive Radio, Software Defined Radio, and Adaptive Wireless Systems*, ch 4, pp. 129 - 138. University of South Florida, Tampa, FL, USA: Springer, 2007.

[2] "*Software Radio Laboratory LLC*." [en línea], disponible en: http://www.srl-llc.com/. [Consulta del 23/09/2014].

[3] WiNRADiO Comunications, "*WiNRADiO*" [en línea], disponible en: http://www.winradio.com/home/g31ddc.htm. [Consulta del 23/09/2014].

[4] "*FiFi-SDR*" [en línea], disponible en: http://o28.sischa.net/fifisdr/trac. [Consulta del 23/09/2014].

[5] Nuand LLC, "*bladeRF - the USB 3.0 Superspeed Software Defined Radio*" [en línea], disponible en: http://www.nuand.com/. [Consulta del 23/09/2014].

[6] Micrisat, "*easySDR USB Dongle*" [en línea], disponible en: http://microsat.com.pl/product_info.php?products_id=35. [Consulta del 23/09/2014].

[7] M. Ossmann, "*HackRF, an open source SDR platform*" [en línea], disponible en: https://www.kickstarter.com/projects/mossmann/hackrf-an-open-source-sdr-platform. [Consulta del 07/10/2014].

[8] Ettus Research, "*USRP N200/N210 NETWORKED SERIES*" [en línea], disponible en: https://www.ettus.com/content/files/07495_Ettus_N200-210_DS_Flyer_HR_1.pdf. [Consulta del 23/09/2014].

[9] National Instruments, "*Out-of-the-Box Video With NI USRP*" [en línea], disponible en: http://www.ni.com/white-paper/13880/en/. [Consulta del 23/09/2014].

[10] MathWorks, "*USRP Support Package from Communications System Toolbox*" [en línea], disponible en: http://www.mathworks.com/hardware-support/usrp.html. [Consulta del 23/09/2014].

[11] "Modulación de Frecuencia-FM" [en línea], disponible en: http://www.textoscientificos.com/redes/modulacion/frecuencia. [Consulta del24/09/2014].

[12] F. Montealegre Alfaro, "*Instalación, puesta en funcionamiento de tarjeta de BER (razón de bits erróneos) en generador de alta frecuencia y elaboración de prácticas de laboratorio de Telecomunicaciones*", Universidad de Costa Rica, Diciembre 2007.

[13] "*Estudio y comparación en eficiencia espectral y probabilidad de error de los esquemas de modulación GMSK y DBPSK*" [en línea], disponible en: http://www.scielo.org.co/scielo.php?pid=S0120-56092008000300010&script=sci_arttext. [Consulta del 24/09/2014].

[14] "*WBX 50-2200 MHz Rx/Tx (40MHz)*" [en línea], disponible en: https://www.ettus.com/product/details/WBX. [Consulta del 28/09/2014].

[15] D. Tucker and G. Tagliarini, "*Prototyping with gnu radio and the usrp – where to begin*", in Southeastcon, 2009. SOUTHEASTCON '09. IEEE, pp. 50-54, March 2009.

[16] V. Rodríguez and J. Sánchez, "*Empowering software radio: It++ as a gnu radio out-of-tree implementation*", Latin America Transactions, IEEE (Revista IEEE America Latina), vol. 12, pp. 269276, March 2014.

[17] A. Galvis Quintero, C. A. Ceballos Betancour, and L. De Sanctis Gil, "*SDR: La alternativa para la evolución inalámbrica a nivel físico*", Universidad Pontificia Bolivariana - Medellín.

[18] J. P. Montero Hidalgo, "*Implementación de un Sistema de Comunicaciones basado en Software Radio*", Departamento de Radiofrecuencia. Escuela Politécnica Superior y Universidad Autónoma de Madrid, Enero 2014.

[19] J. H. Aguilar Rentería and A. Navarro Cadavid, "*Radio cognitiva - Estado del arte*", Sistemas y Telemática; Vol.9 No.16, 2011.

[20] M. Abirami, V. Hariharan, M. Sruthi, R. Gandhiraj, and K. Soman, "*Exploiting gnu radio and usrp: An economical test bed for real time communication systems*", in Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on, pp. 1-6, July 2013.

[21] F. Bruce, "*Cognitive Radio Technology"*, ch. 1, pp. 1-17. ELSEVIER, 1 ed., 2006.

[22] T. Wayne, "*Sistemas de Comunicaciones Electrónicas*", pp. 864-901. 5 ed., 2005.

[23] GNURadio, "*What is GNU Radio and why do I want it*". [en línea], disponible en: http://gnuradio.org/redmine/projects/gnuradio/wiki/WhatIsGR. [Consulta del 23/09/2014].

[24] "*Modulación y Transmisión de Datos*" [en línea], disponible en: http://sistemas.uniandes.edu.co/~isis1301/dokuwiki/lib/exe/fetch.php?media=recursos:06_modulacion.pdf. [Consulta del 24/09/2014].

[25] C. P. Vega, "*Modulación de Amplitud*". Ed. Universidad de Cantabria.

[26] "*Laurent Moise Schwartz*" [en línea], disponible en: http://www-history.mcs.st-and.ac.uk/Biographies/Schwartz.html. [Consulta del 26/09/2014].

[27] M. Faúndes Zanuy, "*Sistemas de Comunicaciones*", ch. 7, pp. 117-173. MARCOMBO, S.A., 2001.

[28] E. A. Quintero, A. Agudelo, and P. C. Bernal, "*Modulación gmsk para transmisión de información a través de líneas eléctricas*", Scientia et Technica, vol. 3, no. 46, pp. 86-90, 2010.

[29] Z. Tong, M. Arifianto, and C. Liau, "*Wireless transmission using universal software radio peripheral*", in Space Science and Communication, 2009. IconSpace 2009. International Conference on, pp. 19-23, IEEE, 2009.

[30] "USRP N210" [en línea], disponible en: https://www.ettus.com/product/details/UN210-KIT. [Consulta del 24/09/2014].

[31] G. Eichinger, K. Chowdhury, and M. Leeser, "*Crush: Cognitive radio universal software hardware*", in Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on, pp. 2632, IEEE, 2012.

[32] "Multivibradores" [en línea], disponible en: http://www.areaelectronica.com/circuitos-basicos/multivibradores.htm. [Consulta del 26/09/2014].

[33] K. Dabcevic, "*Evaluation of software defined radio platform with respect to implementation of 802.15. 4 zigbee*", 2011.

[34] "*Installing the Ettus Research GPSDO Kit for USRP N200 Series/E100 Series*" [en línea], disponible en: https://www.ettus.com/content/files/gpsdo-kit_4.pdf. [Consulta del 24/09/2014].

[35] W. Taymans, S. Baker, A. Wingo, R. S. Bultje and S. Kost, "*GStreamer Application Development Manual (1.4.3)*" [en línea], disponible en: http://gstreamer.freedesktop.org/data/doc/gstreamer/head/manual/manual.pdf. [Consulta del 01/10/2014].

[36] "*GStreamer open source multimedia framework*" [en línea], disponible en: http://gstreamer.freedesktop.org/features/. [Consulta del 01/10/2014].

[37] "*MPlayer*" [en línea], disponible en: http://www.guia-ubuntu.com/index.php?title=MPlayer. [Consulta del 01/10/2014].

[38] E. Manso Castillo, J. Correro Fernández, E. Velasquez Gil, P. Delgado Salas y E. García Fernández, "*Trabajo Proyecto Integrado 4º ESO IES Torre Almirante*", [en línea], disponible en: http://proyecto4esoiestorrealmirante.blogspot.com/. [Consulta del 30/11/2014]

ANEXOS

ANEXO A

INSTALACIÓN DE GNU RADIO *COMPANION*

GNU Radio es un *software* de desarrollo de herramientas de código libre y abierto que proporciona bloques de procesamiento de señales para implementar *software* radio. Esta plataforma está disponible para tres diferentes sistemas operativos como son: Linux, Windows y MAC OS X y para esta investigación se ha usado Linux y el sistema operativo Ubuntu con su versión 13.04; para Linux existen dos formas de instalar GNU Radio *Companion*, una que es directamente a través de un comando que se escribe en la terminal del sistema y es: # apt-get install gnuradio.

Pero esta instalación es sumamente rápida y puede que la versión este desactualizada así como no se pueden instalar todos los paquetes.

La instalación mencionada anteriormente puede ser necesaria para la mayoría de usuarios y para personas que recién comiencen en el uso de GNU Radio, pero si se necesita requisitos o algunos paquetes especiales es mejor instalarlo desde la fuente, que es nuestro caso.

Hay dos formas de instalar GNU Radio *Companion* desde la fuente: ya sea mediante el uso de paquetes binarios pre-compilados, o manualmente compilando desde el código fuente. En nuestro caso instalaremos los paquetes binarios pre-compilados, para ello tenemos que descargarnos el *script* que contiene el código fuente como lo observamos en la Figura A.1

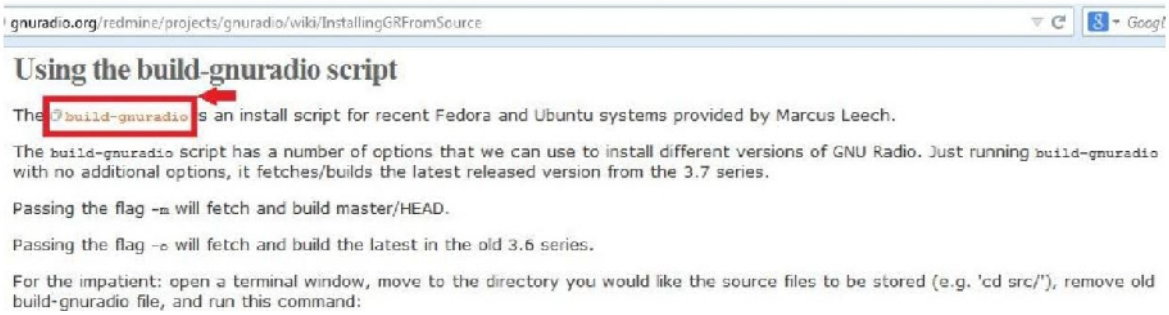

Figura A.1. Descargar *script* de código fuente de GNU Radio *Companion.*
Fuente: Imagen propia de los autores.

A través de la terminal en Ubuntu procedemos a ubicar el archivo en la carpeta que se guardó y por medio del comando: chmod a+x build-gnuradio, procedemos a darle los respectivos permisos de lectura y escritura para poder ejecutarlo y se pueda compilar con el comando: ./build-gnuradio.

Como observamos en la Figura A.2 una vez realizados los pasos anteriores se procede con la instalación de GNU Radio *Companion* y nos da una aproximación del espacio en el disco que necesitará así como el tiempo estimado para completar dicho proceso, los comandos mencionados anteriormente se los coloca de forma global, es decir no en el modo sudo.

```
alex@ubuntu:-$ cd Descargas/
alex@ubuntu:~/Descargas$ ls
build-gnuradio
alex@ubuntu:~/Descargas$ chmod a+x build-gnuradio
alex@ubuntu:~/Descargas$ ./build-gnuradio
This script will install Gnu Radio from current GIT sources
You will require Internet access from the computer on wich this
script runs. You will require SUDO access. You will require
approximately 500MB of free disk space to perform the build.

This script will, as a side-effect, remove any existing Gnu Radio
installation that was installed from your Linux distribution packages.
It must do this to prevent problems due to interference between
a linux-distribution-installed Gnu Radio/UHD and one installed from GIT source.

The whole process may take up to two hours to complete, depending on the
capabilities of your system.
```

Figura A.2. Permisos y compilación del código fuente de GNU Radio.
Fuente: Imagen propia de los autores.

Para la instalación de todos los componentes de GNU Radio *Companion* se requiere privilegios de administrador, se procedemos a colocar la letra “y” además de colocar la contraseña del usuario, el computador comenzará la instalación de todos los componentes del código fuente como podemos observar en la Figura A.3.

```
intelligent. It tries to make life a little easier for you, but at the end of the
day
if it runs into truble, a certain amount of knowledge on your part about
system configuration and idiosyncrasies will inervitably be necessary


Proced?y
Starting all functions at: mié sep 10 18:22:43 ECT 2014
SUDO privileges are required
Doyou have SUDO privileges?y
Continuing wiyh script
[sudo] password for alex:
Installing prerequisites.
====> THIS MAY TAKE QUITTE SOME TIME <====
Checking for package libfantconfig1-dev
Checking for package libxreodec-dev
Checking for package swig
Checking for package g++
Checking for package automake
Checking for package autoconf
Checking for packagelibtool
Checking for package python-dev
```

Figura A.3. Instalación de GNU Radio *Companion*.
Fuente: Imagen propia de los autores.

ANEXO B

MODELADO DEL CANAL PARA MODULACIONES DIGITALES

En este anexo se presenta la implementación de un programa diseñado en GNU Radio *Companion* mediante el cual, se simularán los efectos más significativos que sufre una señal digital transmitida por un medio inalámbrico, como son: ruido, multitrayecto, desvanecimiento, desviaciones en frecuencia, fase o diferencias en los tiempos de muestreo.

Este apartado es muy importante al momento de pasar de un entorno de simulación a un entorno real con los equipos USRP, debido a que se presentarán los efectos indeseables mencionados; de tal manera que se simularán cada uno de ellos, se observará su efecto sobre las señales y se mostrará la manera de reducirlos o solucionarlos para poder implementar correctamente los sistemas de comunicaciones propuestos.

En la Figura B.1 se presenta el esquema del programa realizado.

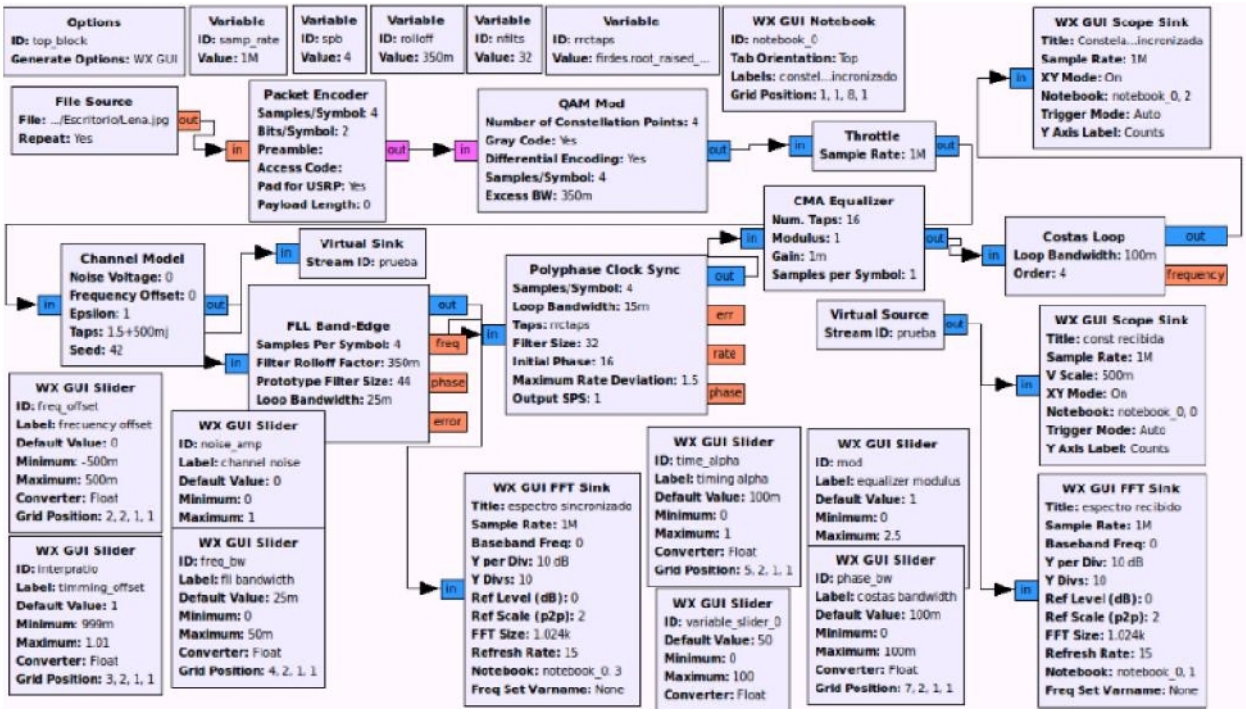

Figura B.1. Programa para modelado de canal y sincronismo.
Fuente: Imagen propia de los autores.

El bloque que nos permite simular todos estos efectos es **Channel Model**, el cual consta de los siguientes parámetros:

- **Noise Voltaje** que representa el nivel de ruido Gaussiano como un voltaje.
- **Frequency Offset** que representa el desplazamiento en frecuencia.
- **Epsilon** que emula una diferencia de los tiempos de muestreo entre el transmisor y el receptor.
- **Taps** que emula los multitrayectos de la señal transmitida.

B.1 Ruido

En la Figura B.2 se puede observar el espectro de una señal QAM correctamente sincronizada en tiempo, fase y frecuencia.

Figura B.2. Espectro de una señal QAM correctamente sincronizada.
Fuente: Imagen propia de los autores.

En la Figura B.3 se puede observar el diagrama de constelación de la señal en cuestión.

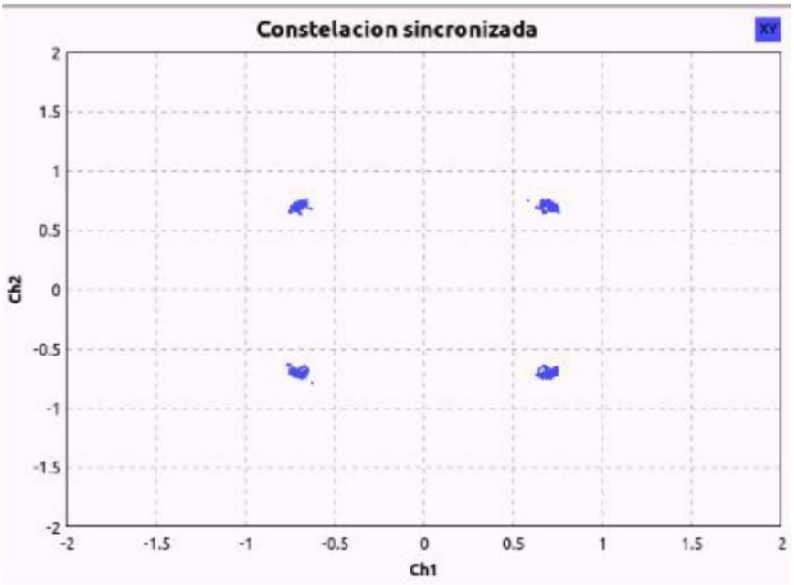

Figura B.3. Diagrama de constelación de la señal QAM sincronizada.
Fuente: Imagen propia de los autores.

Modificando el parámetro **Noise Voltaje** se puede aumentar o disminuir el voltaje de una fuente de ruido gaussiano (AWGN) que se adiciona a la señal. En la Figura B.4 se puede observar la señal QAM con un voltaje de ruido de 300mV y el efecto producido.

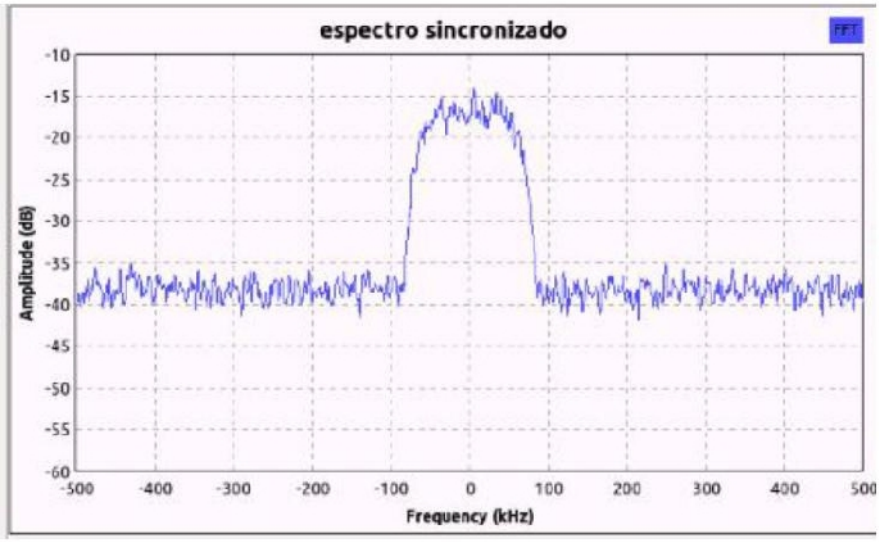

Figura B.4. Espectro de la señal QAM sincronizada con un nivel de ruido de 300mV.
Fuente: Imagen propia de los autores.

