



Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería Mecánica Eléctrica

**IMPLEMENTACIÓN DE LA TECNOLOGÍA RADIO DEFINIDA POR SOFTWARE (SDR)
UTILIZANDO EL MÓDULO RTL BASADO EN EL CHIP REALTEK RTL2832U Y RASPBERRY
PI VERSION 3 EN SISTEMAS DE RADIOCOMUNICACIONES**

Walter Gabriel Alexander Estupinian Cifuentes

Asesorado por el Ing. Guillermo Antonio Puente Romero

Guatemala, febrero de 2021

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA



FACULTAD DE INGENIERÍA

**IMPLEMENTACIÓN DE LA TECNOLOGÍA RADIO DEFINIDA POR SOFTWARE (SDR)
UTILIZANDO EL MÓDULO RTL BASADO EN EL CHIP REALTEK RTL2832U Y RASPBERRY
PI VERSION 3 EN SISTEMAS DE RADIOCOMUNICACIONES**

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

WALTER GABRIEL ALEXANDER ESTUPINIAN CIFUENTES
ASESORADO POR EL ING. GUILLERMO ANTONIO PUENTE ROMERO

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO ELECTRÓNICO

GUATEMALA, FEBRERO DE 2021

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA



NÓMINA DE JUNTA DIRECTIVA

DECANA	Inga. Aurelia Anabela Cordova Estrada
VOCAL I	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martínez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Christian Moisés de la Cruz Leal
VOCAL V	Br. Kevin Vladimir Armando Cruz Lorente
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANA	Inga. Aurelia Anabela Cordova Estrada
EXAMINADOR	Ing. Julio César Solares Peñate
EXAMINADOR	Ing. Carlos Eduardo Guzmán Salazar
EXAMINADOR	Ing. José Aníbal Silva de los Ángeles
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

IMPLEMENTACIÓN DE LA TECNOLOGÍA RADIO DEFINIDA POR SOFTWARE (SDR) UTILIZANDO EL MÓDULO RTL BASADO EN EL CHIP REALTEK RTL2832U Y RASPBERRY PI VERSION 3 EN SISTEMAS DE RADIOCOMUNICACIONES

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería Mecánica Eléctrica, con fecha 3 de junio del 2019.



Walter Gabriel Alexander Estupinian Cifuentes

Guatemala, 11 de agosto de 2020.

Ing. Julio César Solares Peñate
Coordinador de Área de Electrónica
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC.

Ingeniero Solares:

Por este medio me permito dar aprobación al Trabajo de Graduación titulado: "Implementación de la tecnología Radio Definida por Software (SDR) utilizando el módulo RTL basado en el chip Realtek RTL2832u y Raspberry Pi versión 3 en sistemas de radiocomunicaciones", desarrollado por el estudiante Walter Gabriel Alexander Estupinian Cifuentes con carné No. 2014-03833, ya que considero que cumple con los requisitos establecidos, por lo que el autor y mi persona somos responsables del contenido y conclusiones del mismo.

Sin otro particular, aprovecho la oportunidad para saludarlo.

Atentamente,



Ing. Guillermo Antonio Puente Romero
ASESOR
Colegiado 5898

Guillermo A. Puente R.
INGENIERO ELECTRÓNICO
COL. # 5898



Guatemala, 24 de agosto de 2020

Señor Director
Armando Alonso Rivera Carrillo
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC

Estimado Señor Director:

Por este medio me permito dar aprobación al Trabajo de Graduación titulado **IMPLEMENTACIÓN DE LA TECNOLOGÍA RADIO DEFINIDA POR SOFTWARE (SDR) UTILIZANDO EL MÓDULO RTL BASADO EN EL CHIP REALTEK RTL2832U Y RASPBERRY PI VERSIÓN 3 EN SISTEMAS DE RADIOCOMUNICACIONES**, desarrollado por el estudiante **Walter Gabriel Alexander Estupinian Cifuentes**, ya que considero que cumple con los requisitos establecidos.

Sin otro particular, aprovecho la oportunidad para saludarlo.

Atentamente,

ID Y ENSEÑAD A TODOS

Una firma manuscrita en tinta azul que parece decir 'Julio Solares Peñate'.

Ing. Julio César Solares Peñate
Coordinador de Electrónica

REF. EIME 261.2020.

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen del Asesor, con el Visto Bueno del Coordinador de Área , al trabajo de Graduación del estudiante Walter Gabriel Alexander Estupinian Cifuentes: **IMPLEMENTACIÓN DE LA TECNOLOGÍA RADIO DEFINIDA POR SOFTWARE (SDR) UTILIZANDO EL MÓDULO RTL BASADO EN EL CHIP REALTEK RTL2832U Y RASPBERRY PI VERSIÓN 3 EN SISTEMAS DE RADIOCOMUNICACIONES**, procede a la autorización del mismo.



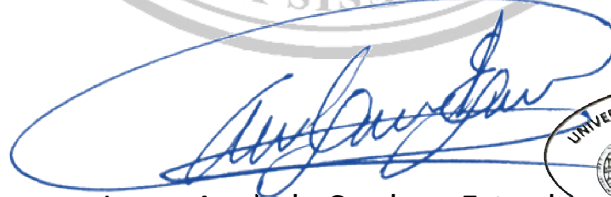
Ing. Armando Alonso Rivera Carrillo

Guatemala, 15 de octubre de 2020.