En la Figura B.5 se muestra el efecto del ruido sobre el diagrama de constelación.

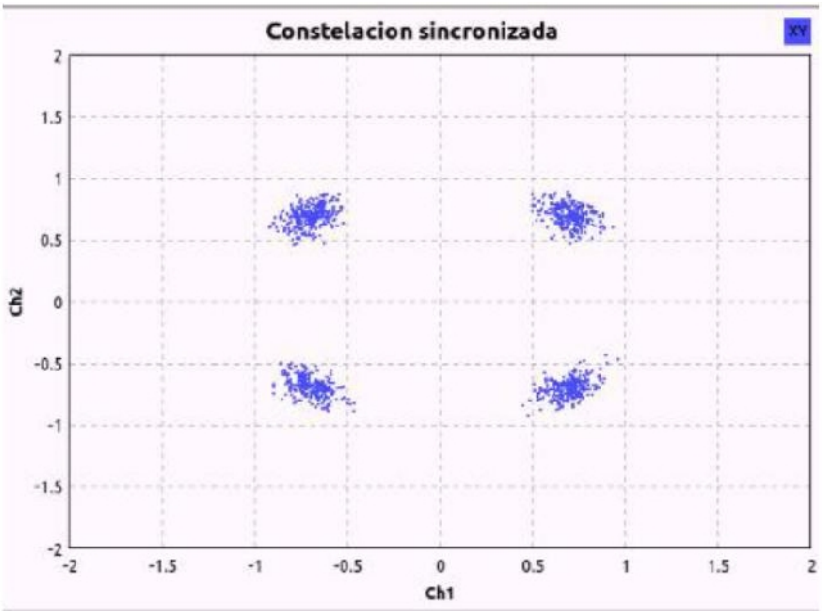

Figura B.5. Diagrama de Constelación de la señal QAM sincronizada con un nivel de ruido de 300mV.
Fuente: Imagen propia de los autores.

B.2 Sincronización

En esta sección se analizarán los tipos de sincronismos empleados en los receptores digitales, entre los cuales se analizan los siguientes:

- Sincronización de paquete.
- Sincronización de portadora (frecuencia y fase).
- Sincronización de reloj (recuperación de señal de reloj).

B.2.1 Sincronización de paquete.

La sincronización de paquete tiene por objetivo indicar al receptor cuando comienza la información útil, para ello se utiliza un código prefijado llamado preámbulo al inicio de la transmisión.

El bloque encargado de esta función es el **Packet Encoder**. Este bloque, introduce un preámbulo de 16 bits (0xA4F2) además de introducir un código de acceso mediante el cual se controla que no se propague la salida si ocurren más errores del prefijado, esto se realiza mediante el umbral. Por defecto, esto ocurrirá si acontecen más de 12 errores en el código de acceso cuya longitud es de 64 bits (0xACDDA4E2F28C20FC) [18].

El paquete estará conformado por el preámbulo seguido del código de acceso, de la longitud del *payload* y finalmente del código CRC (código de detección de errores) como se muestra en la Figura B.6.

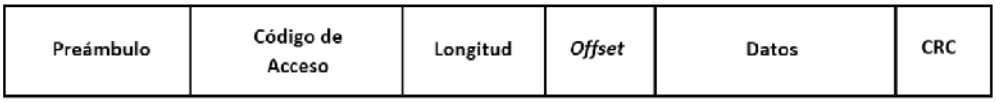

Figura B.6. Paquete generado por el **Packet Encoder**.
Fuente: Imagen propia de los autores.

B.2.2 Sincronización de portadora.

La sincronización de portadora tiene por objetivo compensar las diferencias en cuanto a fase y frecuencia entre los osciladores del transmisor y receptor así como los desplazamientos que haya sufrido la propia portadora debido a su propagación por el medio (efecto *doppler*). Por lo tanto, este tipo de sincronismo juega un papel importante en la demodulación coherente.

B.2.2.1 Sincronización de frecuencia.

En esta sección se muestran los efectos que produce un desplazamiento en frecuencia en la recepción de una señal digital QAM. Para ello, se modifica el parámetro **Frequency Offset**.

En la Figura B.7 se muestra el espectro de la señal QAM desplazada en frecuencia 100KHz.

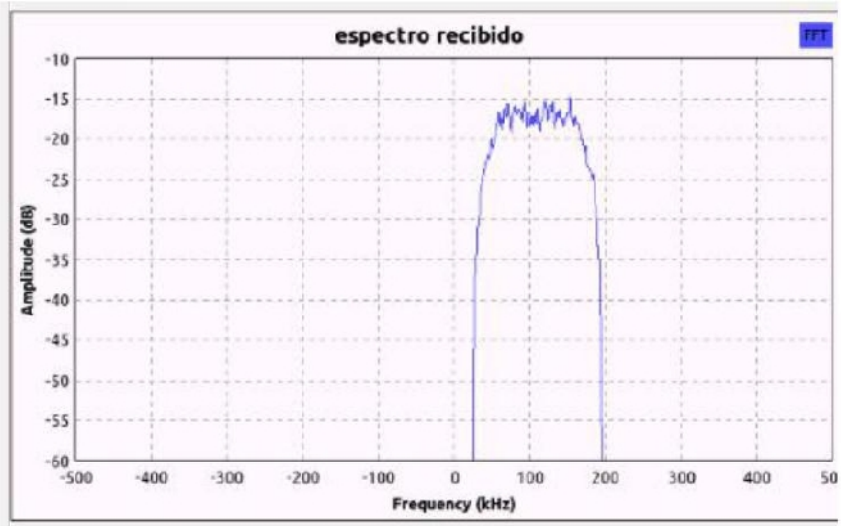

Figura B.7. Espectro de la señal QAM desplazada en frecuencia.
Fuente: Imagen propia de los autores.

En la Figura B.8 se muestra el diagrama de constelación de la señal QAM antes de la sincronización y filtrado.

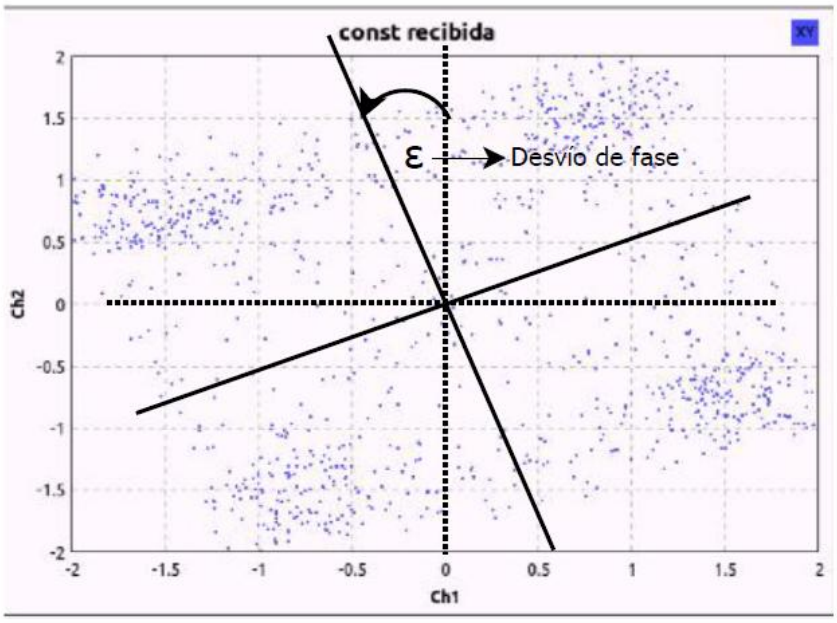

Figura B.8. Diagrama de constelación de la señal QAM desfasado.
Fuente: Imagen propia de los autores.

Como se puede observar en la Figura B.8, se muestra el diagrama de constelación de una señal QAM desfasado; para que los símbolos puedan ser receptados correctamente se emplearán filtros de coseno alzado con la finalidad de reducir la interferencia inter-símbolo (ISI).

En la Figura B.9 se muestra el diagrama de constelación de la señal QAM desplazada en frecuencia antes de la sincronización.

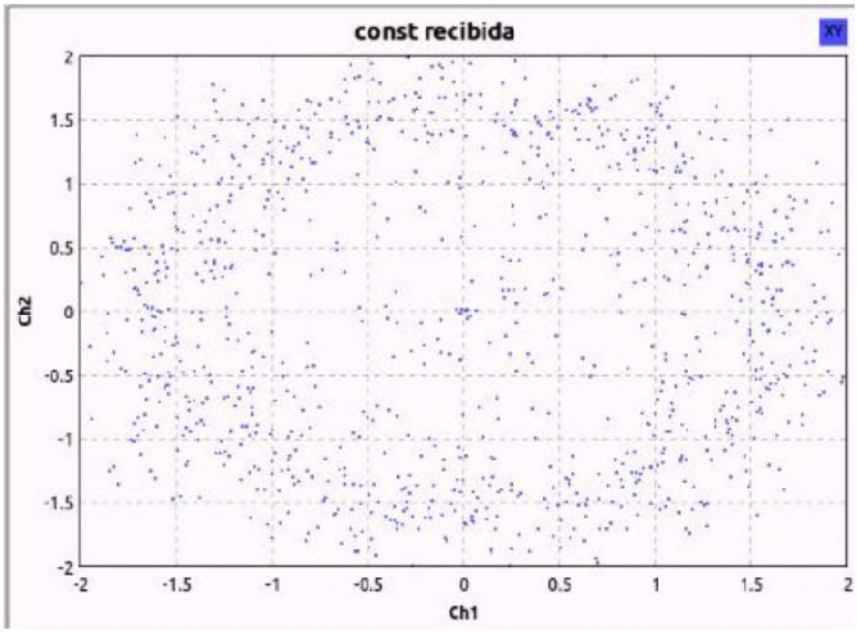

Figura B.9. Diagrama de constelación de la señal QAM desplazada en frecuencia antes de sincronización y filtrado.
Fuente: Imagen propia de los autores.

Al introducir un desplazamiento en frecuencia los símbolos se demodularán erróneamente (dependendiendo de la magnitud del desplazamiento). Para corregir este desplazamiento en frecuencia se introduce un bloque llamado **FLL Band-Edge**, el cual implementa un lazo de seguimiento de frecuencia.

En la Figura B.10 se presenta el resultado obtenido.

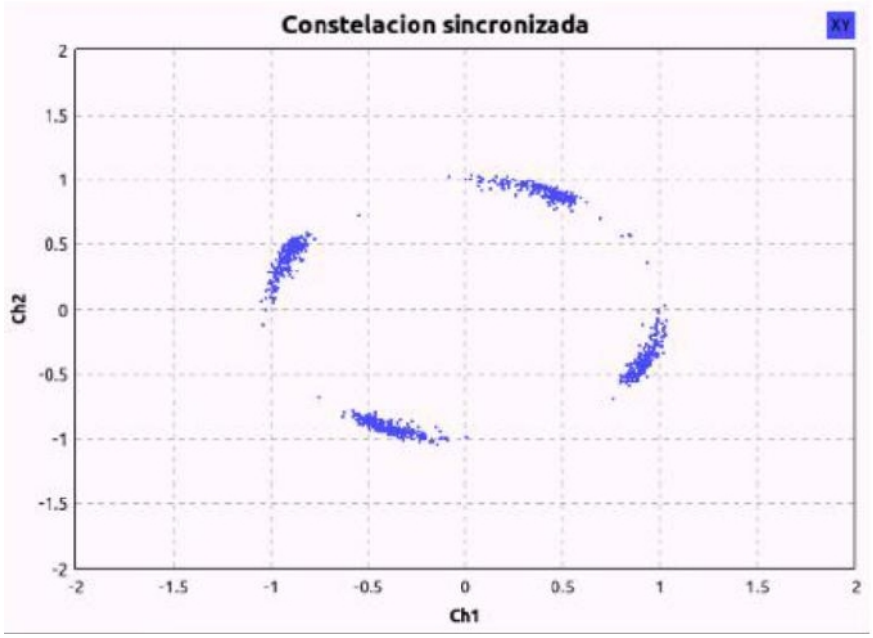

Figura B.10. Diagrama de la señal QAM luego del lazo de seguimiento de frecuencia.
Fuente: Imagen propia de los autores.

B.2.2.2 Sincronización de fase.

Los problemas con la sincronización de fase causan que los diagramas de constelación giren sobre sí mismos, impidiendo una correcta demodulación de la señal; para subsanar este problema se emplea un bloque denominado **Costas Loop** que implementa el algoritmo del mismo nombre.

En la Figura B.11 se muestra el resultado obtenido.

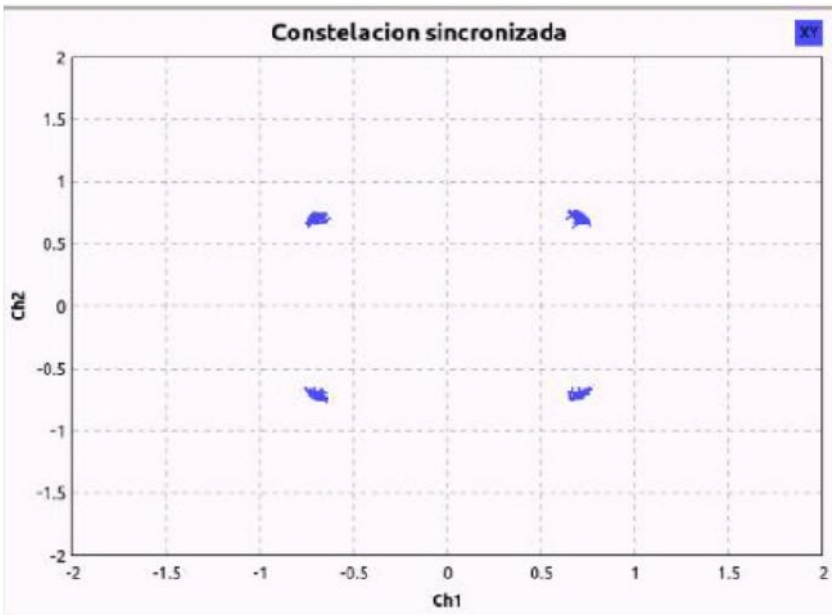

Figura B.11. Diagrama de constelación luego del bloque **Costas Loop**.
Fuente: Imagen propia de los autores.

B.2.3 Recuperación de señal de reloj.

Para solucionar este problema, se utiliza el bloque **Poliphase Clock Sync**. Este bloque utiliza un algoritmo que implementa un lazo de control para encontrar el tiempo de muestreo exacto y así arreglar las diferencias entre el transmisor y el receptor.

B.3 Multitrayecto

Como se puede observar en la Figura B.8 el multitrayecto causa que al llegar varios rayos al receptor, afecte directamente a la amplitud y a la fase (a no ser que los rayos estén en fase y/o contrafase) de la señal recibida; por este motivo, se puede apreciar como la constelación aparece rotada y en este caso con una amplitud superior.

Para corregir este problema se emplean los bloques: **Costas Loop** (mencionado anteriormente) y **CMA Equalizer**. Los resultados se muestran en la Figura B.12.

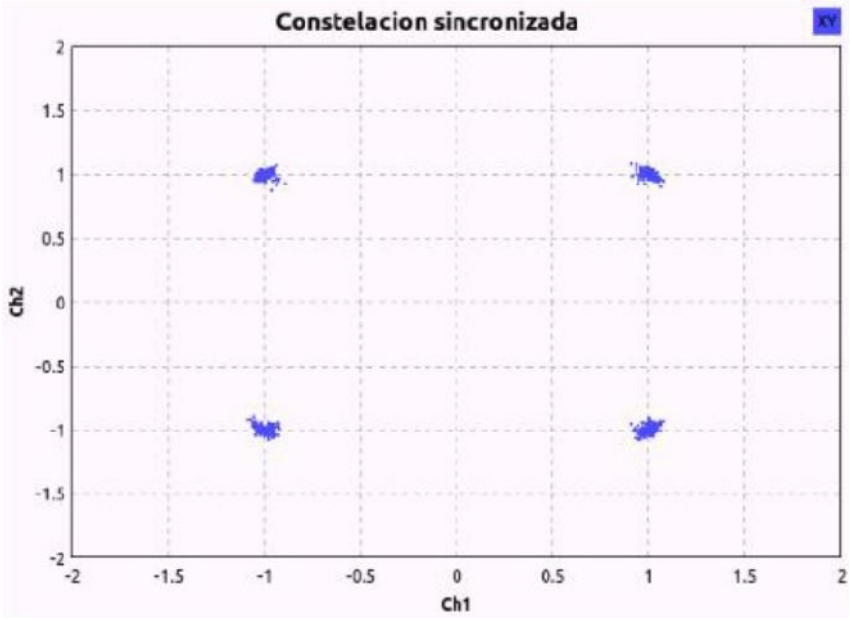

Figura B.12. Diagrama de constelación sincronizado y ecualizado.
Fuente: Imagen propia de los autores.

ANEXO C

CARGA DEL FIRMWARE

Para poder utilizar los USRPs N210 es necesario cargar el *firmware* respectivo en la tarjeta principal del dispositivo, para ello se emplea una herramienta gráfica que viene incluida con los *drivers* UHD. En la Figura C.1 se aprecia la ruta en el computador que se debe seguir para poder acceder al entorno gráfico.

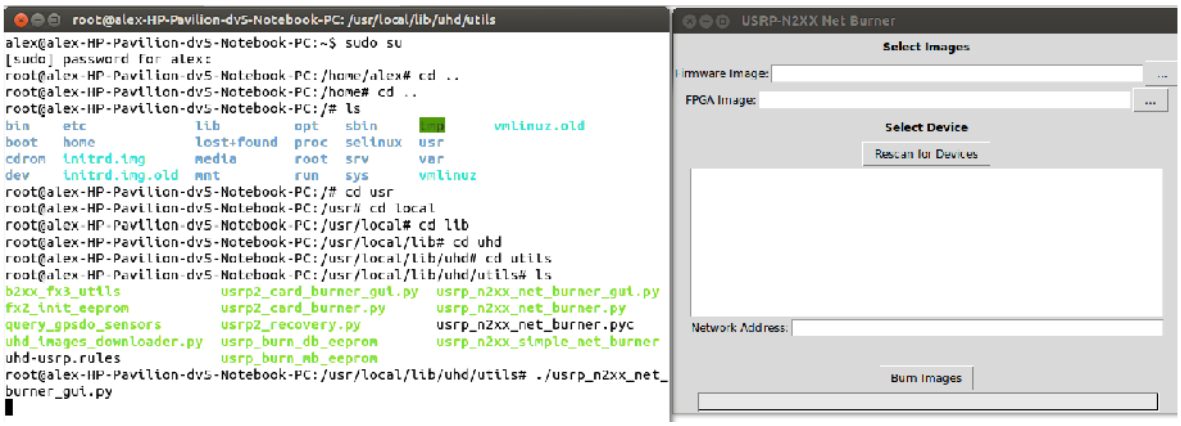

Figura C.1. Ruta para cargar imagen del USRP.
Fuente: Imagen propia de los autores.

Luego, se procede a cargar las imágenes tanto del *firmware* como de la FPGA, seleccionando los archivos en el computador e ingresando la ruta respectiva como se aprecia en la Figura C.2.

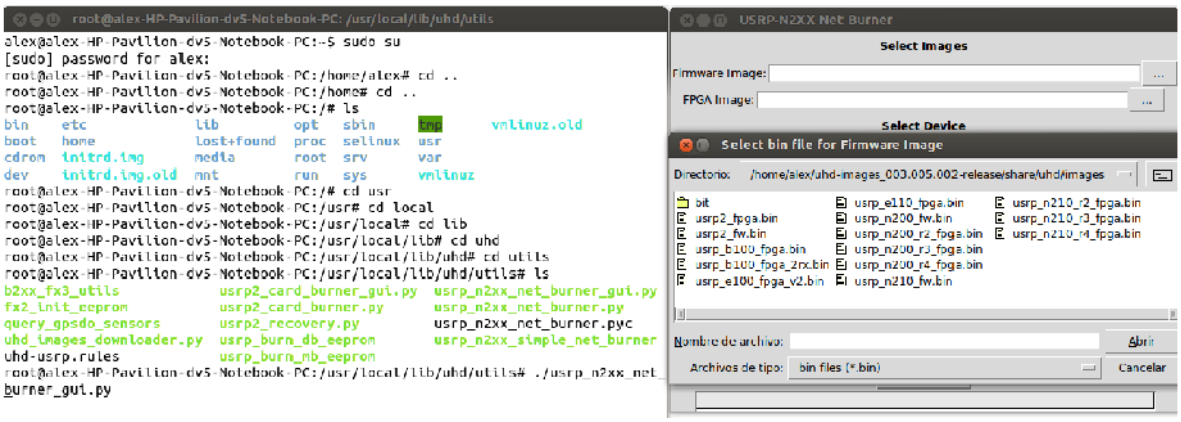

Figura C.2. Ruta para cargar imagen del *firmware* del USRP.
Fuente: Imagen propia de los autores.

A continuación se especifica la dirección IP del dispositivo USRP N210 y se procede a grabar las imágenes; en caso de haber alguna anterior, esta se sobrescribe, como se observa en la Figura C.3.

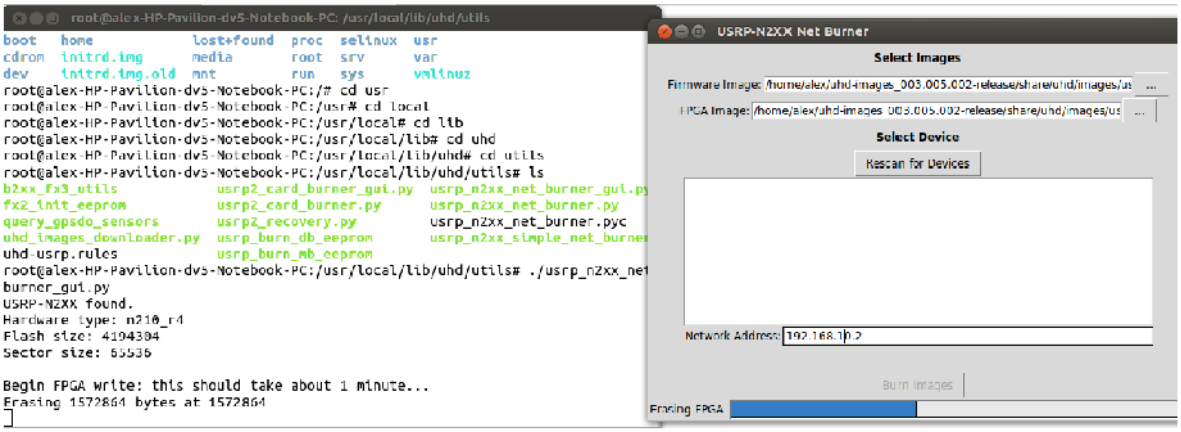

Figura C.3. Especificación de dirección IP del dispositivo.
Fuente: Imagen propia de los autores.

Para terminar el proceso el programa pedirá el reinicio del dispositivo como se aprecia en la Figura C.4.

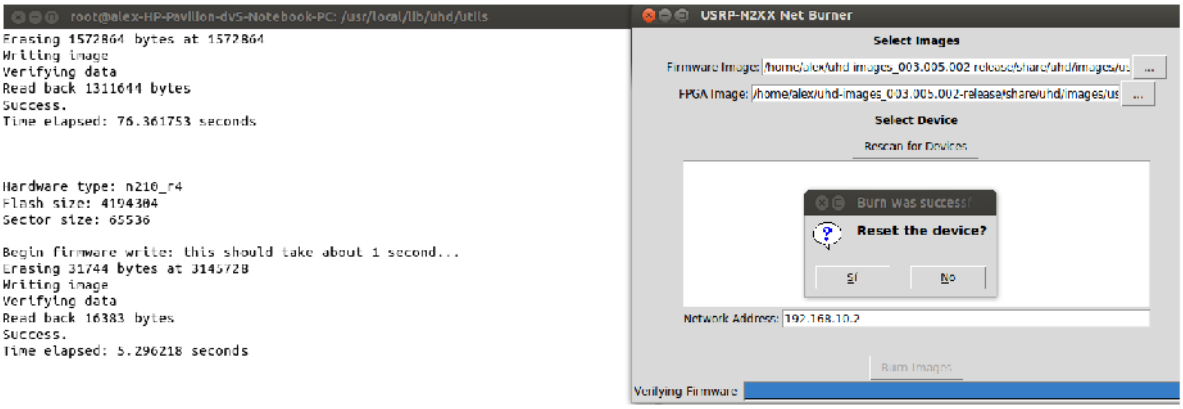

Figura C.4. Reinicio del dispositivo.
Fuente: Imagen propia de los autores.

ANEXO D

PRÁCTICAS DE LABORATORIO EMPLEANDO TECNOLOGÍA SDR

D.1 Práctica # 1

D.1.1 Analizador de espectros.

En esta primera práctica se implementará un analizador de espectros. La elaboración de este módulo es muy sencilla, solo se debe añadir el esquemático de la fuente USRP (de donde se reciben las señales) y conectarla al sumidero gráfico FFT con la configuración que se explicará a continuación.

Se debe especificar el rango de frecuencias que nos interesa analizar, siempre y cuando estas sean soportadas por el dispositivo USRP; en este caso nos interesa analizar el dial de frecuencias correspondientes a la radio FM. El esquemático implementado aparece en la Figura D.1.

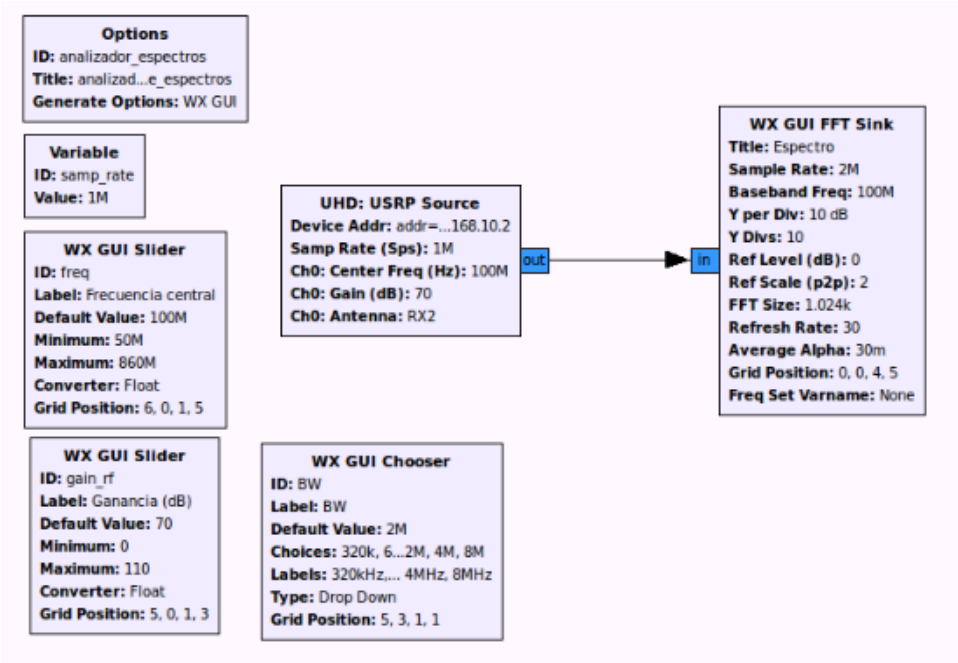

Figura D.1. Diagrama de bloques del analizador de espectros en GNU Radio *Companion.*
Fuente: Imagen propia de los autores.

A continuación se irá explicando paso a paso la configuración de cada bloque que conforma el analizador de espectros:

1. Definición de la fuente: en este caso, la señal será la que proporcione el USRP a través de la interfaz Gigabit Ethernet. La configuración de este bloque se muestra en la Figura D.2.

   Como se puede observar en la Figura D.2 la fuente es de tipo complejo; algo muy importante es especificar la dirección IP del dispositivo que por defecto es: **192.168.10.2**, los valores del reloj interno del dispositivo se los deja por defecto, en

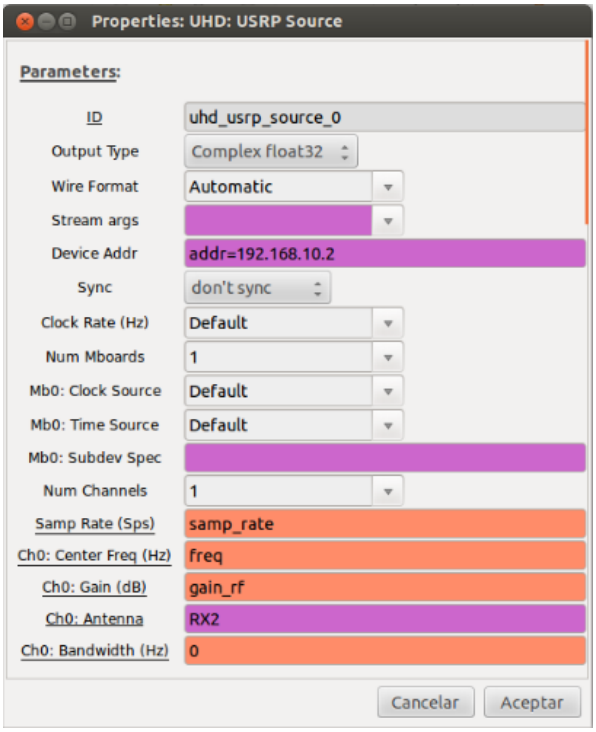

Figura D.2. Configuración del bloque **USRP Source**.
Fuente: Imagen propia de los autores.

este bloque también se debe especificar la tasa de muestreo *samples per second*, la frecuencia de recepción, ganancia del dispositivo, tipo de antena y ancho de banda; todos estos parámetros se los analizará posteriormente.

2. Selección del sumidero: el sumidero del flujo de señal será un sumidero gráfico de tipo FFT. Su configuración se muestra en la Figura D.3.

   Este sumidero es de tipo complejo. Obviamente, los extremos de una conexión tienen que ser del mismo tipo. Se especifica el título del bloque en la casilla *Title* y se asigna a *Sample rate* el identificador de la variable *BW* que se creará posteriormente. Para la frecuencia que se desee analizar, se creará el identificador *freq* (correspondiente al bloque **Variable Slider** en la interfaz gráfica). El resto de parámetros están asociados a la visualización: escala, número de muestras de la FFT (se ha seleccionado 1024 muestras para obtener una buena resolución del espectro), tasa de refresco, etc.

3. Inserción de variables y opciones: en este paso se explicará el proceso para crear variables que posteriormente puedan ser modificadas desde la interfaz gráfica al ejecutar el programa sin tener que parar su ejecución para modificar estos valores.

   Comenzamos con las variables *slider* mostradas en las Figuras D.4 y D.5.

Figura D.3. Propiedades del sumidero **FFT**.
Fuente: Imagen propia de los autores.

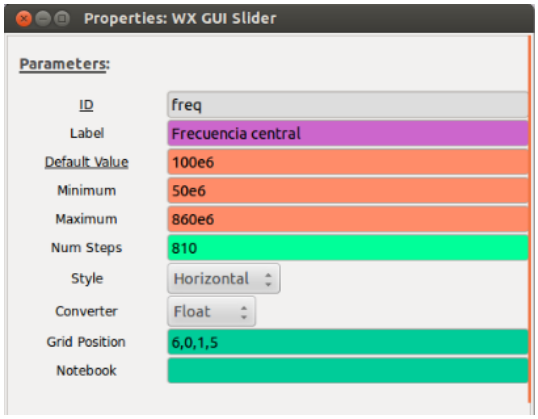

Figura D.4. Propiedades del sumidero **Slider**.
Fuente: Imagen propia de los autores.

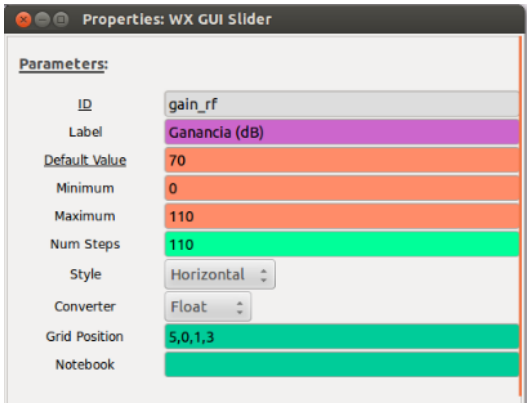

Figura D.5. Propiedades del sumidero **Slider**.
Fuente: Imagen propia de los autores.

Primero se debe indicar el nombre de la variable que se asocia a ellas para identificarlas en el resto del esquemático. Después se selecciona la etiqueta que se mostrará en la interfaz gráfica, el rango de valores posibles y el estilo.

Debemos establecer estas variables; tanto de frecuencia como de ganancia, en valores que sean soportados por el dispositivo y no exceder estos límites para no causar un mal funcionamiento del equipo o comportamientos inesperados del mismo.

La última variable definida en el esquemático es del tipo **Variable Chooser** para ajustar el ancho de banda. Con esta variable se pueden elegir valores discretos previamente definidos. Sus propiedades aparecen en la Figura D.6.

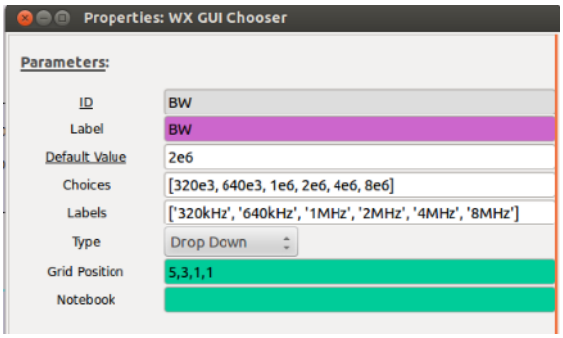

Figura D.6. Selección del ancho de banda.
Fuente: Imagen propia de los autores.

El único bloque que falta por comentar es el de opciones. En éste, se selecciona el nombre que tendrá el fichero .py que se genera una vez se compile el esquemático, el título de la interfaz gráfica, el autor y algún otro detalle de visualización como una descripción de la práctica o el tamaño de la ventana de visualización de la interfaz.

Para ajustar todos los elementos creados en la interfaz gráfica en la posición y tamaño que se desee, se utiliza el parámetro *Grid Position*, que se puede apreciar en todos los bloques (menos el de opciones, pues no aparece en la interfaz). La nomenclatura es: fila, columna, alto (en número de filas), ancho (en número de columnas).

Finalmente, tras haber finalizado el diseño esquemático procedemos a compilar y ejecutar; generándose así el archivo Python correspondiente; el resultado se muestra en la Figura D.7 en el que podemos observar el espectro correspondiente a la emisora Luz y Vida de la ciudad de Loja con frecuencia central de 88.1MHz.

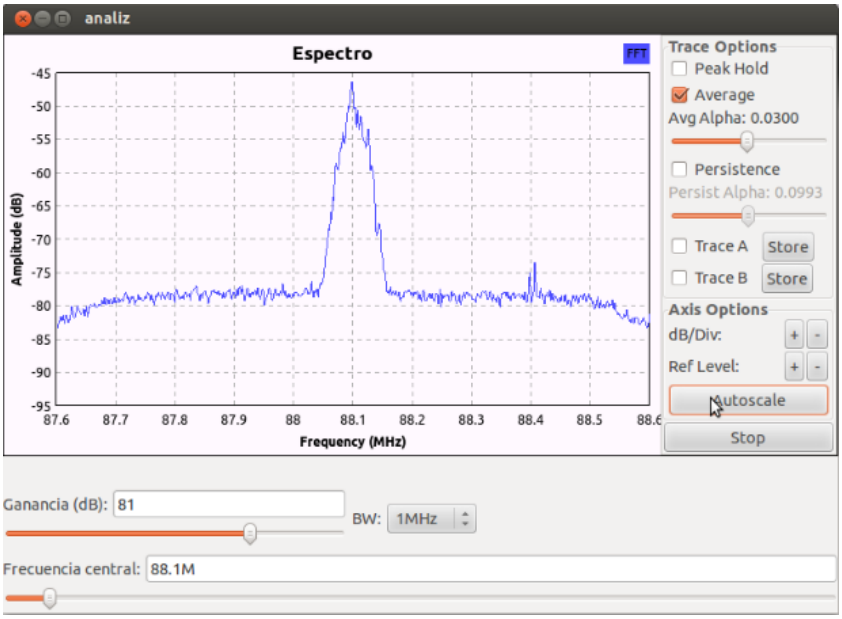

Figura D.7. Analizador de espectros en banda de FM.
Fuente: Imagen propia de los autores.

D.2 Práctica # 2

D.2.1 Osciloscopio.

Al igual que se ha implementado el analizador de espectros, se puede realizar el diseño de un osciloscopio simplemente sustituyendo el bloque **FFT Sink** del esquemático de la práctica #1 por el bloque **Scope Sink**. En las siguientes prácticas utilizaremos este bloque para visualizar la señal en el dominio temporal, o incluso obtener la constelación activando el modo XY útil para los sistemas de comunicaciones digitales.

El esquemático del osciloscopio se muestra en la Figura D.8.

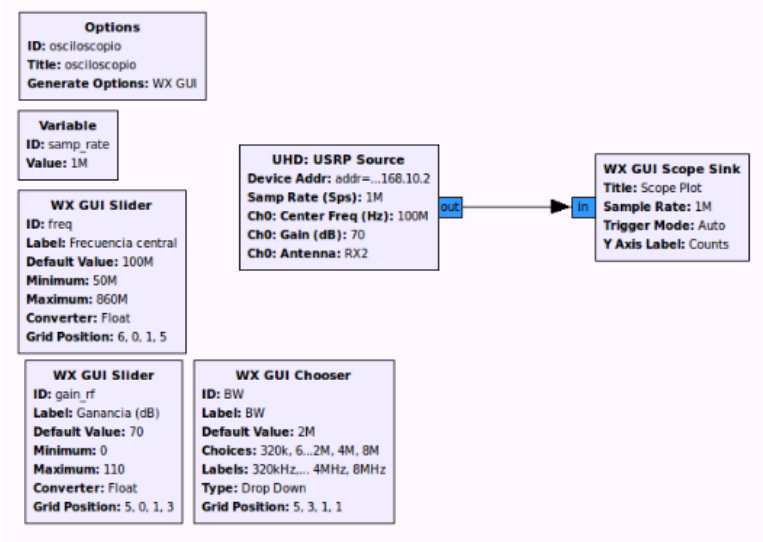

Figura D.8. Esquemático del osciloscopio.
Fuente: Imagen propia de los autores.

En la Figura D.9 se muestra la configuración de un **Slider** para la frecuencia de recepción.

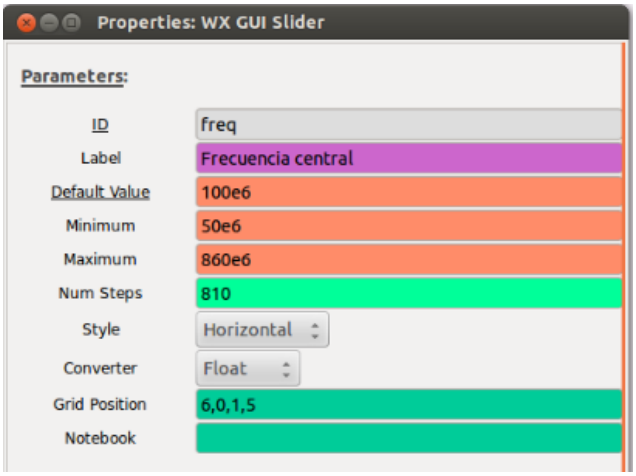

Figura D.9. Frecuencia de recepción del osciloscopio.
Fuente: Imagen propia de los autores.

En la Figura D.10 se configura un **Slider** para la ganancia variable de recepción.