DTG. 027.2021.

La Decana de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Eléctrica, al Trabajo de Graduación titulado: **IMPLEMENTACIÓN DE LA TECNOLOGÍA RADIO DEFINIDA POR SOFTWARE (SDR) UTILIZANDO EL MÓDULO RTL BASADO EN EL CHIP REALTEK RTL2832U Y RASPBERRY PI VERSION 3 EN SISTEMAS DE RADIOCOMUNICACIONES**, presentado por el estudiante universitario: **Walter Gabriel Alexander Estupinian Cifuentes**, y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:



Inga. Anabela Cordova Estrada
Decana



Guatemala, febrero de 2021.

AACE/asga

ACTO QUE DEDICO A:

- Dios** Por permitirme llegar a este punto, culminando una parte muy importante en mi vida.
- Mis padres** Angélica Cifuentes y Walter Estupinian, por todo el cariño y apoyo que siempre me dan de manera incondicional.
- Mis hermanas** Katherine, Michelle y Jennipher Estupinian, porque siempre me motivaron a ser mejor y me apoyaron en todo.
- Mis abuelos** Evelia Castillo y Odilia Chutan, por darme siempre su cariño; Víctor Estupinian y Jorge Cifuentes (q. e. p. d.), por todas sus enseñanzas.
- Mis amigos** Por todas las aventuras, experiencias, emociones y noches de desvelo trabajando proyectos o estudiando para exámenes; por los momentos compartidos.
- Mis alumnos** A los jóvenes a quienes tuve la oportunidad de compartir mi conocimiento e impartirles el laboratorio de Comunicaciones 1, y que inspiraron este trabajo de graduación.

AGRADECIMIENTOS A:

Mi familia	Por todo el cariño y apoyo que recibo de su parte.
Universidad de San Carlos de Guatemala	Por ser mi casa de estudio y brindarme una serie de oportunidades que me permitieron crecer académicamente y como persona.
Facultad de Ingeniería	Por hacer posible el sueño de estudiar esta carrera universitaria y por la confianza brindada para representar a la Facultad en distintas actividades.
Mis amigos de la Facultad	Pablo Cazali, Daniel Fernández, Hugo Lemus, José Luis Fuentes, Byron Paiz, Marielena Ramazzini, Isis González, Melannie Chávez, Marny Morataya, Víctor Carranza, Chantelle Cruz, Simón Sica, María Fernanda Barahona y a todos aquellos que hicieron de mis años de universidad, los mejores de mi vida.
Rama estudiantil IEEE USAC	Por todas las experiencias, conocimientos, talentos y amistades que me ayudaron a desarrollar a lo largo de los años que formé parte del equipo de la rama estudiantil de IEEE de la USAC.

**Laboratorio de
Electrónica**

Por darme la oportunidad de formar parte del equipo de auxiliares de los laboratorios de Electrónica, y compartir con otros estudiantes mis experiencias y conocimientos adquiridos a lo largo de la carrera.

Ing. Otto Andrino

Por la confianza que depositó en mí al permitirme ser su auxiliar; por todos los consejos y buenos momentos compartidos en los últimos años.

Ing. Byron Arrivillaga

Por el conocimiento compartido en las actividades realizadas en el laboratorio de Electrónica, y las convivencias con el grupo de ingenieros y auxiliares.

Ing. Guillermo Puente

Por todo el apoyo y asesoría brindados durante la realización de este trabajo de graduación y las experiencias vividas a lo largo de los semestres en los que trabajamos el laboratorio de Comunicaciones 1.

**Mercury Robotics
Challenge Colombia**

Por permitirme vivir una de las mejores experiencias de mi vida, participando en una competencia de Robótica a nivel latinoamericano.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES.....	IX
LISTA DE SÍMBOLOS.....	XVII
GLOSARIO	XIX
RESUMEN	XXIII
OBJETIVOS.....	XXV
INTRODUCCIÓN.....	XXVII
1. FUNDAMENTOS DE LA MODULACIÓN DE SEÑALES EN SISTEMAS DE TELECOMUNICACIONES ELECTRÓNICAS	1
1.1. Modulación	1
1.2. Modulación analógica	2
1.2.1. Modulación en amplitud.....	2
1.2.1.1. Índice de modulación	5
1.2.1.2. Modulación banda lateral única SSB SC.	7
1.2.1.3. Modulación doble banda lateral DSB SC	8
1.2.2. Modulación angular.....	8
1.2.2.1. Modulación de fase	9
1.2.2.1.1. Modulación de frecuencia	9
1.2.2.1.2. Índice de modulación FM.....	11
1.3. Modulación digital	11
1.3.1. Modulación por desplazamiento de fase PSK.....	12

1.3.1.1.	Modulación por desplazamiento de fase binaria BPSK	13
1.3.1.2.	Modulación por desplazamiento de fase en cuadratura QPSK	15
1.3.2.	Modulación por desplazamiento de frecuencia FSK.	17
1.3.3.	Modulación por desplazamiento de amplitud ASK..	19
1.3.4.	Modulación por pulso codificado PCM.....	20
1.3.5.	Modulación por amplitud en cuadratura QAM	21
1.3.5.1.	4QAM	211
1.3.5.2.	8QAM	24
2.	TECNOLOGÍA RADIO DEFINIDA POR SOFTWARE	29
2.1.	Herramientas de hardware	29
2.1.1.	Módulo USB basado en el chip RTL2832u.....	29
2.1.2.	Raspberry pi versión 3	311
2.1.2.1.	Características Raspberry pi versión 3B+	311
2.2.	Herramientas de software.....	322
2.2.1.	GQRX (Ubuntu)	322
2.2.2.	SDR# (Windows)	333
2.2.3.	Librería RPITX para transmisión en Raspberry pi.	344
2.2.4.	GNU Radio.....	355
2.2.4.1.	Bloques	366
3.	ENLACE EN SISTEMAS DE TELECOMUNICACIONES UTILIZANDO EL MÓDULO RTL 2832U Y RASPBERRY PI 3.	411
3.1.	Radio enlace.....	411
3.1.1.	Antenas	433

	3.1.1.1.	Parámetros de antenas	433
	3.1.1.2.	Tipos de antenas.....	49
	3.1.2.	Zonas de Fresnell	555
3.2.		Raspberry pi 3 como transmisor de señales de radio frecuencia	566
3.3.		Receptor de señales de radio frecuencias con GNU Radio y módulo RTL 2832u	58
4.		PRÁCTICAS APLICABLES AL LABORATORIO DE COMUNICACIONES 1.....	61
4.1.		Práctica 1: instalación de <i>drivers</i> y software en sistema operativo Ubuntu.....	611
	4.1.1.	<i>Driver</i> módulo RTL.....	611
		4.1.1.1. Instalación <i>git</i>	611
		4.1.1.2. Instalación <i>Cmake</i>	622
		4.1.1.3. Instalación <i>libusb</i>	63
		4.1.1.4. Instalación <i>build-essential</i>	633
		4.1.1.5. Descarga de repositorio <i>driver</i> RTL- SDR.....	644
		4.1.1.6. Configuración e instalación	65
		4.1.1.7. Creación <i>blacklist</i>	688
	4.1.2.	GNU Radio	700
	4.1.3.	GQRX	700
	4.1.4.	RPITX	722
		4.1.4.1. Repositorio RPITX	733
4.2.		Práctica 2: amplitud modulada, transmisión y recepción GNU Radio.....	755
	4.2.1.	Transmisor amplitud modulada.....	766
		4.2.1.1. <i>Wav file source</i> transmisor AM.....	777

4.2.1.2.	<i>Float to complex</i> transmisor AM	777
4.2.1.3.	<i>Add constant</i> transmisor AM.....	788
4.2.1.4.	<i>Signal source</i> transmisor AM.....	79
4.2.1.5.	<i>Trottle</i> transmisor AM.....	79
4.2.1.6.	<i>Multiply</i> transmisor AM.....	800
4.2.1.7.	WX GUI FFT <i>sink</i> transmisor AM	811
4.2.1.8.	TCP <i>sink</i> transmisor AM	822
4.2.1.9.	Variable transmisor AM.....	822
4.2.2.	Receptor amplitud modulada.....	833
4.2.2.1.	Variables receptor AM.....	844
4.2.2.2.	RTL-SDR <i>source</i> receptor AM.....	844
4.2.2.3.	AM <i>demod</i> receptor AM	855
4.2.2.4.	<i>Rational resampler</i> receptor AM.....	866
4.2.2.5.	WX GUI FFT <i>sink</i> receptor AM	877
4.2.2.6.	<i>Audio sink</i> receptor AM.....	888
4.3.	Práctica 3: frecuencia modulada, transmisión y recepción	
	GNU Radio	88
4.3.1.	Transmisor frecuencia modulada banda ancha	89
4.3.1.1.	<i>Wav file source</i> transmisor WBFM.....	900
4.3.1.2.	<i>Trottle</i> transmisor WBFM	900
4.3.1.3.	AGC2 transmisor WBFM.....	911
4.3.1.4.	<i>Band pass filter</i> transmisor WBFM.....	911
4.3.1.5.	WBFM <i>transmit</i> transmisor WBFM.....	922
4.3.1.6.	TCP <i>sink</i> transmisor WBFM.....	933
4.3.1.7.	WX GUI FFT <i>sink</i> transmisor WBFM .	933
4.3.1.8.	Variable transmisor WBFM.....	944
4.3.2.	Transmisor frecuencia modulada banda angosta .	955
4.3.2.1.	<i>Wav file source</i> transmisor NBFM.....	966
4.3.2.2.	<i>Trottle</i> transmisor NBFM.....	966