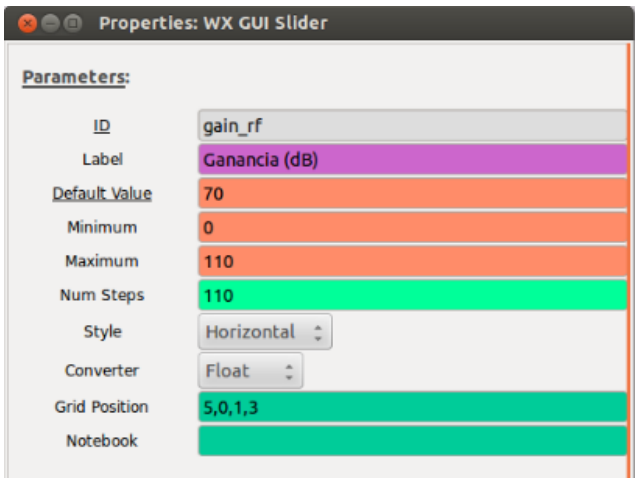

Figura D.10. Ganancia variable de recepción.
Fuente: Imagen propia de los autores.

En la Figura D.11 se observa la configuración del bloque **Chooser** osciloscopio.

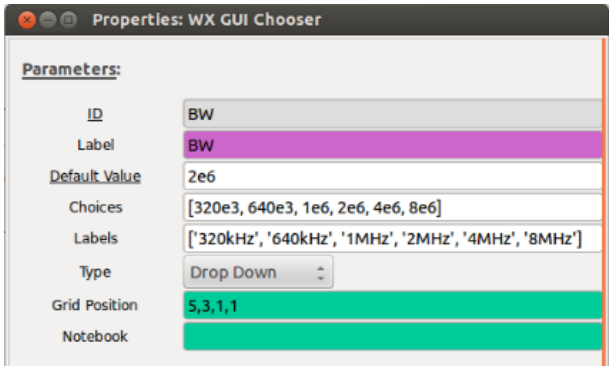

Figura D.11. Configuración del bloque **Chooser**.
Fuente: Imagen propia de los autores.

En la Figura D.12 se observa la configuración del bloque **USRP Source**.

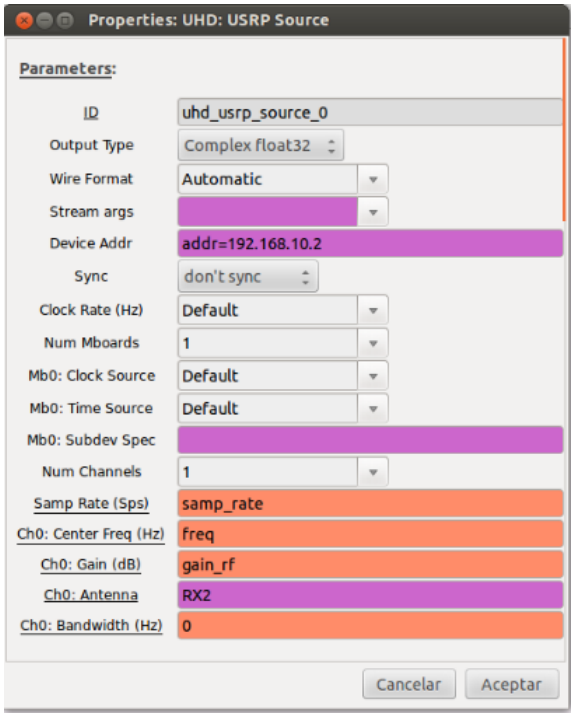

Figura D.12. Configuración del bloque **USRP Source**.
Fuente: Imagen propia de los autores.

En la Figura D.13 se observa la configuración del bloque **Scope Sink**.

Figura D.13. Configuración del bloque **Scope Sink**.
Fuente: Imagen propia de los autores.

Tras haber finalizado el diseño esquemático procedemos a compilar y ejecutar; generándose el archivo Python. El resultado se muestra en la Figura D.14.

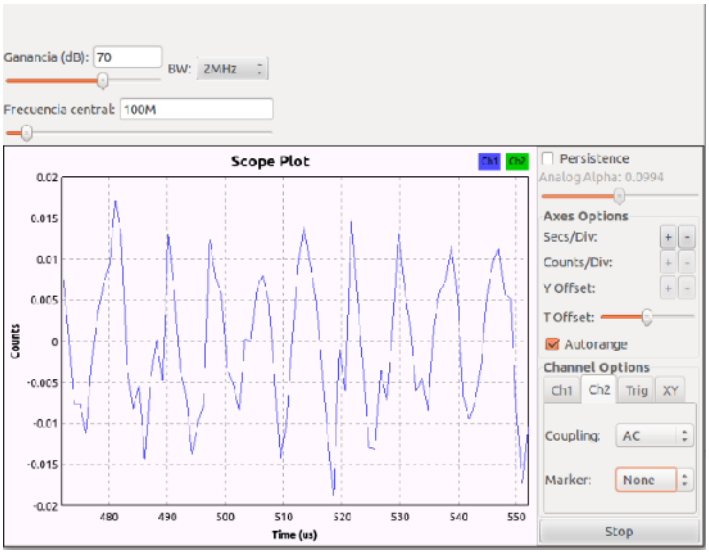

Figura D.14. Osciloscopio en Banda FM.
Fuente: Imagen propia de los autores.

D.3 Práctica # 3

D.3.1 Sistema de recepción FM.

Una de las formas sencillas de realizar un receptor en frecuencia modulada es a través del USRP, ya que por medio de una programación por bloques se lo puede realizar y evitar hacer un circuito físico que nos demanda una cantidad de tiempo considerable, entonces teniendo a disposición el USRP se procedió a implementarlo, tal cual se muestra en la Figura D.15.

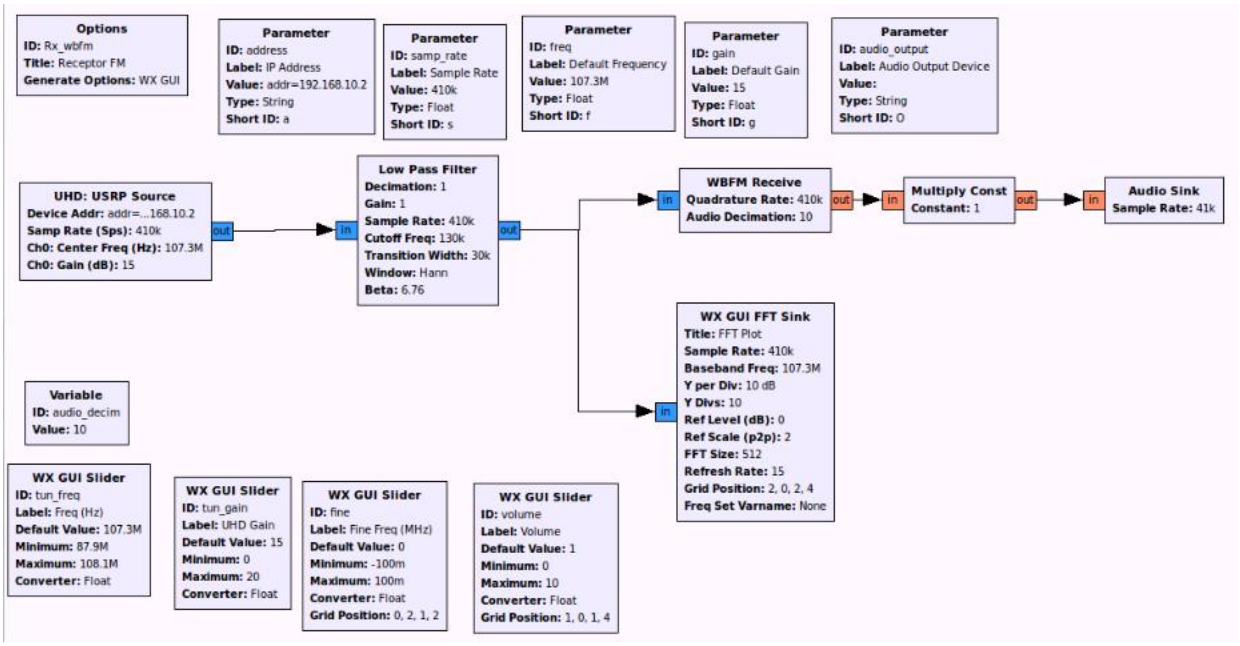

Figura D.15. Diagrama de bloques del receptor FM en GNU Radio *Companion*.
Fuente: Imagen propia de los autores.

A continuación se presenta paso a paso la configuración de los bloques que conforman el receptor FM:

1. Definición de la fuente USRP: en las propiedades del UHD, es donde se coloca la fuente de información que va a venir desde el USRP y se colocan diferentes parámetros, como la dirección IP del dispositivo a través del parámetro *address* que ya describimos anteriormente, así mismo la tasa de muestreo que se va a emplear *samp rate*, la frecuencia central a la que va a estar funcionando el dispositivo.

   Se empleará para seleccionar la frecuencia un *tun freq* que es un **Slider** que se colocó para cambiar con facilidad las diversas emisoras en el dial de la radio más un *fine* que es un sintonizador de frecuencia fino y también la ganancia entre los principales elementos que se ocuparon para la configuración de la fuente USRP como se muestra en la Figura D.16.

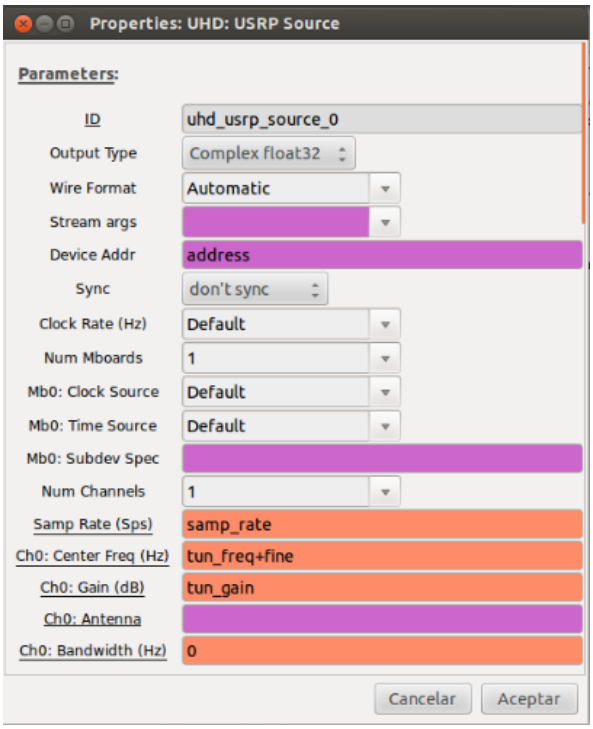

Figura D.16. **USRP Source** del receptor FM.
Fuente: Imagen propia de los autores.

2. Adición de un filtro pasabajas: para poder recibir la señal FM deseada y atenuar el ruido, se coloca un filtro pasabajas donde se coloca la tasa de muestreo a la que está funcionado el equipo y la frecuencia de corte para que sólo pase la señal deseada y el resto se atenúe como se muestra en la Figura D.17.
3. Demodulación de la señal: luego de recibir la señal en el USRP y pasar por un filtro pasabajas como se dijo anteriormente, es necesario un bloque para demodular la

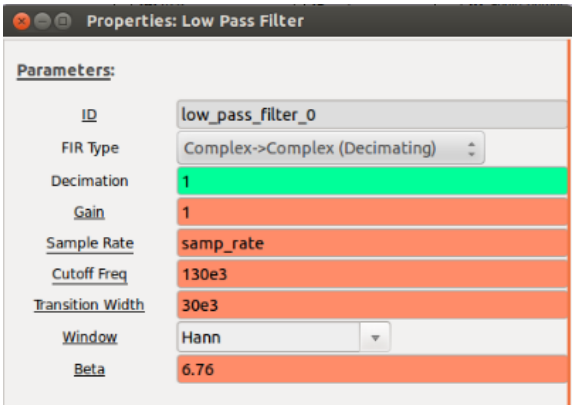

Figura D.17. Filtro Pasabajas para recepción FM.
Fuente: Imagen propia de los autores.

señal FM, así que ocupamos el demodulador **WBFM** (FM de banda ancha);donde se configuran dos parámetros como son la tasa de cuadratura que va a ser la misma que la tasa de muestreo y el diezmado de audio que se va a tener como se muestra en la Figura D.18.

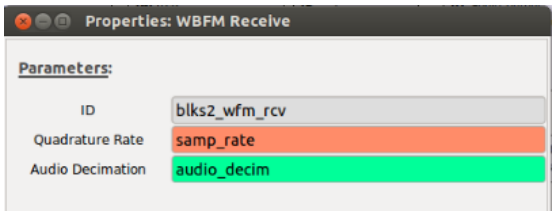

Figura D.18. Propiedades del bloque **WBFM Receive**.
Fuente: Imagen propia de los autores.

Se colocó un bloque multiplicador para el control de volumen y al final el bloque **Audio Sink** que tendrá una frecuencia de muestreo de 41KHz que servirá para reproducir la señal en los altavoces del computador.

4. Gráfica de la señal recibida: para poder graficar el espectro de la señal FM que estamos recibiendo hacemos uso del bloque **WX GUI FFT Sink** que es una interfaz gráfica para visualizar el espectro mencionado anteriormente y cuya configuración se muestra en la Figura D.19.

   En este bloque se grafica el espectro de la señal de FM que se está recibiendo, en donde la tasa de muestreo es la misma que se ha ocupado en toda la práctica y para que se grafique el espectro según se vaya cambiando la frecuencia con el **Slider** y sintonizador de frecuencia fina se colocó en el parámetro de frecuencia banda-base *tune freq+fine* como ya se lo explicó en la sección de la fuente del USRP.

Figura D.19. Bloque **FFT Sink** del receptor FM.
Fuente: Imagen propia de los autores.

Una vez que todos estos bloques se han configurado correctamente tenemos como resultado lo presentado en la Figura D.20 donde se puede apreciar el espectro de la señal FM que se está receptando, en la frecuencia 99.3MHz.

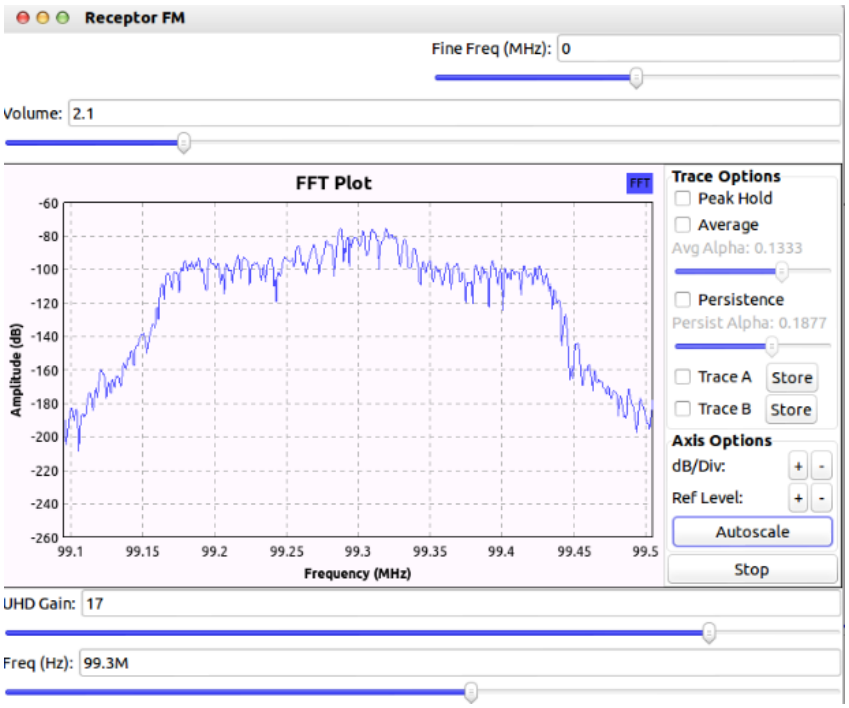

Figura D.20. Espectro del receptor FM realizado en GNU Radio.
Fuente: Imagen propia de los autores.

D.4 Práctica # 4

D.4.1 Sistema de transmisión FM.

El sistema FM es muy utilizado actualmente debido a sus características y numerosas aplicaciones, en la Figura D.21 se presenta la configuración de un transmisor FM empleando un archivo de audio en formato .wav desde nuestro computador como señal moduladora.

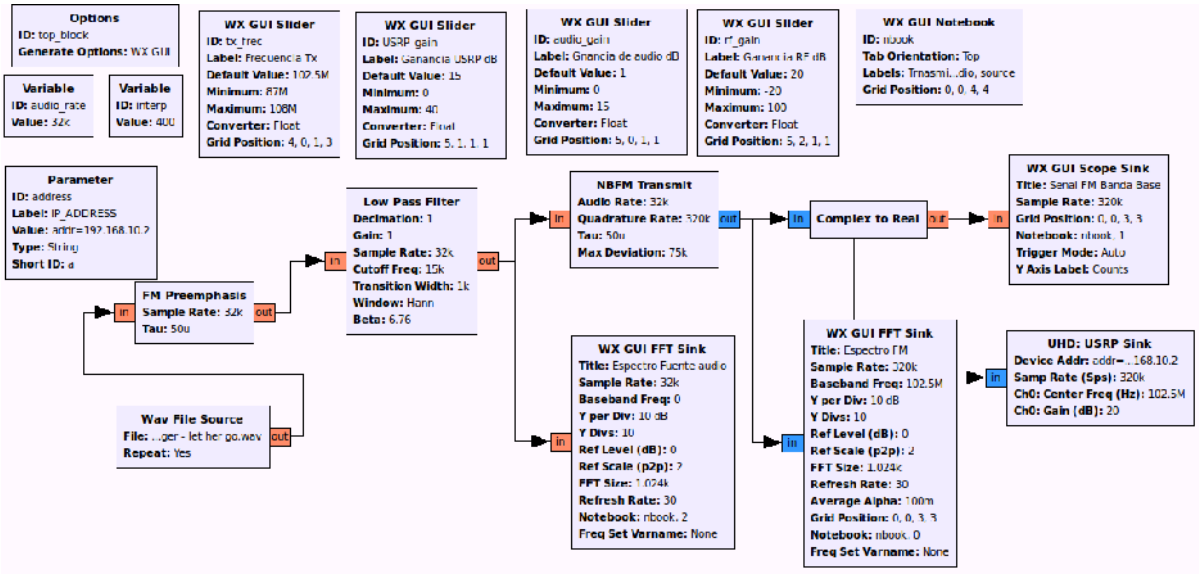

Figura D.21. Esquema del transmisor FM diseñado en GNU Radio *Companion*.
Fuente: Imagen propia de los autores.

Para el transmisor se puede cargar el archivo de sonido de diferentes formas: a través de la opción **Wav File Source**, en el que se cargan los archivos .wav; pero estos deben estar a la misma tasa de muestreo que se emplee para el sistema, en este caso 32KHz, o también obtener las señales de voz desde el micrófono del computador mediante el bloque **Signal Source**.

Para el transmisor realizado se utilizó el bloque **Wav File Source** como fuente del archivo de audio y cuya configuración se muestra en la Figura D.22.

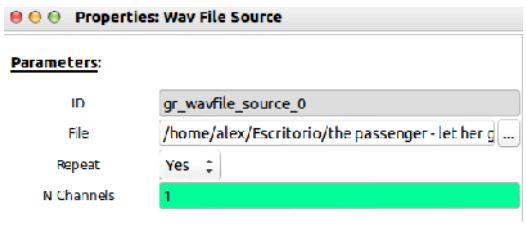

Figura D.22. Propiedades del bloque **Wav File Source**.
Fuente: Imagen propia de los autores.

En la opción *file* se carga el archivo de audio que se desea transmitir desde el computador y debe tener una tasa de muestreo de 32KHz si no dará un error de interpolación y no se

transmitirá; la opción *Repeat* sirve para que el archivo .wav que elegimos se repita continuamente o sólo se reproduzca una vez.

A continuación del archivo fuente se coloca un bloque **Preemphasis** que utiliza la misma tasa de muestreo que el archivo .wav y nos ayudará a disminuir el ruido de alta frecuencia que se genera durante la modulación en FM.

Con el archivo que se desea transmitir en el mismo tipo de dato que el filtro se procede a la configuración se sus parámetros como se indica en la Figura D.23.

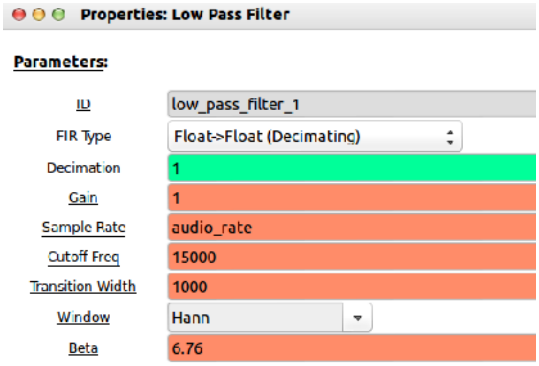

Figura D.23. Filtro pasabajas del transmisor FM.
Fuente: Imagen propia de los autores.

Se ha definido este filtro con una frecuencia de corte de 15KHz, la tasa de muestreo va a ser la misma que ocupamos durante la elaboración del esquemático (32KHz), el filtro tendrá un enventanado tipo Hann.

Luego que la señal ha pasado por una etapa de filtrado es necesario modularla para que pueda ser transmitida en el dial de FM, para esto utilizamos el bloque **NBFM Transmit**, que es de banda estrecha y configuramos algunos parámetros como se muestra en la Figura D.24.

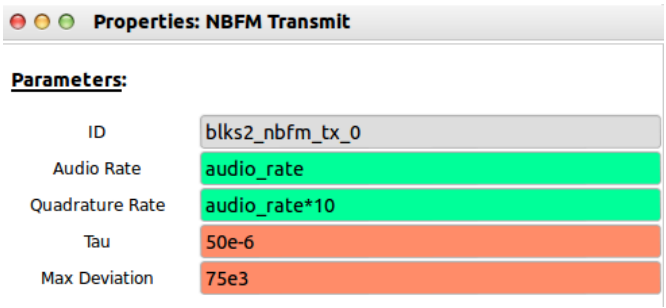

Figura D.24. Propiedades del bloque **NBFM Transmit**.
Fuente: Imagen propia de los autores.

La tasa de audio es la tasa de muestreo multiplicada por un valor de 10; por ende, la tasa de cuadratura será 10 veces mayor ya que ésta debe ser múltiplo de acuerdo a la tasa de

muestreo que se ocupó, la constante de tiempo *tau* es de $50\ \mu s$ y la desviación máxima de frecuencia para FM es de 75KHz de acuerdo con el estándar comercial.

Por último, se añade la fuente de transmisión del USRP, donde se colocan todos los parámetros para transmitir de manera correcta la señal de audio como se muestra en la Figura D.25.

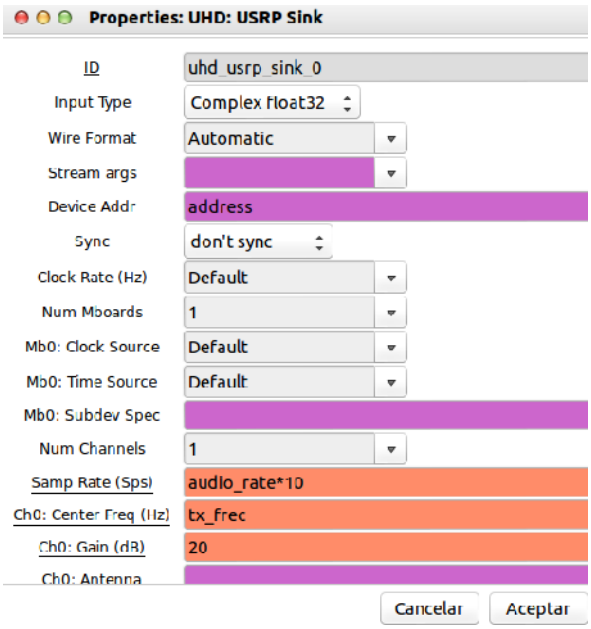

Figura D.25. Sumidero USRP del transmisor FM.
Fuente: Imagen propia de los autores.

En esta sección del transmisor se especifica la dirección IP del equipo que por defecto es **192.168.10.2**, la frecuencia de muestreo que se colocó multiplicada por un factor de diez, la frecuencia central a la que va a estar transmitiendo el equipo con su respectiva ganancia. Con toda esta configuración se transmitió una señal de audio para la frecuencia de 102.5MHz, que puede modificarse por medio de un slider que se configuró para todo el dial que conforma FM.

Para visualizar el espectro FM que se está transmitiendo se colocó un sumidero **FFT Sink** cuya configuración se observa en la Figura D.26. Posee una interfaz gráfica y permite observar en tiempo real el espectro que se está transmitiendo.

Sin embargo, se debe elegir una frecuencia libre para poder transmitir, ya que si elegimos una frecuencia ocupada, interferiremos con esa transmisión o nuestra señal se atenuará ya que es de baja potencia, en relación a las transmisiones de radio FM locales.

El resultado del transmisor FM se muestra en la Figura D.27.

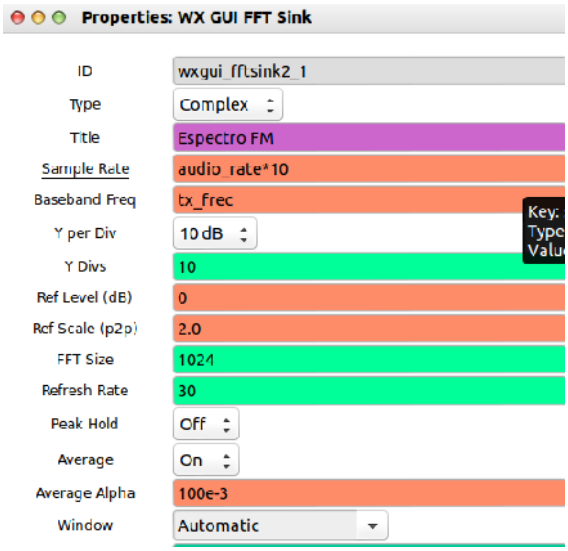

Figura D.26. Bloque FFT Sink del transmisor FM.
Fuente: Imagen propia de los autores.

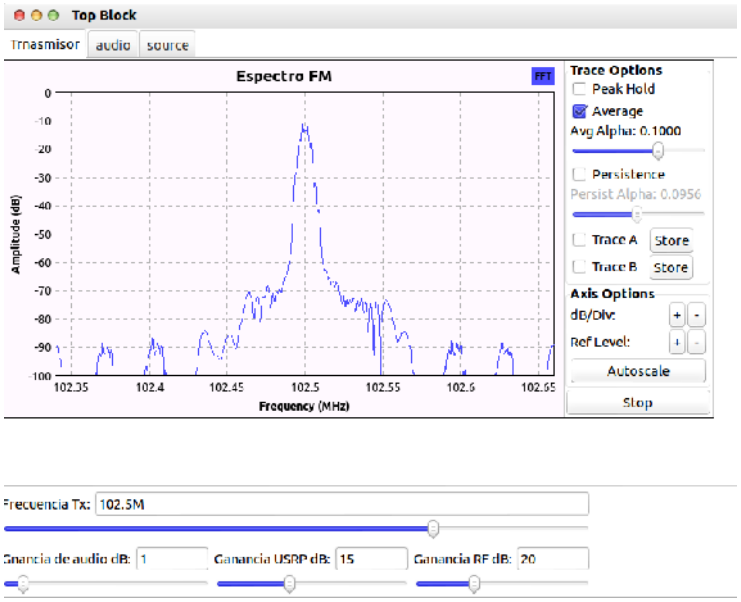

Figura D.27. Espectro FM transmitido.
Fuente: Imagen propia de los autores.

A la salida del modulador **NBFM** se colocó un conversor de complejo a real para observar la señal de audio en banda base, en donde colocamos la tasa de audio mencionada anteriormente.

Como se puede observar en la Figura 2.27 el archivo de audio se está modulando y transmitiendo en la frecuencia 102.5MHz en FM y el resto de armónicos se están atenuando por medio del filtro; el alcance que se logró con este transmisor fue alrededor de 50m, esto se podría mejorar con otro tipo de antenas que tenga una mayor ganancia o empleando amplificadores externos al USRP y con ello tener mayor alcance de transmisión.

D.5 Práctica # 5

D.5.1 Sistema de transmisión FM.

Este sistema de transmisión es similar al que se realizó anteriormente, pero para este caso se deben colocar fuentes adicionales para transmitir en diversas frecuencias (varias portadoras); el esquemático realizado se presenta en la Figura D.28.

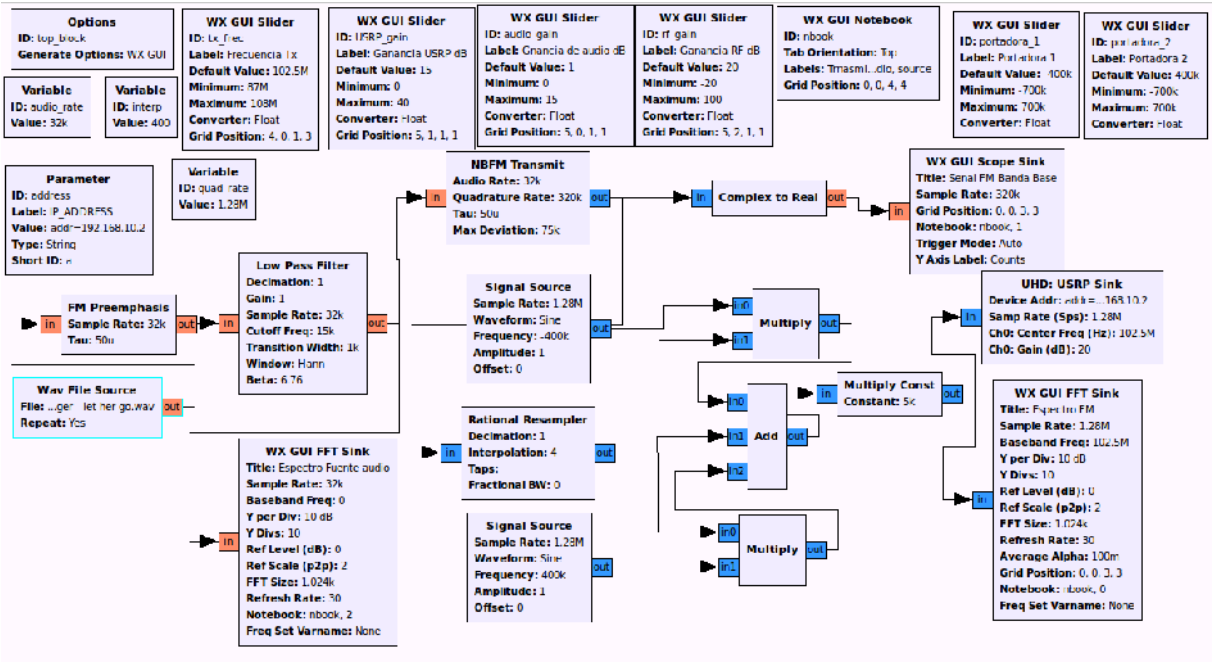

Figura D.28. Esquemático del transmisor FM con múltiples portadoras.
Fuente: Imagen propia de los autores.

A la práctica del sistema de transmisión FM simplemente se le colocó dos fuentes adicionales que serán de tipo coseno (portadoras) y se multiplicarán con la señal de audio .wav que se ingresó al principio de la práctica, estas frecuencias se ubicarán a 300KHz y -300KHz de la frecuencia central y por ende se estará transmitiendo en tres frecuencias el mismo sonido.

La configuración de estas fuentes se presenta en las Figuras D.29 y D.30.

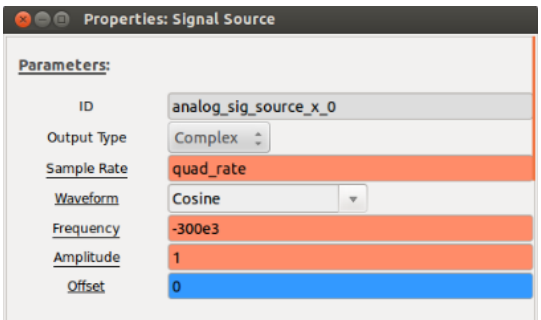

Figura D.29. Portadora adicional 1.
Fuente: Imagen propia de los autores.

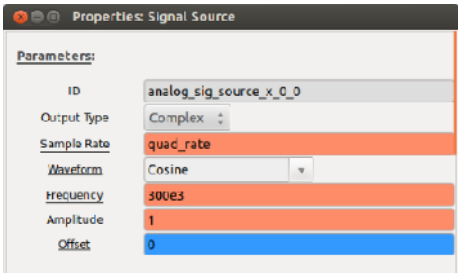

Figura D.30. Portadora adicional 2.
Fuente: Imagen propia de los autores.

La tasa de muestreo va a ser la misma que se puso en la tasa de cuadratura y las frecuencias para este caso están definidas en cada **Slider** de las diferentes portadoras que se deseen transmitir.

La señal que sale del transmisor tiene que pasar por un remuestreador (*resampler*), su configuración se muestra en la Figura D.31. Posteriormente esta señal tendrá que multiplicarse con las nuevas portadoras y sumarse con la señal FM original (señal modulada en la frecuencia central) y puedan transmitirse al mismo tiempo.

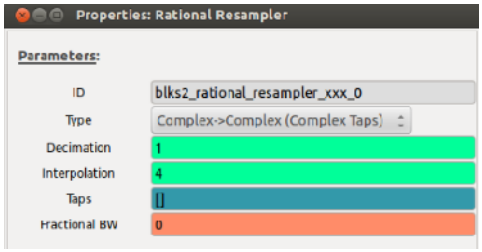

Figura D.31. Configuración del remuestreador.
Fuente: Imagen propia de los autores.

El sumidero va a ser la fuente que va a transmitir la señal hacia el USRP, su configuración se muestra en la Figura D.32.

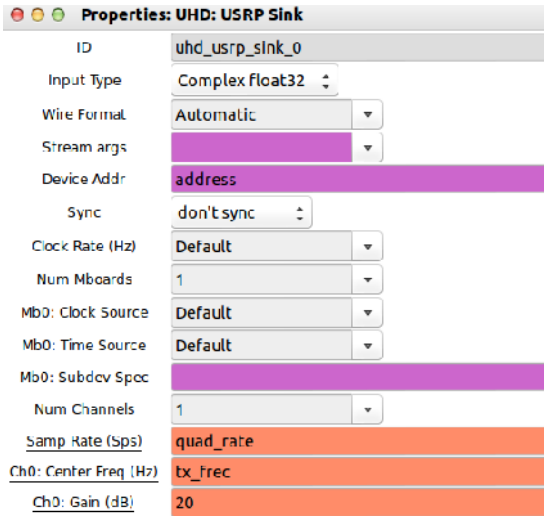

Figura D.32. Configuración del bloque USRP.
Fuente: Imagen propia de los autores.

Mediante el bloque **FFT Sink** se procede a graficar el espectro de las señales en FM transmitidas su configuración se presenta en la Figura D.33.

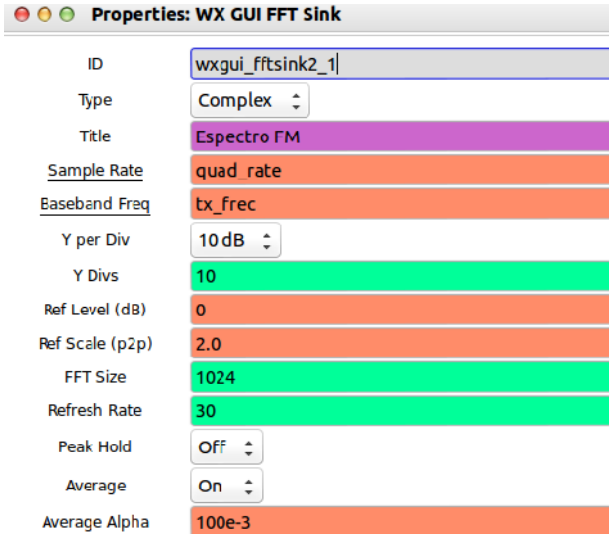

Figura D.33. Configuración del bloque **FFT Sink**.
Fuente: Imagen propia de los autores.

En la Figura D.34 se presenta el resultado obtenido, en la que se puede apreciar la transmisión en tres frecuencias distintas la frecuencia central a la que se está transmitiendo que es 102.5MHz, y las dos frecuencias adicionales se encuentran separadas a 400KHz de la frecuencia central, se las puede variar con los *sliders*. Así, tenemos las tres portadoras transmitiéndose simultáneamente.

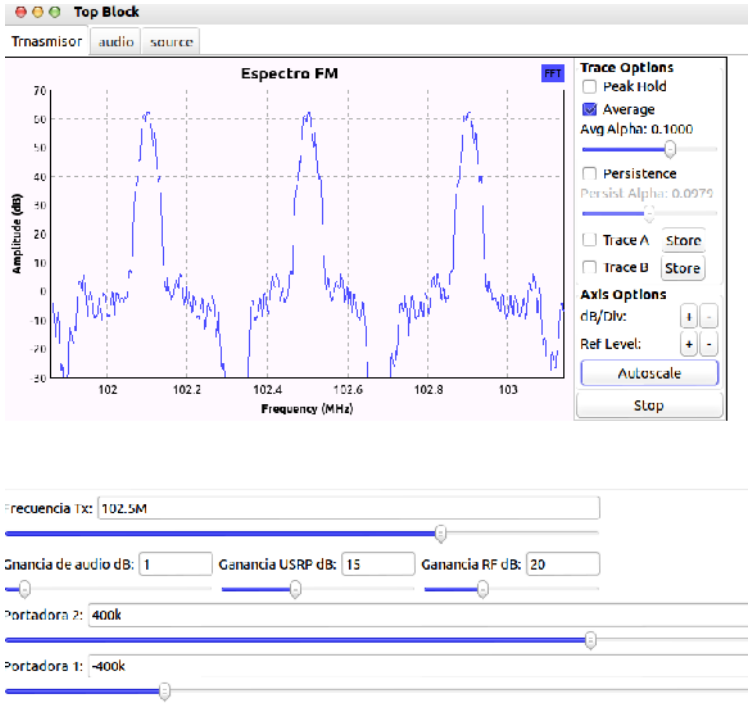

Figura D.34. Transmisor FM multicanal.
Fuente: Imagen propia de los autores.

D.6 Práctica # 6

D.6.1 Sistema de transmisión FM.

En la práctica anterior se ha transmitido una misma señal de audio en diferentes portadoras, ahora lo que se realiza es transmitir una señal de audio diferente en cada una de las portadoras, dando como resultado el esquemático mostrado en la Figura D.35.

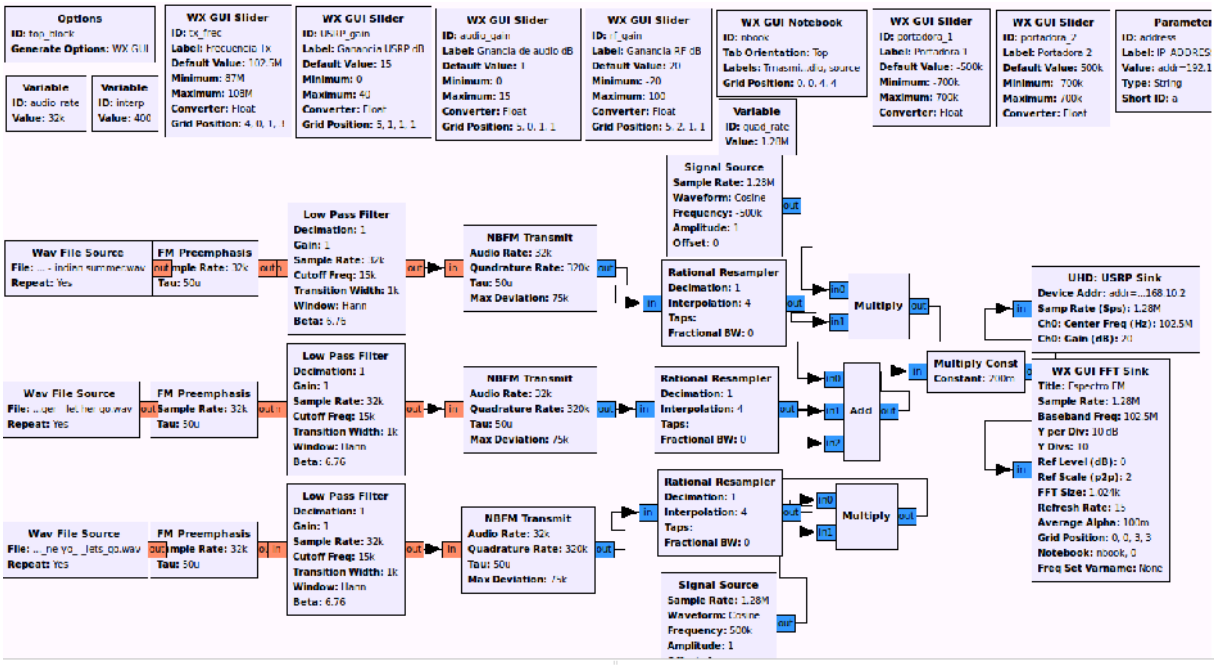

Figura D.35. Esquemático del transmisor FM con múltiples portadoras y múltiples fuentes de audio.
Fuente: Imagen propia de los autores.