4.3.2.3.	AGC2 transmisor NBFM.....	977
4.3.2.4.	<i>Band pass filter</i> transmisor NBFM	98
4.3.2.5.	NBFM <i>transmit</i> transmisor NBFM	98
4.3.2.6.	TCP <i>sink</i> transmisor NBFM.....	99
4.3.2.7.	WX GUI FFT <i>sink</i> transmisor NBFM.	1000
4.3.2.8.	Variable transmisor NBFM	1011
4.3.3.	Receptor frecuencia modulada	1022
4.3.3.1.	Variables receptor FM.....	1022
4.3.3.2.	RTL-SDR <i>source</i> receptor FM.....	1033
4.3.3.3.	<i>Low pass filter</i> receptor FM	1044
4.3.3.4.	WBFM <i>demod</i> receptor FM	1044
4.3.3.5.	<i>Rational resampler</i> receptor FM.....	1055
4.3.3.6.	<i>Audio sink</i> receptor FM.....	1066
4.3.3.7.	WX GUI FFT <i>sink</i> receptor FM.....	1066
4.4.	Práctica 4: modulación BPSK	10707
4.4.1.	Modulador BPSK.....	10808
4.4.1.1.	<i>Signal source</i> modulador BPSK.....	10909
4.4.1.2.	<i>Add constant</i> modulador BPSK.....	1100
4.4.1.3.	<i>Multiply</i> modulador BPSK.....	1111
4.4.1.4.	Variable modulador BPSK.....	1111
4.4.1.5.	WX GUI <i>scope sink</i> modulador BPSK	1122
4.5.	Práctica 5: modulación QPSK.....	1133
4.5.1.	Modulación QPSK.....	1133
4.5.1.1.	<i>Vector source</i> modulador QPSK.....	1144
4.5.1.2.	<i>Repack bits</i> modulador QPSK.....	1144
4.5.1.3.	PSK <i>mod</i> modulador QPSK	1155
4.5.1.4.	<i>Multiply</i> modulador QPSK.....	11616
4.5.1.5.	<i>Signal source</i> modulador QPSK	11616

	4.5.1.6.	Variable modulador QPSK.....	11717
	4.5.1.7.	WX GUI <i>constellation sink</i> modulador QPSK	11717
4.6.		Práctica 6: modulación ASK.....	11818
	4.6.1.	Modulador ASK.....	11919
	4.6.1.1.	<i>Signal source</i> transmisor ASK.....	1200
	4.6.1.2.	<i>Trottle</i> transmisor ASK	1211
	4.6.1.3.	<i>Multiply</i> transmisor ASK	1222
	4.6.1.4.	<i>Add constant</i> transmisor ASK.....	1222
	4.6.1.5.	WX GUI FFT <i>sink</i> transmisor ASK ...	1233
	4.6.1.6.	TCP <i>sink</i> transmisor ASK	1244
	4.6.1.7.	Variable transmisor ASK.....	1255
4.7.		Práctica 7: modulación FSK.....	12626
	4.7.1.	Modulador FSK.....	12626
	4.7.1.1.	<i>Signal source</i> modulador FSK.....	12727
	4.7.1.2.	<i>Throttle</i> modulador FSK	12929
	4.7.1.3.	<i>Not</i> modulador FSK.....	12929
	4.7.1.4.	<i>Short to float</i> modulador FSK	1300
	4.7.1.5.	<i>Multiply</i> modulador FSK.....	1300
	4.7.1.6.	<i>Add</i> modulador FSK.....	1311
	4.7.1.7.	Variable modulador FSK.....	1311
	4.7.1.8.	WX GUI <i>scope sink</i> modulador FSK	1322
4.8.		Práctica 8: modulación QAM.....	1322
	4.8.1.	Modulador QAM.....	1333
	4.8.1.1.	<i>Vector source</i> modulador QAM	1344
	4.8.1.2.	<i>Repack bits</i> modulador QAM.....	1344
	4.8.1.3.	QAM <i>mod</i> modulador QAM.....	13535
	4.8.1.4.	<i>Multiply</i> modulador QAM.....	13636
	4.8.1.5.	<i>Signal source</i> modulador QAM.....	13636

4.8.1.6.	Variable modulador QAM	13737
4.8.1.7.	WX GUI <i>constellation sink</i> modulador QAM.....	13737
CONCLUSIONES.....		13939
RECOMENDACIONES		1411
BIBLIOGRAFÍA.....		1433
ANEXOS		1455

ÍNDICE DE ILUSTRACIONES

FIGURAS

1.	Onda moduladora AM	3
2.	Onda portadora AM	3
3.	Onda modulada AM.....	4
4.	Amplitud modulada en el dominio de la frecuencia.....	5
5.	50 % de índice de modulación AM	6
6.	100 % de índice de modulación AM.....	6
7.	150 % de índice de modulación AM.....	7
8.	Onda moduladora FM.....	10
9.	Onda portadora FM	10
10.	Onda modulada FM.....	11
11.	Diagrama de constelaciones.....	12
12.	Onda portadora desfasada 0°.....	13
13.	Onda portadora desfasada 180°.....	14
14.	Diagrama de constelaciones BPSK 0° y 180°	14
15.	Onda portadora desfasada 45°	15
16.	Onda portadora desfasada 135°.....	15
17.	Onda portadora desfasada 225°.....	16
18.	Onda portadora desfasada 315°.....	16
19.	Diagrama de constelaciones BPSK 45° 135° 225° 315°	17
20.	Ejemplo de gráfica representando un 1 lógico en FSK.....	18
21.	Ejemplo de gráfica representando un 0 lógico en FSK.....	18
22.	Ejemplo de gráfica representando un 1 lógico en ASK.....	19
23.	Ejemplo de gráfica representando un 0 lógico en ASK.....	19