Como se ve en el esquemático mostrado en la Figura D.35, a diferencia de la práctica anterior que sólo se tenía un bloque **Wav File Source** para la señal de audio, ahora tendremos tres ya se transmite en tres diferentes frecuencias del dial de FM.

Ya que la señal ha pasado por un filtro es necesario modularla para que pueda ser transmitida en el dial de FM, para esto utilizamos el bloque **NBFM**, que es de banda estrecha y configuramos algunos parámetros como se muestra en la Figura D.36.

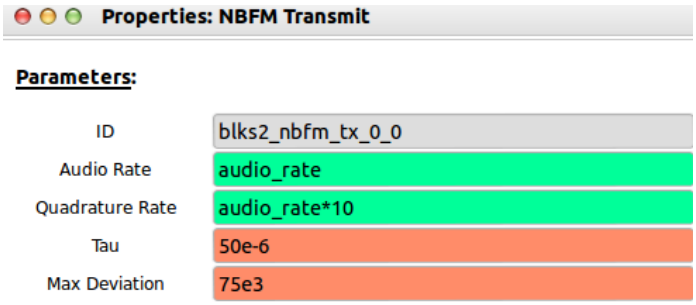

Figura D.36. Transmisor FM multicanal.
Fuente: Imagen propia de los autores.

Todas las opciones mencionadas anteriormente hay que realizarlas las veces que sea necesario, ya que estas señales posteriormente se deben multiplicar por una señal cosenoidal para tener una adecuada modulación. A continuación se suman estas señales portadoras y luego se multiplican por una constante, que regula el volumen con que se va a transmitir y para esta práctica el valor óptimo fue de 0,2 como se puede apreciar en la Figura D.37.

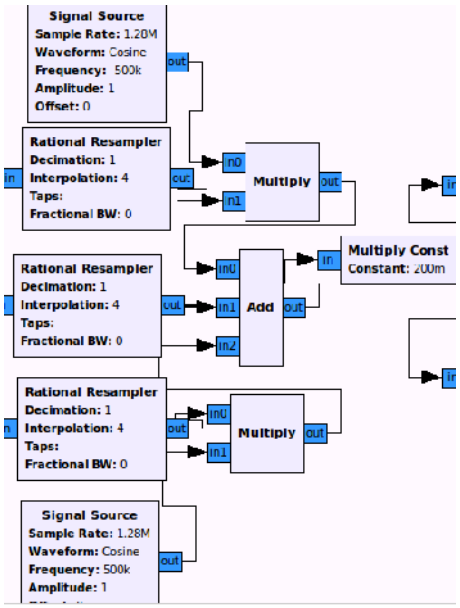

Figura D.37. Multiplicación de fuentes.
Fuente: Imagen propia de los autores.

Al igual que en la práctica anterior se pone al final un **FFT Sink** y **USRP Sink** para poder graficar el espectro de audio y poner las propiedades de la fuente del transmisor respectivamente. Como resultado se obtiene la Figura D.38 en la que se observa las tres portadoras con una separación de 500KHz y siendo cada una de ellas moduladas con un audio diferente.

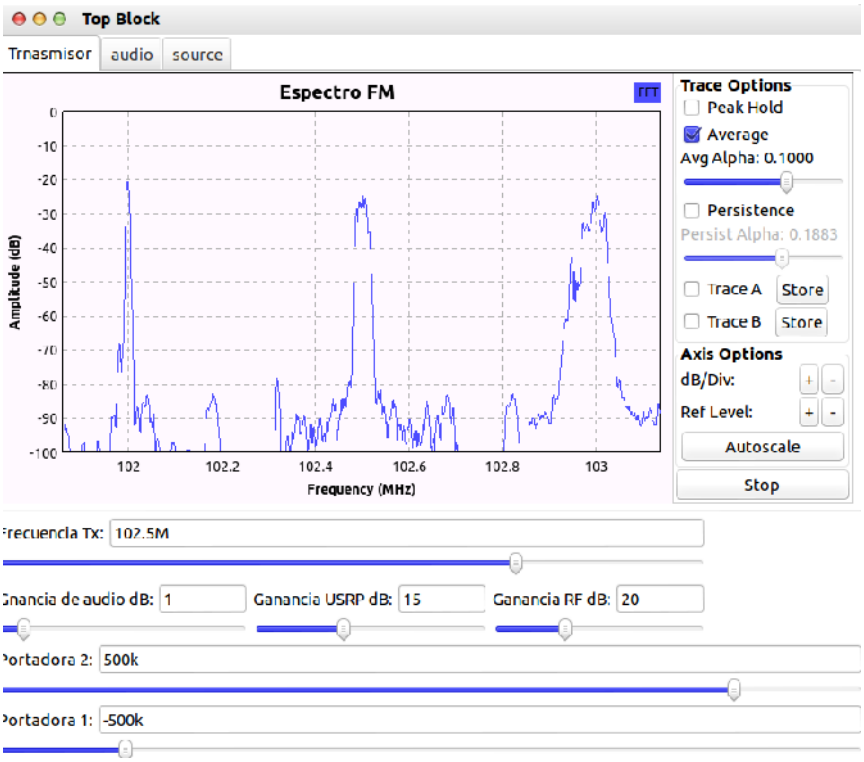

Figura D.38. Espectro de las diferentes portadoras moduladas con audios diferentes.
Fuente: Imagen propia de los autores.

D.7 Práctica # 7

D.7.1 Simulación del cálculo de la tasa de bits erróneos transmitidos (BER).

En esta práctica se implementará un esquema para obtener la tasa de bits errados; se utilizarán dos bloques **File Source**; en los cuales, se cargará una imagen en cada uno de ellos. En el primer bloque se pondrá la imagen y en el segundo la misma pero distorsionada. El bloque **Throttle** se usa para limitar el uso de la CPU, debido a que no se emplean plataformas *hardware* en este caso. El bloque **Error Rate** calcula el BER y lo muestra en un sumidero numérico.

El esquemático para el cálculo del BER se muestra en la Figura D.39.

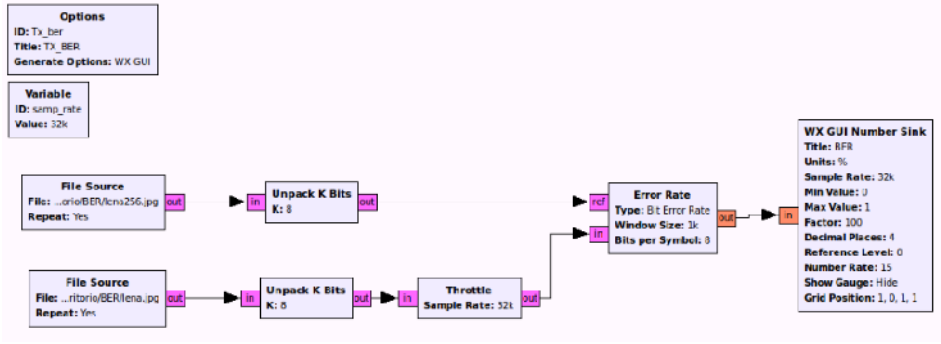

Figura D.39. Esquemático para el cálculo del BER.
Fuente: Imagen propia de los autores.

En la Figura D.40 se muestran las imágenes cargadas en los bloques **File Source**.

Figura D.40. Imágenes cargadas en el transmisor (a) y receptor (b).
Fuente: Imagen propia de los autores.

La configuración de los bloques **File Source** se muestra en la Figura D.41.

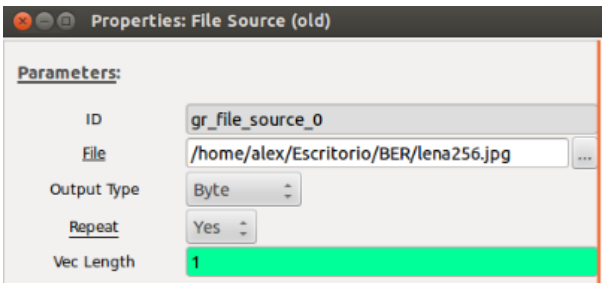

Figura D.41. Configuración de los bloques **File Source**.
Fuente: Imagen propia de los autores.

En las propiedades del archivo *file* se coloca la ruta de las imágenes a cargar, la salida que tiene esta imagen es de tipo *byte* y para que continuamente se repita la trasmisión de esta imagen se activa la función repetir (si se desea).

En el bloque de la tasa de error mostrado en la Figura D.42 se elige el tipo de error que se quiere comparar y en este caso va a ser la BER. Se debe colocar la misma cantidad de bits por símbolo que se desempaquetaron anteriormente. Este bloque va a tener dos entradas que va a ser la de imagen que se está transmitiendo y la otra para la imagen receptada y una salida para visualizar el BER obtenido en un sumidero numérico, cuya configuración se muestra en la Figura D.43.

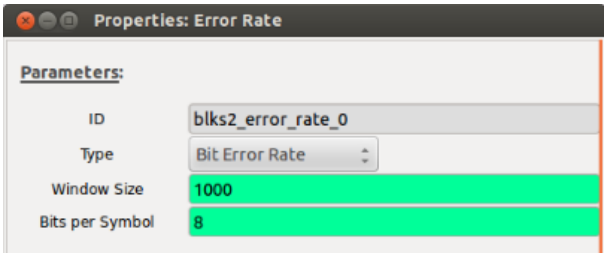

Figura D.42. Propiedades del bloque **Error Rate**.
Fuente: Imagen propia de los autores.

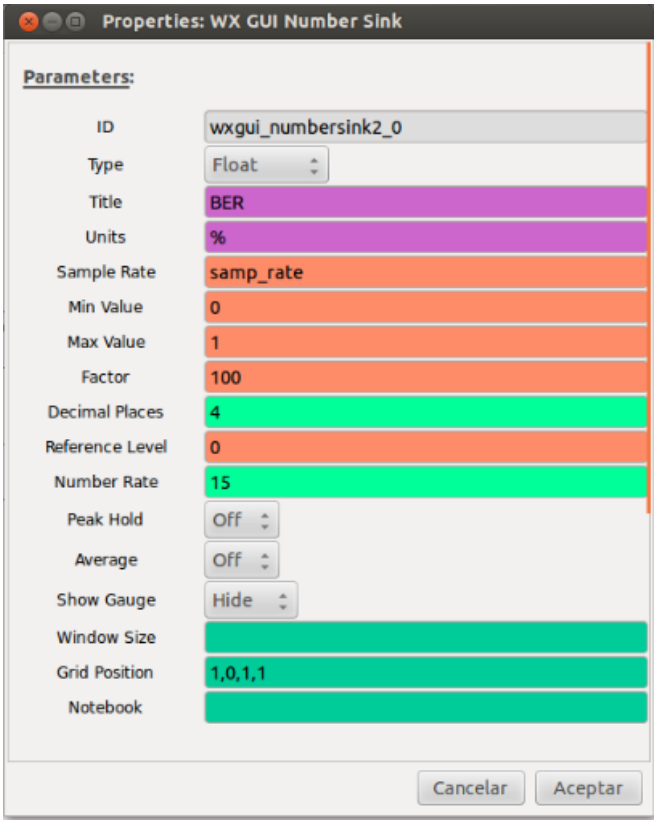

Figura D.43. Propiedades del sumidero **Number Sink**.
Fuente: Imagen propia de los autores.

Las unidades que se van a tener serán un porcentaje; la tasa de muestreo va a ser 32Kbps y finalmente se presenta el resultado en la Figura D.44.

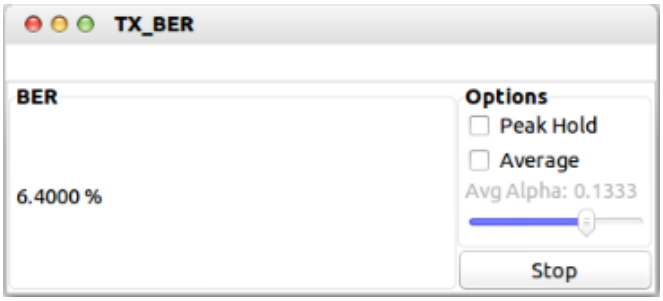

Figura D.44. Cálculo de la tasa de Bit errados (BER).
Fuente: Imagen propia de los autores.

ANEXO E

INSTALACIÓN DEL SOFTWARE MPLAYER

En este anexo se presentará la instalación de MPlayer desde el código fuente, incluyendo también los requisitos previos, *codecs* más importantes e indicando como se ejecuta.

E.1 Instalación de requisitos previos

Se instalará las herramientas necesarias para compilar MPlayer, pero previamente se debe ejecutar los comandos *update* y *upgrade*, que nos servirán para actualizar los paquetes e instalarlos a la última versión respectivamente, los comandos se muestran en la Lista E.1.

```
# sudo apt-get update
# sudo apt-get upgrade
# sudo aptitude install build-essential manpages-dev autoconf automake libtool flex bison libjpeg-dev
# sudo aptitude install libpng-dev gcc-doc x-window-system-core g++ libgtk1.2-dev gcc-3.4
```

Lista E.1. Instalación de requisitos previos.

E.2 Instalación de MPlayer

Luego se procede a instalar el *software* en sí, para ello hay como hacerlo de tres formas: desde la terminal, desde el centro de *software* de Ubuntu y mediante el gestor de paquetes Synaptic.

Se ha elegido hacerlo desde la terminal de Ubuntu y se lo hace con el comando:

```
# sudo apt-get install mplayer.
```

Lista E.2. Instalación de MPlayer.

E.3 Instalación de los *codecs* de MPlayer

Para que MPayer pueda reproducir los archivos de video se necesita de *codecs* y para ello los descargamos las siguientes listas de comandos:

```
# wget http://www2.mplayerhq.hu/MPlayer/releases/codecs/ essential-20061022.tar.bz2
```

Lista E.3. *Codecs* de MPlayer.

Procedemos a descomprimirlo.

```
# tar xjvf essential-20061022.tar.bz2
```

Lista E.4. Descomprimir *Codecs* de MPlayer.

Creamos el directorio que almacenara los *codecs.*

```
# sudo mkdir -p /usr/local/lib/codecs
```

Lista E.5.Creación de directorio.

Copiamos los *codecs.*

```
# sudo cp essential-*/* /usr/local/lib/codecs/
```

Lista E.6. Copia de *codecs* en directorio.

E.4 Instalación de los *codecs* de MPlayer

Ingresamos al directorio donde se instaló MPlayer.

```
# cd MPlayer
```

Lista E.7.Ingreso al directorio.

Creamos el Makefile, que incluye las instrucciones de compilación.

```
# ./configure --enable-gui
```

Lista E.8. Creación del Makefile.

Compilamos.

```
# make
```

Lista E.9. Compilación de Mplayer.

Copiamos los binarios ya compilados del programa y lo instalamos

```
# sudo make install
```

Lista E.10. Instalación en modo de usuario privilegiado.

La reproducción de video con MPlayer es muy sencilla, se lo realiza con el comando **mplayer** y el nombre del archivo de video como se lo muestra a continuación.

```
# mplayer video1.mp4
```

Lista E.11. Reproducción de un archivo de video.

ANEXO F

INSTALACIÓN DEL SOFTWARE GSTREAMER

F.1 Descarga de Repositorios

Se descarga y actualiza el repositorio de donde se descarga GStreamer.

```
# sudo apt-add-repository ppa:gstreamer-developers/ppa
# sudo apt-get update
```

Lista F.1. Descarga de Repositorios.

F.1 Instalación Gstreamer

Procedemos a instalar GStreamer y las herramientas necesarias para su correcto funcionamiento.

```
# sudo apt-get install libgstreamer0.10-0# sudo apt-get update
# sudo apt-get install gstreamer0.10-tools gnome-media
```

Lista F.2. Instalación Gstreamer.

ANEXO G

PAPER DEL PROYECTO DE FIN DE TITULACIÓN

Desarrollo de sistemas de comunicaciones usando tecnología libre de Radio Definida mediante Software

Alexander Sócola[1], Max Peralta[2], Manuel Quiñones[3]
Departamento de Ciencias de la Computación y Electrónica
Universidad Técnica Particular de Loja, UTPL
Loja, Ecuador
Email: [1]apsocola@utpl.edu.ec, [2]maperalta3@utpl.edu.ec
[3]mfquinones@utpl.edu.ec

***Abstract*—In the present research, communications systems using software defined radio (SDR) are developed. This technology allows the integration of free software and flexible hardware, making the same hardware device be able to perform different functions at different time instants through changes in programming by reconfigurable software. To fulfill this purpose we have employed the USRP N210 device, which uses a Gigabit Ethernet interface for communication with the computer, Ubuntu 13.04 (base operating system) and GNU Radio for programming different communication systems.**

***Index Terms*—SDR, GNU Radio, USRP N210, Ubuntu, Flexible Hardware, Reconfigurable Software.**

I. Introducción

Actualmente existen sistemas de comunicaciones que son empleados para una función específica y que operan dentro de una banda determinada; en algunos casos, se necesita de varios dispositivos para poder implementarlos. La tecnología SDR (*Software Defined Radio*) permite sustituir algunos componentes de *hardware* por rutinas de *software* reduciendo considerablemente costos y el tamaño de dichos sistemas.

El término SR (*Software Radio*) fue acuñado por Joseph Mitola III, en 1991, para referirse a un tipo de radios reprogramables o reconfigurables [1].

La primera implementación conocida del concepto SDR fue el proyecto militar estadounidense *SpeakEasy*, cuyo objetivo principal era establecer más de diez tipos de tecnologías de telecomunicaciones inalámbricas (las más usadas por el ejército americano) en un solo equipo programable, que operaría en un rango de frecuencias desde los 2 MHz hasta los 200 MHz [1].

Algunas de las características de emplear tecnología SDR se mencionan a continuación:

- Las aplicaciones de los diferentes tipos de radio existentes son fáciles de implementar, además de ofrecer nuevos tipos de aplicaciones.
- Desarrollo de Radio Cognitiva mediante la flexibilidad y capacidades computacionales de SDR.
- Flexibilidad en la actualización de *software*.

El desarrollo y programación de los diferentes sistemas de comunicaciones se lo realizó en GNU Radio *Companion* que es un entorno gráfico de programación a través de bloques que contienen moduladores analógicos y digitales, filtros, operadores aritméticos, etc., por medio de la combinación de los mismos se efectúa el procesamiento de señales; la comunicación con el USRP N210 se la realiza a través de la interfaz Gigabit Ethernet, el USRP es el dispositivo que realiza la conversión analógica a digital (ADC) o digital a analógica (DAC), así como la conversión de radiofrecuencia (RF) a una frecuencia intermedia (IF) o viceversa; este proceso fue desarrollado sobre el sistema operativo Ubuntu que es de libre distribución.

La parte restante de este artículo está estructurado de la siguiente manera: en la sección 2 se da una descripción de la radio refinida mediante *software*, la sección 3 se refiere al estudio de la plataforma *hardware*, en la sección 4 se revisan las herramientas de *software* usadas para los diferentes sistemas de comunicaciones, en 5 se presenta el análisis de resultados, en la sección 6 se detallan trabajos futuros y finalmente en la sección 7 se presentan las conclusiones obtenidas luego de la implementación de los sistemas.

II. Radio definida mediante software

SDR es una tecnología de comunicación de radio basada en un protocolo de comunicaciones inalámbricas definidas por *software* en lugar de su implementación en *hardware*, es capaz de ser reprogramado y reconfigurado con el fin de operar con diferentes formas de onda y protocolos. SDR proporciona una solución eficiente y de bajo costo al construir o implementar dispositivos inalámbricos multimodo, multiportadora y multibanda que pueden ser actualizados o

mejorados a través de optimizaciones de *software* [2].

Los desarrollos en radios inteligentes y adaptativos se han enmarcado a lo que hoy es un radio definido mediante *software*, el cual es definido, según el *Wireless Inovation Forum*, de la siguiente manera:

"Radio en el cual algunas o todas las funciones de la capa física son definidas mediante software".

III. Plataforma hardware

El presente trabajo de investigación se lo ha realizado con los dispositivos SDR USRP N210 de la empresa Ettuss Research, debido a la facilidad con que se puede implementar un sistema de comunicación y el bajo costo. Para tener totalmente en operación el sistema, sólo se necesita una tarjeta secundaria de TX/RX (la cual trabaja en distintas bandas de frecuencia de acuerdo al modelo de la misma). Esta debe estar instalada en la placa base (tarjeta principal) de propósito general, y conectado a un computador personal. El dispositivo empleado para el desarrollo de los sistemas es el USRP N210 mostrado en la Figura 1.

Fig. 1. USRP N210.

A continuación se detalla el *hardware* empleado:

A. Tarjeta principal

En el caso del USRP N210, su arquitectura emplea una FPGA (*field-programable gate array*) Spartan 3A DSP 3400, con un conversor analógico/digital (ADC) dual de 100MS/s con 14 bits y un conversor digital/analógico (DAC) dual de 400MS/s con 16 bits. Utiliza memoria SRAM de alta velocidad de 1MB y conectividad Gigabit Ethernet que permite enviar al computador hasta 50MS/s [3].

Esta placa se encarga de comunicar la señal generada vía *software* desde el *host* hacia el módulo de RF, el cuál realizará lo necesario a la señal para disponer de ésta a la frecuencia requerida por la aplicación a desarrollar. La Figura 2 muestra esquemáticamente las funciones a realizar por la placa principal [4].

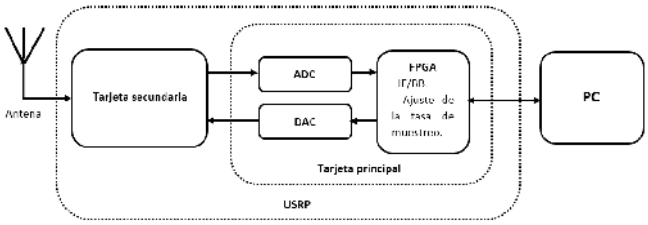

Fig. 2. Funciones de la tarjeta principal. Tomada de [4].

Su función empieza a la salida del computador y acaba cuando la señal atraviesa el DAC en el caso del transmisor, mientras que cuando se emplea como receptor su función comienza cuando la señal sale de la tarjeta secundaria y termina cuando la señal es conducida hacia el computador, encargándose de la conversión de la señal desde el dominio digital al analógico o del analógico al digital dependiendo de cuál sea el caso y adecuando la tasa de muestreo de la señal para su transmisión por medio de operaciones de diezmado e interpolación [4].

B. Tarjeta secundaria

Son las encargadas de cumplir gran parte de las funciones del transmisor y/o del receptor de radiofrecuencia clásico. La Figura 3 muestra esquemáticamente las funciones que realiza la placa secundaria:

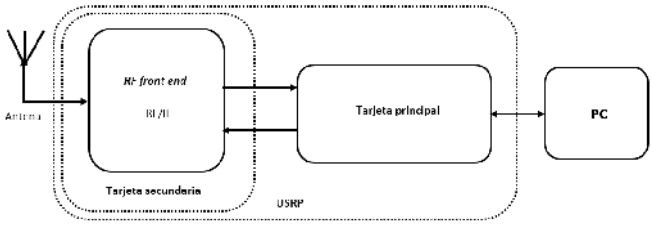

Fig. 3. Funciones de la tarjeta secundaria. Tomada de [4].

Con el USRP N210 se emplearon tarjetas WBX, las cuales presentan una potencia máxima de salida de 100mW y un ancho de banda de 40MHz. Dicha placa está diseñada también para soportar MIMO (*multiple inputs-multiple outputs*). El rango de operación de la placa es de 50MHz-2200MHz presentando ganancias variables tanto en la cadena del transmisor como en la del receptor, en el caso del transmisor la ganancia variable va desde 0dB hasta los 25dB mientras que en el receptor va desde 0dB hasta los 31.5dB, su figura de ruido va desde 5dB hasta 10dB [5].

En la Figura 4 se muestra la tarjeta secundaria WBX RX/TX.

Fig. 4. Imagen de la tarjeta WBX. Imagen tomada de [5]

Su función empieza a la salida del DAC y termina cuando es conducida la señal a transmitir hasta el conector SMA en el caso del transmisor, en el caso del receptor empieza cuando la señal llega al conector SMA y finaliza cuando ésta es conducida hacia el ADC, por lo tanto este tipo de placas se conecta físicamente a la tarjeta principal [4].

Cada tarjeta secundaria cuenta con una *electrically eraseable programmable read-only memory* (EEPROM), la cual aparte de servir como identificador para el sistema, permite establecer al *host* la configuración adecuada para su uso correcto (por ejemplo los distintos rangos de ganancia permitidos) [4].

C. VERT900

Es una antena omnidireccional vertical para las bandas de 824 a 960 MHz y la banda de 1710 a 1990 MHz Quad-Band Celular/PCS y banda ISM, con una longitud de 9 pulgadas y con una ganacia de 3dBi [6] como se aprecia en la Figura 5.

Fig. 5. Antena VERT900. Imagen tomada de [6].

D. GPSDO

El dispositivo *global positioning system disciplined oscillator* (GPSDO), permite una sincronización precisa de muestras para transmisión TX y recepción RX en el USRP, generando una frecuencia de reloj de 10MHz en su oscilador de salida con una precisión de 0.01ppm (partes por millón), acercándose a la frecuencia de 13MHz con 0.02ppm requerido en la mayoría de las implementaciones GSM tanto para la generación de frecuencia como para la sincronización del reloj [7].

El USRP N210 tiene incorporado en su tarjeta principal un oscilador de cristal de temperatura compensada (TCXO) con precisión de 2.5ppm, muy baja para trabajar como reloj principal en la red. Por lo que se recomienda usar la señal del GPSDO como entrada de reloj de referencia REF en la tarjeta principal del USRP. Los pasos correctos para la conexión del GPSDO con el equipo USRP los podemos encontrar en la guía desarrollada por el fabricante [8].

En la Figura 6 se presenta la arquitectura del USRP N210 con todos sus módulos conectados:

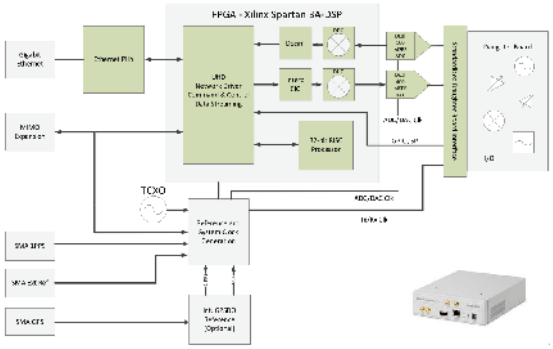

Fig. 6. Arquitectura SDR del USRP N210. Imagen tomada de [9].

IV. Plataforma Software

Para que el dispositivo USRP N210 pueda interactuar con el computador es necesario un *driver* denominado UHD (USRP *hardware driver*), que puede funcionar en los sistemas operativos: Linux, Windows y Mac.

Posibilita una interfaz de programación de aplicaciones (API) para la investigación de nuevos servicios que se pueden adaptar a los radios definidos mediante software. Los usuarios de UHD pueden realizar la comunicación del *hardware* USRP bajo plataformas de desarrollo de software como: GNU Radio, LabVIEW, MATLAB, OpenBTS.

Para la implementación de los sistemas de comunicación se ha empleado la plataforma GNU Radio *Companion*.

A. GNU Radio

El control y adquisición de datos desde el USRP se inició como un proyecto denominado software de código abierto OSS (*open source software*) conocido como GNU Radio, fundado por Eric Blossom [10] en 1998 con el objetivo de desarrollar un marco de trabajo para *software radio*. Se trata de una herramienta *software* libre y de código abierto, constituida por un conjunto de archivos y librerías que proporcionan bloques de procesamiento de señales, permitiendo así el diseño y la simulación de sistemas basados en *software radio* [10].

Dentro de GNU Radio existen dos niveles que un programador puede observar. Bloques de código de bajo nivel que son escritos en C++ para eficiencia, y consiste de componentes de procesamiento de señal pequeños como podemos apreciar en la Figura 7 [10]. Los códigos de alto nivel están

escritos en Python que principalmente su conexión es de varios bloques de procesamiento de señales juntos en un solo grafo dirigido. Como Python es un lenguaje interpretado no requiere tiempo adicional de compilación durante desarrollo o experimentación; y esto puede ser usado en beneficio del desarrollador para el despliegue rápido de aplicaciones RAD textitrapid application deployment) [10].

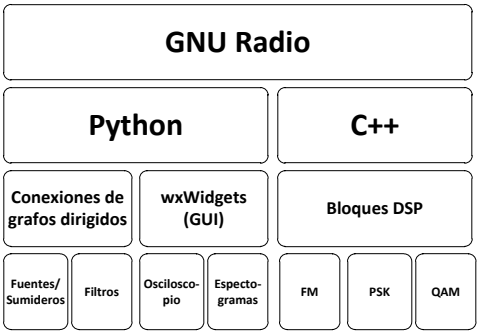

Fig. 7. Arquitectura de GNU Radio. Tomado de [10]

Por lo tanto, la arquitectura de GNURadio se basa en un híbrido de C++/Python donde los bloques de procesamiento digital de señales son implementados en C++, mientras que los grafos, las reglas de definición del funcionamiento y las opciones de configuración se aplican en Python [11].

1) GNU Radio Companion: Esta herramienta gráfica permite al usuario la implementación de sistemas de procesamiento de señales para comunicaciones sin requerir conocimientos previos en Python ya que interpreta los bloques y los convierte a código de lenguaje de programación Python. Esta interfaz no es un compilador, solo es un intérprete [12].

GNU Radio *Companion* surge como alternativa a la programación directa en Python de la aplicación, se trata de una interfaz que permite el diseño de sistemas mediante programación visual. Esta herramienta es muy similar a Simulink de Matlab o Labview que son un entorno de programación gráfico.

Para realizar un esquemático en GNU Radio *Companion* basta con arrastrar los bloques que se necesita hacia el entorno de programación e interconectarlos como se puede apreciar en la Figura 8, en el orden preciso y dependiendo de las características del sistema que vamos a implementar la cantidad de bloques variará.

B. GStreamer

GStreamer es un *framework* para la creación de aplicaciones de sistemas multimedia. El diseño fundamental proviene de "*video pipeline*" en la Universidad de Oregon, así como algunas ideas de DirectShow [13].

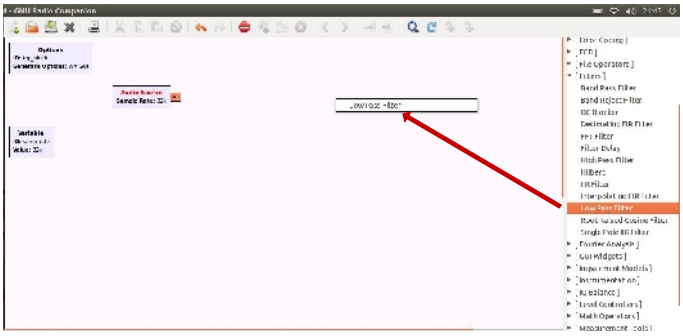

Fig. 8. Entorno de programación de GNU Radio-companion.

La estructura de desarrollo de GStreamer hace posible la escritura de cualquier tipo de aplicaciones de transmisión multimedia, haciendo posible la implementación de manera sencilla para manejar audio, video o ambos a la vez como se lo puede observar en la Figura 9. El diseño de *pipeline* está hecho para tener una pequeña sobrecarga por encima de lo que inducen los filtros aplicados [13].

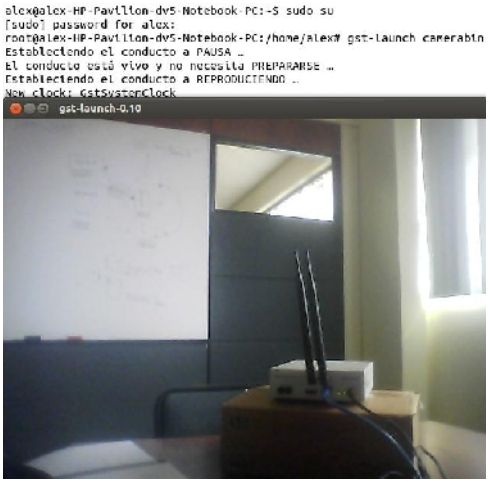

Fig. 9. Captura de video con gstreamer desde la webcam de la PC.

C. MPlayer

MPlayer es un reproductor multimedia libre. Reproduce la mayoría de los archivos MPEG, VOB, AVI, OGG/OGM, VIVO, ASF/WMA/WMV, QT/MOV/MP4, FLI, RM, NuppelVideo, YUV4MPEG, FILM, RoQ, PVA, soportados por algunos *codecs* nativos, XAnim, y DLL Win32. Además puede reproducir VideoCD, SVCD, DVD, 3ivx y DivX 3/4/5. En la Figura 10 se observa la reproducción de u archivo de video con Mplayer.

Junto al paquete de descarga de MPlayer, se puede encontrar la aplicación MEncoder, una herramienta esencial para el proceso de codificación de video o audio. Además trae por defecto un GUI hecho en GTK, Qmplayer, aunque

existen también algunos otros GUI's más potentes como por ejemplo KMPlayer, el cual está hecho en Qt. [14]

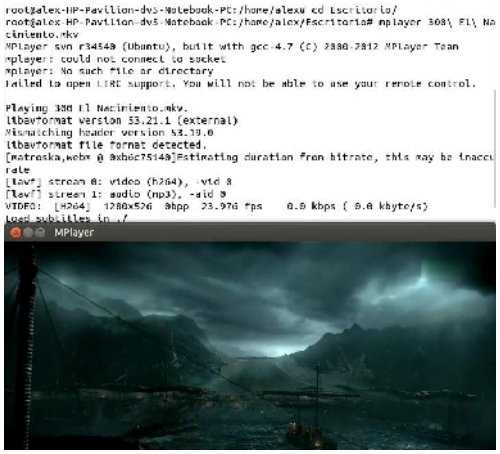

Fig. 10. Reproductor MPlayer.

V. Implementación y Resultados

Para la implementación de los sistemas de comunicaciones planteados se ha empleado un esquema transmisor-receptor; utilizando los siguientes equipos:

- Dos computadores portátiles con sistema operativo Ubuntu 13.04; una de ellas consta con un procesador Intel Core I5 y 6GB de RAM y la otra un procesador AMD Tourion II Dual Core de 4GB de RAM. Cualquiera de ellas puede funcionar como transmisor y como receptor.
- Dos equipos SDR USRP N210 con las características anteriormente mencionadas.
- Para la transmisión de video se empleará la *webcam* propia del computador.

En la Figura 11 se muestra el esquema de trabajo transmisor-receptor de los equipos SDR correctamente conectados.

Fig. 11. Esquema de trabajo TX-RX.

A. Analizador de espectros

En este apartado se procedió al desarrollo de un analizador de espectros, que opera en una banda de frecuencias desde 50MHz a 2.2GHz debido a las limitaciones de la tarjeta secundaria WBX.

En la Figura 12 se muestra el esquema de bloques del analizador de espectros.

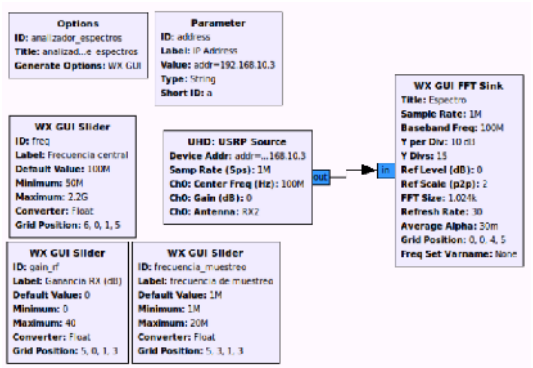

Fig. 12. Esquema del analizador de espectros.

Como podemos observar su configuración es sencilla; solo es necesario añadir dos bloques principales que son: **UHD: USRP Source**; que es el encargado de recibir las señales desde el USRP a través de la interfaz Gigabit Ethernet para su procesamiento en el computador y el bloque **FFT Sink** que nos permite visualizar el espectro de las señales al aplicar la FFT (Transforada Rápida de Fourier) a las mismas.

En la Figura 13 se presentan los resultados obtenidos, pudiendose observar el espectro de varias emisoras del dial FM.

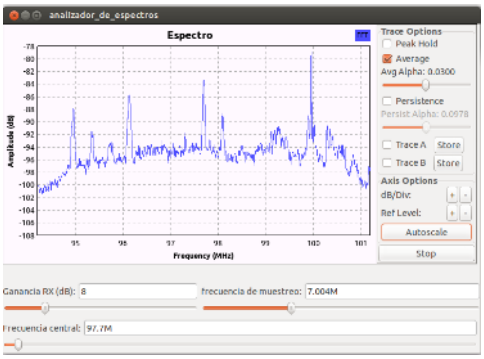

Fig. 13. Espectro de señales FM.

B. Sistema de transmisión FM

Para la implementación de un transmisor FM se han empleado los bloques y configuraciones mostradas en la Figura 14. Como se puede observar en este caso la señal moduladora

es un archivo de audio que se carga en formato **.wav**, luego esta señal pasa por una etapa de filtrado para su posterior modulación en banda base (ya sea de banda ancha o banda angosta) y finalmente se utiliza el bloque **UHD: USRP Sink** para realizar el traslado en frecuencia y realizar la transmisión. También se han implementado bloques **FFT Sink** y **Scope Sink** para la visualización de la señal modulada en FM y del espectro de audio respectivamente.

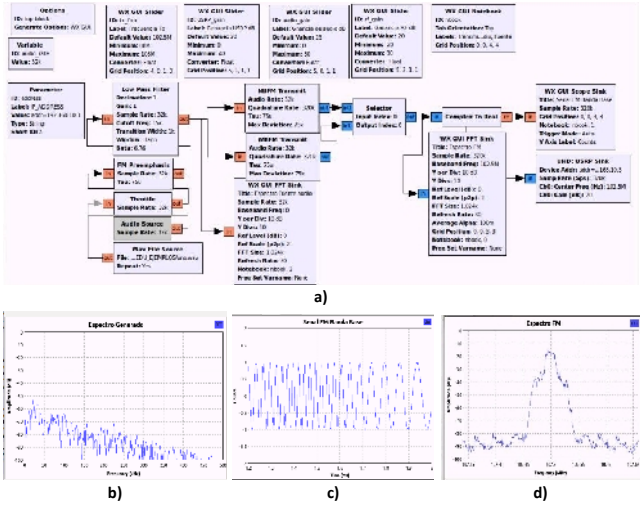

Fig. 14. Esquema del Transmisor FM.

C. Sistema de recepción FM

Para este receptor las señales son recibidas desde del USRP a través de la interfaz Gigabit Ethernet por medio del bloque **UHD:USRP Source**. A continuación estas señales pasan por un proceso de filtrado para su posterior demodulación llevada a cabo por el bloque **WBFM Receive**; finalmente la señal FM demodulada pasa por el bloque **Multiply Const** que servirá para controlar el volumen de la señal que se reproducirá en los altavoces; función que cumple el bloque **Audio Sink**. Como podemos observar en la Figura 15

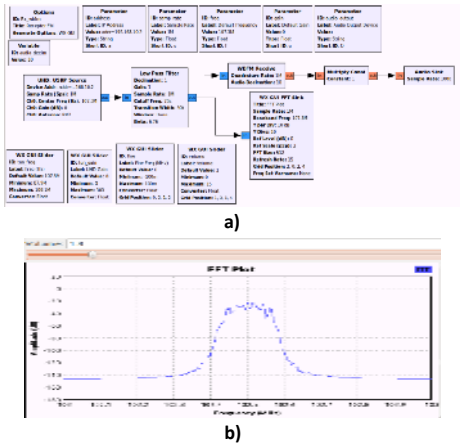

Fig. 15. a) Esquema del Receptor FM, b) Espectro de la señal FM recivida.