24.	Modulación PCM.....	20
25.	Onda portadora desfasada 45°	21
26.	Onda portadora desfasada 135°	22
27.	Onda portadora desfasada 225°	22
28.	Onda portadora desfasada 315°	23
29.	Diagrama de constelaciones QPSK 45° 135° 225° 315°	23
30.	Onda portadora desfasada 45° con amplitud A	24
31.	Onda portadora desfasada 135° con amplitud A.....	25
32.	Onda portadora desfasada 225° con amplitud A.....	25
33.	Onda portadora desfasada 315° con amplitud A.....	26
34.	Onda portadora desfasada 45° con amplitud 2A.....	26
35.	Onda portadora desfasada 135° con amplitud 2A.....	27
36.	Onda portadora desfasada 225° con amplitud 2A.....	27
37.	Onda portadora desfasada 315° con amplitud 2A.....	28
38.	Diagrama de constelaciones 8QAM	28
39.	Diagrama esquemático módulo USB DVB-T basado en el chip RTL2832u.....	30
40.	Módulo USB DVB-T basado en el chip RTL2832u.....	30
41.	Raspberry pi 3B+	32
42.	Interfaz GQRX Ubuntu.....	33
43.	Interfaz SDR#	34
44.	Interfaz RPITX Raspberry pi	35
45.	Interfaz GNU Radio.....	36
46.	Bloque fuente RTL-SDR.....	37
47.	Variable.....	37
48.	<i>Audio source</i>	38
49.	<i>Rational resampler</i>	38
50.	<i>TCP sink</i>	39
51.	<i>Audio sink</i>	40

52.	WX GUI FFT <i>sink</i>	40
53.	Radio enlace	41
54.	Patrón de radiación direccional 2 dimensiones	44
55.	Patrón de radiación direccional 3 dimensiones	45
56.	Patrón de radiación antena omnidireccional 2 dimensiones	46
57.	Patrón de radiación antena omnidireccional 3 dimensiones	46
58.	Tipos de polarización.....	49
59.	Dipolo	50
60.	Patrón de radiación dipolo	51
61.	Antena Yagi	52
62.	Arreglo de antenas planas circular	53
63.	Antena tipo bocina	54
64.	Antena parabólica.....	54
65.	Zonas de Fresnell	56
66.	GPIO pinout Raspberry pi 3.....	57
67.	Diagrama de bloques transmisor NBFM con GNURadio.....	58
68.	Diagrama de bloques receptor de señales de radio frecuencias con GNU radio	59
69.	Instalacion librería <i>git</i>	62
70.	Instalación <i>CMake</i>	62
71.	Instalacion <i>libusb</i>	63
72.	Instalación <i>build-essential</i>	64
73.	<i>Gitclone</i>	64
74.	<i>Mkdir build</i>	65
75.	<i>CMAKE</i>	66
76.	<i>Make</i>	66
77.	<i>Make Install</i>	67
78.	<i>Idconfig</i>	67
79.	<i>Rtl-sdr.rules</i>	68

80.	Directorio <i>modprobe.d</i>	69
81.	<i>Blacklist-rtl.conf</i>	69
82.	Contenido <i>Blacklist-rtl.conf</i>	70
83.	Instalación GNURADIO	70
84.	Instalación GQRX	71
85.	Configuración GQRX	71
86.	Prueba GQRX	72
87.	<i>Gitclone</i> RPITX	73
88.	Instalación RPITX	73
89.	<i>Easytest</i> RPITX 1.....	74
90.	<i>Easytest</i> RPITX 2.....	74
91.	<i>Easytest</i> RPITX 3.....	75
92.	Diagrama de bloques transmisor amplitud modulada	76
93.	Parámetros <i>wav file source</i> transmisor AM.....	77
94.	Parámetros <i>float to complex</i> transmisor AM	78
95.	Parámetros <i>Add constant</i> transmisor AM.....	78
96.	Parámetros <i>signal source</i> transmisor AM.....	79
97.	Parámetros <i>trottle</i> transmisor AM	80
98.	Parámetros <i>multiply</i> transmisor AM.....	80
99.	Parámetros WX GUI FFT <i>sink</i> transmisor AM	81
100.	Parámetros TCP <i>sink</i> transmisor AM	82
101.	Parámetros <i>variable</i> transmisor AM.....	82
102.	Diagrama de bloques receptor modulación en amplitud	83
103.	Parámetros <i>variable 1</i> receptor AM	84
104.	Parámetros <i>variable 2</i> receptor AM	84
105.	Parámetros RTL-SDR <i>source</i> receptor AM.....	85
106.	Parámetros AM <i>demod</i> receptor AM	86
107.	Parámetros <i>rational resampler</i> receptor AM	86
108.	Parámetros WX GUI FFT <i>sink</i> receptor AM.....	87