D. Sistema de transmisión FM con múltiples portadoras y múltiples fuentes de audio

En este caso tendremos tres fuentes de audio del tipo **.wav**, se mantiene el esquema del transmisor FM normal para la portadora central. Luego de esto para adicionar las nuevas portadoras se emplean los bloques **Signal Source** mediante los cuales se modularán las otras dos fuentes de audio; para ello basta con multiplicar estas portadoras por las señales previamente moduladas. El esquema del transmisor se muestra en la Figura 16.

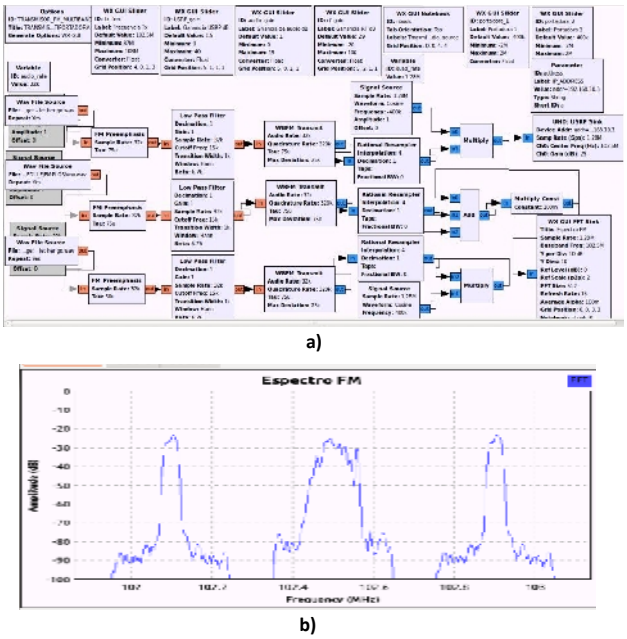

Fig. 16. a) Esquema del transmisor FM con múltiples portadoras y múltiples fuentes de audio, b) espectro de las señales FM transmitidas.

E. Sistemas de comunicaciones digitales

En esta sección se procederá al desarrollo de un sistema de comunicación digital; el mismo que constará de un transmisor y un receptor. Para este sistema se podrán seleccionar tres tipos de modulaciones digitales: M-QAM, M-PSK y GMSK.

1) Transmisor digital: En la Figura 17 se muestra el esquema del transmisor digital:

Como podemos observar en la Figura 17 se utilizará como fuente de información una señal de audio generada a través del micrófono del computador, esta señal es obtenida mediante el bloque **Audio Source** . Cabe mencionar que se puede emplear otros tipos de fuentes como: **Signal Source** que es un bloque generador de señales o el bloque **Wav File Source** que permite cargar archivos de audio desde el computador en formato **.wav**.

El bloque **Throttle** tiene la función de limitar el uso

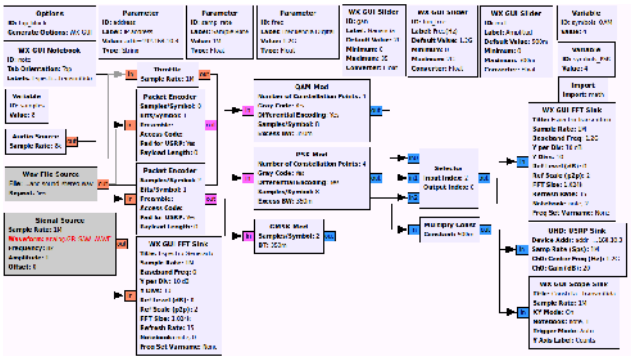

Fig. 17. Esquema del Transmisor Digital.

de la CPU; el bloque **Packet Encoder** tiene la función de transformar las muestras tipo float en tipo byte empaquetando los datos. Luego se procede a modular estas señales, mediante el selector se escoge la modulación deseada y a través del bloque **Scope Sink** podemos observar el diagrama de constelación obtenido.

En la Figura 18 se muestran los resultados del transmisor digital con una señal de audio empleando modulación **4-QAM**, en la que se aprecia el espectro de la señal de audio generada por el micrófono del computador, el diagrama de constelación y el espectro transmitido en la frecuencia 1.2 GHz.

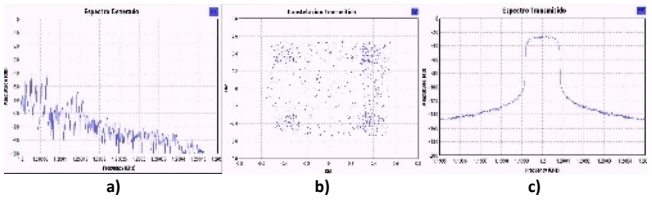

Fig. 18. Esquema del Transmisor Digital con modulación 4-QAM.

En la Figura 19 se muestran los resultados del transmisor digital con una señal de audio empleando modulación **4-PSK**, en la que se aprecia el espectro de la señal de audio generada por el micrófono del computador, el diagrama de constelación y el espectro transmitido en la frecuencia 1.2 GHz.

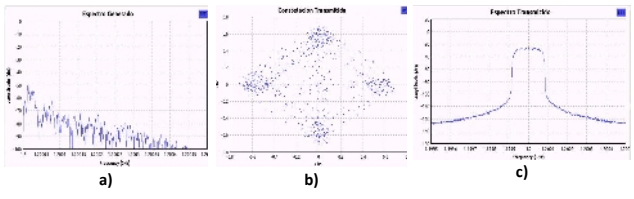

Fig. 19. Esquema del Transmisor Digital con modulación 4-PSK.

En la Figura 20 se muestran los resultados del transmisor digital con una señal de audio empleando modulación **GMSK**, en la que se aprecia el espectro de la señal de audio generada por el micrófono del computador, el diagrama de constelación y el espectro transmitido en la frecuencia 1.2 GHz.

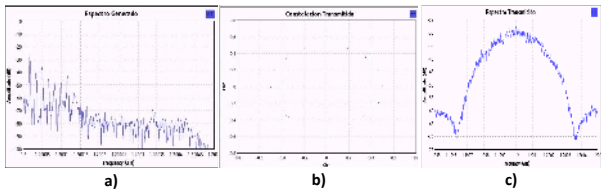

Fig. 20. Esquema del Transmisor Digital con modulación GMSK.

El bloque **Multiply Const** tiene la función de adecuar la señal a niveles soportados para la tarjeta secundaria y finalmente se procede al translado en frecuencia mediante el bloque **USRP Sink**.

2) Receptor digital: En esta sección se desarrollará el proceso de demodulación de la señal que se emitió en el transmisor, el esquemático del receptor se muestra en la Figura 21:

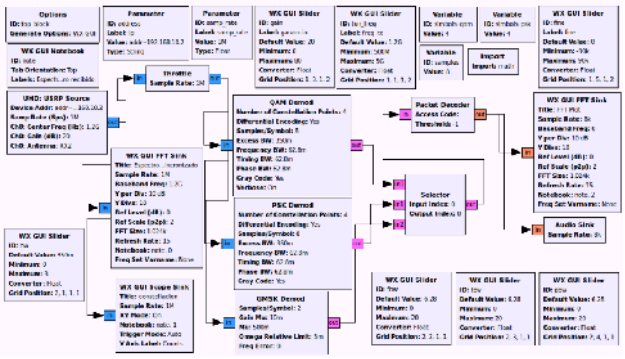

Fig. 21. Esquemático del receptor digital.

Mediante el bloque **UHD: USRP Source** se receptan las señales emitidas por el transmisor digital, realizado anteriormente. Seguidamente se colocó el bloque **Throttle** que tiene la función de limitar el uso de la CPU ; luego de esto se procede a demodular la señal escogiendo la misma a través del selector, posterior a esto la señal demodulada pasa por el bloque **Packet Decoder** convirtiendo la señal del tipo byte a float desempaquetando los datos para poder escuchar la señal enviada a través de los altavoces del computador por medio del bloque **Audio Sink** .

En la Figura 22 se presentan los resultados obtenidos del receptor digital empleando un demodulador **4-QAM**, en la que se aprecia el espectro recibido en la frecuencia 1.2GHz, el diagrama de constelación receptado y el espectro de audio decodificado.

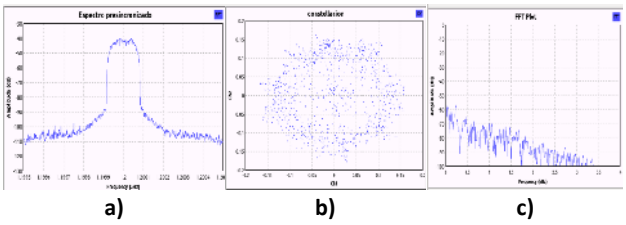

Fig. 22. Esquema del receptor digital con modulación 4-QAM.

En la Figura 23 se presentan los resultados obtenidos del receptor digital empleando un demodulador **4-PSK**, en la que se aprecia el espectro recibido en la frecuencia 1.2GHz, el diagrama de constelación receptado y el espectro de audio decodificado.

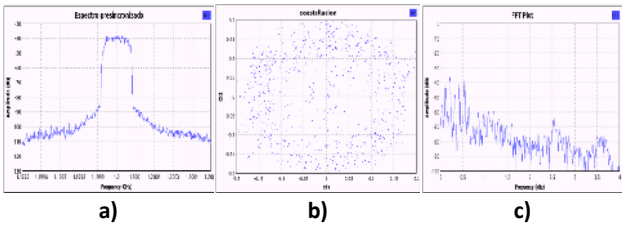

Fig. 23. Esquema del receptor digital con modulación 4-PSK.

En la Figura 24 se presentan los resultados obtenidos del receptor digital empleando un demodulador **GMSK**, en la que se aprecia el espectro recibido en la frecuencia 1.2GHz, el diagrama de constelación receptado y el espectro de audio decodificado.

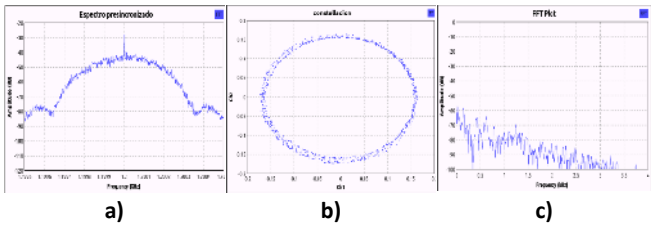

Fig. 24. Esquema del receptor digital con modulación GMSK.

F. Sistema de comunicación experimental para la transmisión y recepción digital de video

Se ha implementado un sistema de transmisión y recepción de video en base al sistema de transmisión - recepción digital y con la ayuda de las herramientas de *software*: **Gstreamer** que será utilizada para capturar y codificar el video obtenido desde la *webcam* del computador en fotmato mpeg4; esta señal de información será procesada y transmitida usando la modulación GMSK a través del USRP. Además se empleará la herramienta de software **Mplayer** para la parte de recepción, que nos permitirá decodificar y reproducir la señal de video recibida.

1) Transmisor de video: Primeramente se debe crear un archivo con extensión *.ts* en el computador que funcionará en conjunto con un USRP N210 como transmisor, este archivo establecerá un conducto entre Gstreamer y GNU Radio para el flujo continuo de video desde la *webcam* del computador.

La creación de este archivo se realiza en la terminal de Ubuntu mediante el comando:

 # mkfifo txfifo.tx

Para iniciar el flujo de video se emplea el siguiente comando:

*# gst-launch v4l2src device=/dev/video0 ! video/x-raw-yuv, width=640, height=480 ! timeoverlay halign=right valign=bottom shaded-background=true ! textoverlay text="Video de Prueba 30fps 640*480 UTPL" halign=left valign=bottom shaded-background=true ! ffmpegcolorspace ! x264enc bitrate=420 ! mpegtsmux ! filesink location=txfifo.ts*

Con este flujo de video creado se procede a ejecutar el esquema del transmisor digital de video que se muestra en la Figura 25 y el espectro generado en 1.2805 GHz con modulación GMSK.

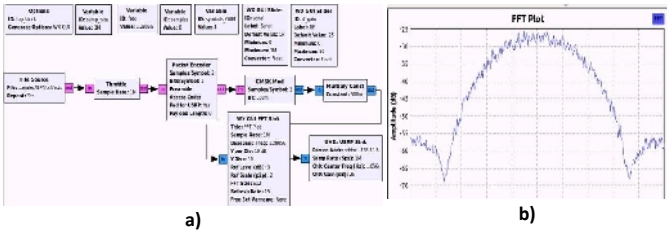

Fig. 25. Esquema del Transmisor de Video.

2) Receptor de video: Para la recepción del flujo de video es necesario crear otro archivo en formato *.ts* en el computador que funcionará en conjunto con un USRP N210 como receptor que en este caso cumplirá la función de conducto entre **GNU Radio** y **Mplayer**.

La creación de este archivo se realiza en la terminal de Ubuntu mediante el comando:

 # mkfifo rxfifo.tx

Seguidamente se prepara este conducto para la recepción del flujo de video mediante el comando:

 # mplayer rxfifo.tx

En la Figura 26 se observa el esquema del receptor de video y el espectro recibido por parte del transmisor en la frecuencia de 1.2805 GHz.

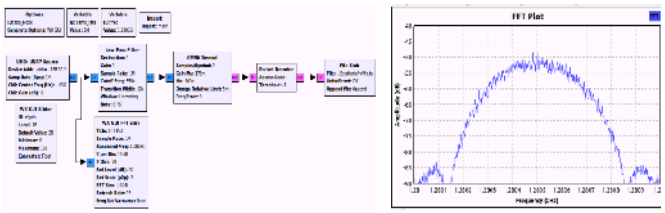

Fig. 26. Esquema del Transmisor de Video.

Al ejecutar el módulo del receptor el conducto detectará el formato del flujo de datos entrante que es *.ts* como se aprecia en la Figura 27, luego de esto se debe esperar unos segundos para sincronización, el programa reconocerá el video recibido en formato *mpeg-4* y comenzará la reproducción del mismo.

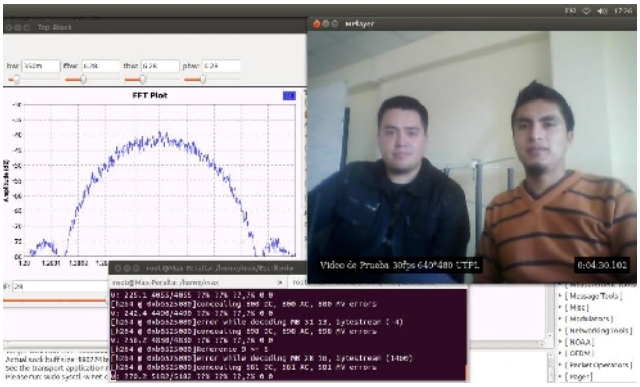

Fig. 27. Reproducción de la señal de video recibid y decodificada.

VI. Trabajo futuro

Se podría realizar o esperar la inclusión de módulos en GNU Radio que tengan soporte para los distintos estándares de televisión digital, de esta manera se podría tener un sistema de radiodifusion de TV digital a muy bajo costo comparado con los sistemas actuales.

VII. Conclusiones

En función de la experiencia obtenida con las plataformas libres para implentación de SDR se llega a las siguientes conclusiones:

- La tecnología SDR está definida por dos partes que son *hardware* y *software*; la primera es la encargada de transformar las señales de radiofrecuencia a banda base o frecuencia intermedia (IF) por medio de la tarjeta secundaria; y la segunda parte es la encargada del procesamiento digital de señales en banda base que se realiza en el computador, logrando así implementar sistemas de comunicaciones tanto analógicos como digitales con *hardware* reducido.
- El equipo SDR USRP N210 opera en un amplio rango de frecuencias (50 - 6000MHz); dependiendo de las características de la tarjeta secundaria empleada. Hemos empleado la tarjeta secundaria WBX que opera en un rango de frecuencias de 50 a 2200MHz, permitiéndonos desarrollar sistemas de comunicaciones analógicos y digitales en las bandas VHF y UHF.
- Se demuestra que mediante la tecnología SDR es posible implementar sistemas de comunicaciones analógicos y digitales de bajo costo comparado con los sistemas tradicionales y con un tiempo reducido de despliegue ya que solo es necesario la conexión del USRP con el computador, esto se logra al sustituir algunos componentes de *hardware* por rutinas de *software*.
- Se debe desarrollar los sistemas de comunicaciones en un entorno de simulación a través de fuentes y sumideros virtuales en GNU Radio *Companion* con la finalidad de probar su correcto funcionamiento y luego de esto proceder a su migración a plataformas SDR; es decir, haciendo uso de dispositivos de *hardware*.
- Al migrar los sistemas de comunicaciones a un entorno real se presentarán efectos indeseados en el canal de telecomunicaciones como: ruido, distorsión, desplazamiento de fase, frecuencia, etc.; siendo importante simular estos efectos aisladamente para conocer su influencia sobre las señales de RF y poder reducirlos a niveles que no afecten el correcto funcionamiento de los sistemas.
- En la programación de los sistemas de comunicaciones es necesario establecer parámetros de diseño como la tasa de muestreo, los índices y tipos de modulación, la cantidad de bits y muestras por símbolo de las diferentes modulaciones a emplearse; en el caso de la tasa de muestreo es posible utilizar operaciones de decimación e interpolado con la finalidad de acoplar la señal a las diferentes etapas de procesamiento.

References

[1] J. H. Aguilar Rentería and A. Navarro Cadavid, "Radio cognitiva - Estado del arte," *Sistemas y Telemática;Vol.9 No.16*, 2011.

[2] M. Abirami, V. Hariharan, M. Sruthi, R. Gandhiraj, and K. Soman, "Exploiting gnu radio and usrp: An economical test bed for real time communication systems," in *Computing, Communications and Networking Technologies (ICCCNT),2013 Fourth International Conference on*, July 2013, pp. 1–6.

[3] "*USRP N210*," ,[en línea], disponible en: https://www.ettus.com/product/details/UN210-KIT, [Consulta del 24/09/2014].

[4] J. P. Montero Hidalgo, "Implementación de un sistema de comunicaciones basado en software radio," Ph.D. dissertation, Departamento de Radiofrecuencia. Escuela Politécnica Superior y Universidad Autónoma de Madrid, Enero 2014.

[5] "*WBX 50-2200 MHz Rx/Tx (40MHz)*," ,[en línea], disponible en: https://www.ettus.com/product/details/WBX, [Consulta del 24/09/2014].

[6] "*VERT900 Antenna)*," ,[en línea], disponible en: https://www.ettus.com/product/details/VERT900, [Consulta del 24/09/2014].

[7] K. Dabcevic, "Evaluation of software defined radio platform with respect to implementation of 802.15. 4 zigbee," 2011.

[8] "*Installing the Ettus Research GPSDO Kit for USRP N200 Series/E100 Series*," ,[en línea], disponible en: https://www.ettus.com/content/files/gpsdo-kit_4.pdf, [Consulta del 24/09/2014].

[9] ETTUS RESEARCH, "*USRP N200/N210 NETWORKED SERIES*," ,[en línea], disponible en: https://www.ettus.com/content/files/07495_Ettus_N200-210_DS_Flyer_HR_1.pdf, [Consulta del 23/09/2014].

[10] D. Tucker and G. Tagliarini, "Prototyping with gnu radio and the usrp - where to begin," in *Southeastcon, 2009. SOUTHEASTCON '09. IEEE*, March 2009, pp. 50–54.

[11] A. Kwasinski, "Analysis of vulnerabilities of telecommunication systems to natural disasters," in *Systems Conference, 2010 4th Annual IEEE*, april 2010, pp. 359 –364.

[12] V. Rodríguez and J. Sánchez, "Empowering software radio: It++ as a gnu radio out-of-tree implementation," *Latin America Transactions, IEEE (Revista IEEE America Latina)*, vol. 12, no. 2, pp. 269–276, March 2014.

[13] Wim Taymans, Steve Baker, Andy Wingo, Ronald S. Bultje, Stefan Kost, "*GStreamer Application Development Manual (1.4.3)*," ,[en línea], disponible en: http://gstreamer.freedesktop.org/data/doc/gstreamer/head/manual/manual.pdf, [Consulta del 01/10/2014].

[14] "*MPlayer*," ,[en línea], disponible en: http://www.guia-ubuntu.com/index.php?title=MPlayer, [Consulta del 01/10/2014].