109.	Parámetros <i>audio sink</i> receptor AM	88
110.	Diagrama de bloques transmisor WBFM	89
111.	Parámetros <i>wav file source</i> transmisor WBFM.....	90
112.	Parámetros <i>trottle</i> transmisor WBFM	90
113.	Parámetros AGC2 transmisor WBFM	91
114.	Parámetros <i>band pass filter</i> transmisor WBFM	92
115.	Parámetros WBFM <i>transmit</i> transmisor WBFM.....	92
116.	Parámetros TCP <i>sink</i> transmisor WBFM	93
117.	Parámetros WX GUI FFT <i>sink</i> transmisor WBFM	94
118.	Parámetro variable transmisor WBFM	95
119.	Diagrama de bloques transmisor NBFM	95
120.	Parámetros <i>wav file source</i> transmisor NBFM	96
121.	Parámetros <i>trottle</i> transmisor NBFM.....	97
122.	Parámetros AGC2 transmisor NBFM	97
123.	Parámetros <i>band pass filter</i> transmisor NBFM.....	98
124.	Parámetros NBFM <i>transmit</i> transmisor NBFM	99
125.	Parámetros TCP <i>sink</i> transmisor NBFM.....	99
126.	Parámetros WX GUI FFT <i>sink</i> transmisor NBFM.....	100
127.	Parámetros variable transmisor NBFM	101
128.	Diagrama de bloques receptor FM	101
129.	Parámetros variable 1 receptor FM	102
130.	Parámetros variable 2 receptor FM	103
131.	Parámetros RTL-SDR <i>source</i> receptor AM	103
132.	Parámetros <i>low pass filter</i> receptor FM.....	104
133.	Parámetros WBFM <i>demod</i> receptor FM	105
134.	Parámetros <i>rational resampler</i> receptor FM	105
135.	Parámetros <i>audio sink</i> receptor FM	106
136.	Parámetros WX GUI FFT <i>sink</i> receptor FM.....	107
137.	Diagrama de bloques modulador BPSK	108

138.	Parámetros <i>signal source 1</i> modulador BPSK.....	109
139.	Parámetros <i>signal source 2</i> modulador BPSK.....	109
140.	Parámetros <i>multiply constant</i> modulador BPSK	110
141.	Parámetros <i>add constant</i> modulador BPSK.....	110
142.	Parámetros <i>multiply</i> modulador BPSK	111
143.	Parámetros <i>variable</i> modulador BPSK.....	111
144.	Parámetros WX GUI <i>scope sink</i> modulador BPSK	112
145.	Diagrama de bloques modulador QPSK	113
146.	Parámetros <i>vector source</i> modulador QPSK	114
147.	Parámetros <i>repack bits</i> modulador QPSK	115
148.	Parámetros <i>PSK mod</i> modulador QPSK.....	115
149.	Parámetros <i>multiply</i> modulador QPSK.....	116
150.	Parámetros <i>signal source</i> modulador QPSK.....	116
151.	Parámetros <i>variable</i> modulador QPSK	117
152.	Parámetros WX GUI <i>constellation sink</i> modulador QPSK.....	118
153.	Diagrama de bloques transmisor ASK.....	119
154.	Parámetros <i>signal source 1</i> transmisor ASK.....	120
155.	Parámetros <i>signal source 2</i> transmisor ASK.....	120
156.	Parámetros <i>signal source 3</i> transmisor ASK.....	121
157.	Parámetros <i>trottle</i> transmisor ASK	121
158.	Parámetros <i>multiply</i> transmisor ASK.....	122
159.	Parámetros <i>add constant</i> transmisor ASK	123
160.	Parámetros WX GUI FFT <i>sink</i> transmisor ASK	124
161.	Parámetros TCP <i>sink</i> transmisor ASK.....	125
162.	Parámetros <i>variable</i> transmisor ASK.....	125
163.	Diagrama de bloques modulador FSK.....	126
164.	Parámetros <i>signal source 1</i> modulador FSK.....	127
165.	Parámetros <i>signal source 2</i> modulador FSK.....	128
166.	Parámetros <i>signal source 3</i> modulador FSK.....	128

167.	Parámetros <i>throttle</i> modulador FSK.....	129
168.	Parámetros <i>not</i> modulador FSK.....	129
169.	Parámetros <i>short to float</i> modulador FSK.....	130
170.	Parámetros <i>multiply</i> modulador FSK.....	130
171.	Parámetros <i>add</i> modulador FSK.....	131
172.	Parámetros variable modulador FSK.....	131
173.	Parámetros WX GUI <i>scope sink</i> modulador FSK.....	132
174.	Diagrama de bloques modulador QAM.....	133
175.	Parámetros <i>vector source</i> modulador QAM.....	134
176.	Parámetros <i>repack bits</i> modulador QAM.....	135
177.	Parámetros QAM <i>mod</i> modulador QAM.....	135
178.	Parámetros <i>multiply</i> modulador QAM.....	136
179.	Parámetros <i>signal source</i> modulador QAM.....	136
180.	Parámetros variable modulador QAM.....	137
181.	Parámetros WX GUI <i>constellation sink</i> modulador QAM.....	138

TABLAS

I.	Ganancias y pérdidas en un radio enlace.....	42
II.	División del espectro electromagnético.....	42
III.	Rango de operación antenas.....	49
IV.	Parámetros dipolo elemental.....	51
V.	Comparación de antenas.....	55

LISTA DE SÍMBOLOS

Símbolo	Significado
ARM	<i>Advance RISC machine</i>
AM	Amplitud modulada
ASK	<i>Amplitude shift key</i> (modulación por desplazamiento de amplitud)
BPSK	<i>Binary phase shift key</i>
Δ	Delta
DSB	<i>Double side band</i> (doble banda lateral)
f	Frecuencia
FM	Frecuencia modulada
GPIO	<i>General purpose in/out pin</i>
=	Igualdad
m	Índice de modulación
kHz	Kilohertz
MHz	Megahertz
*	Multiplicación
PM	<i>Phase modulation</i> (modulación en fase)
PSK	<i>Phase shift key</i> (modulación por desplazamiento de fase)
PCM	<i>Pulse code modulation</i> (modulación por código de pulso)
QAM	<i>Quadrature amplitude modulation</i> (modulación de amplitud en cuadratura)

QPSK	<i>Quadrature phase shift key</i> (modulación por desplazamiento de fase en cuadratura)
SSB	<i>Simple side band</i> (banda lateral única)
+	Suma
TCP	<i>Transfer client protocol</i>
USB	<i>Universal serial bus</i>

GLOSARIO

Binario	Que está compuesto de 2 elementos.
<i>Bit</i>	En informática y otras disciplinas, unidad mínima de información que puede tener solo dos valores: 1 o 0.
Compilar	Traducir con un compilador un programa en lenguaje de alto nivel a lenguaje de máquina.
Constante	Que tiene un valor fijo en un cálculo o proceso.
Desfase	Diferencia de fase entre dos fenómenos periódicos de la misma frecuencia.
<i>Driver</i>	Controlador.
Ecuación	Igualdad entre 2 expresiones.
Enlace	Elemento que permite acceder automáticamente a otro documento o parte de este.
Fase	Indica la situación instantánea en el ciclo.
Frecuencia	Número de vibraciones, ondas o ciclos de un fenómeno periódico, realizado en una unidad de tiempo definida.

Hardware	Conjunto de elementos físicos o materiales que comprenden un sistema informático.
Librería	Conjunto de implementaciones funcionales, codificadas en un lenguaje de programación, que ofrece una interfaz bien definida para la funcionalidad que se invoca.
Onda	Conjunto de partículas que, en la propagación del movimiento vibratorio dentro de un medio o cuerpo elástico; se encuentran en fases distintas intermedias entre dos fases iguales.
Onda electromagnética	Onda producida por cargas eléctricas en movimiento.
Oscilador	Aparato para producir corrientes oscilatorias, especialmente el que se usa en radiotelegrafía y radiotelefonía.
Radiación	Emisión de energía o de partículas que producen algunos cuerpos y que se propagan a través del espacio.
Recepción	Acción de recibir.
Señal	Onda electromagnética que permite transmitir información a un circuito electrónico.

Sistema	Conjunto de elementos o partes coordinadas que responden a una ley, o que, ordenadamente relacionadas entre sí, contribuyen a determinado objeto o función.
Software	Conjunto de programas y rutinas que permiten a la computadora realizar determinadas tareas.
Telecomunicaciones	Sistema de comunicación a distancia que se realiza por medios eléctricos o electromagnéticos.
Transmisión	Acción de transmitir.
Variable	Símbolo que representa el conjunto de valores que puede tomar una determinada magnitud.

RESUMEN

El trabajo de graduación que a continuación se presenta consta de 4 capítulos. En el primero se da a conocer la teoría básica de comunicaciones, comprendida por los principales esquemas de modulación, entre ellos los esquemas de modulación digitales y analógicos.

En el segundo capítulo se presentan las herramientas de software y hardware necesarias para la implementación de la tecnología radio definida por software, utilizando un módulo USB basado en el chip RTL 2832u y una Raspberry pi versión 3 b+. En el tercero se describe la teoría necesaria para trabajar con sistemas de telecomunicaciones, los distintos tipos de antenas, así como la implementación de un radio enlace a nivel de laboratorio, utilizando la tecnología de radio definida por software.

Por último, en el cuarto capítulo, se presenta una serie de prácticas aplicables al laboratorio de Comunicaciones 1, en las cuales se busca la demostración de los distintos esquemas de modulación presentados en el capítulo 1, utilizando la tecnología radio definida por software.

OBJETIVOS

General

Implementar la tecnología de radio definida por software, utilizando el módulo basado en el chip Realtek RTL2832u y una Raspberry pi versión 3, para su utilización en sistemas de telecomunicaciones.

Específicos

1. Presentar los fundamentos de la modulación de señales en sistemas de telecomunicaciones electrónicas.
2. Presentar la tecnología radio definida por software y las herramientas necesarias para su implementación.
3. Presentar un enlace en sistemas de telecomunicaciones, utilizando el módulo RTL 2832u y Raspberry pi 3.
4. Documentar y proponer prácticas aplicables al laboratorio del curso de Comunicaciones 1.

INTRODUCCIÓN

En la carrera de Ingeniería Electrónica de la Facultad de Ingeniería de la Universidad de San Carlos, se tiene una serie de cursos dedicados al área de comunicaciones electrónicas, en los cuales establece que la tecnología de radio definida por software es muy útil como herramienta de laboratorio; la poca disponibilidad de recursos y fuentes de información, así como la barrera del idioma, ya que el material disponible se encuentra en inglés, representan un gran obstáculo para que el estudiante pueda llevar a cabo la demostración de los conceptos aprendidos en el curso, como los distintos esquemas de modulación, aplicando la tecnología de radio definida por software.

La tecnología de radio definida por software permite trabajar los sistemas de radiocomunicación para la transmisión de información, aplicando la menor cantidad de hardware, implementando la mayoría de los componentes y utilizando herramientas de software.

Teniendo en cuenta esto, se presenta una serie de prácticas, implementando dicha tecnología en la que se aplican los conocimientos del curso para mejorar y agilizar el proceso de aprendizaje, realizando la demostración de la teoría.

1. FUNDAMENTOS DE LA MODULACIÓN DE SEÑALES EN SISTEMAS DE TELECOMUNICACIONES ELECTRÓNICAS

A continuación, se define en qué consiste la modulación, así como los esquemas más utilizados y sus fundamentos básicos.

1.1. Modulación

Consiste en una serie de procesos con los cuales se busca modificar o alterar parámetros de una onda portadora, siendo los parámetros a modificar: la amplitud, frecuencia y fase. Modular una onda permite aprovechar el medio de transmisión, haciendo posible enviar información de un punto a otro, utilizando una onda electromagnética de alta frecuencia, buscando aprovechar el canal de transmisión, al mismo tiempo que se busca eliminar la interferencia que pueda generar una onda sobre otra y el ruido generado en el medio de transmisión. Cuando se hace referencia a modulación deben considerarse 3 ondas:

- Moduladora
- Portadora
- Modulada

La onda moduladora es el mensaje o información que se desea procesar en el sistema, con el objetivo de ser enviada de un punto A hacia un punto B, usualmente utilizando una onda portadora; siendo esta una onda electromagnética de alta frecuencia para su transmisión.

La onda portadora, normalmente, es una onda sinusoidal; esta será modificada, ya sea en amplitud, en frecuencia o ambas, dependiendo del esquema de modulación que se desee aplicar; dichas modificaciones serán en función a la onda moduladora y darán como resultado una onda modulada. La onda portadora debe ser de una frecuencia mucho mayor a la moduladora.

Una onda modulada es la portadora, pero alterada en función de la onda moduladora; en sistemas de comunicaciones se puede encontrar la modulación en amplitud y modulación angular; se puede trabajar una combinación de ambas, dependiendo de la aplicación o necesidades del sistema de transmisión.

En sistemas de telecomunicaciones, es posible trabajar esquemas de modulación analógicos y digitales.

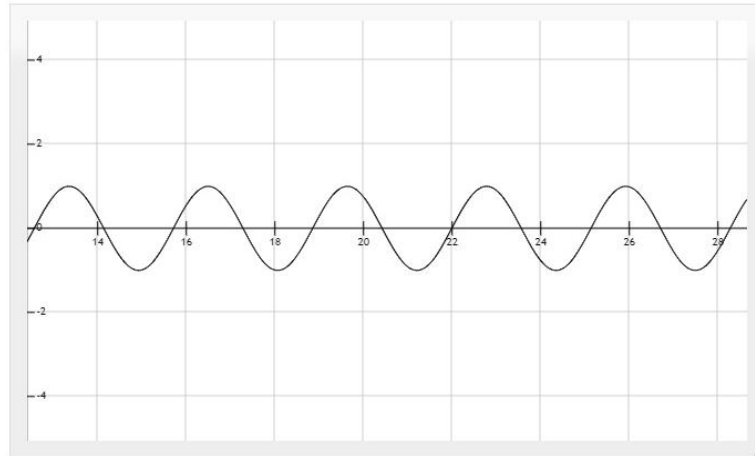
1.2. Modulación analógica

La modulación analógica consiste en procesar una onda continua en el tiempo como onda moduladora, empleando tanto la modulación en amplitud como la angular.

1.2.1. Modulación en amplitud

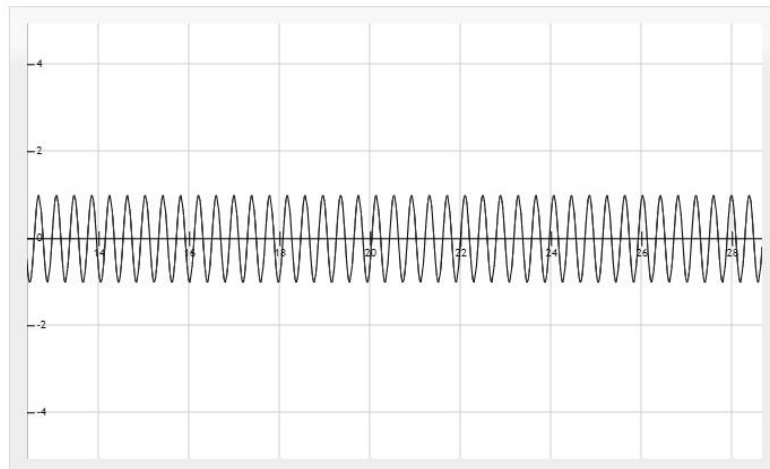
La amplitud modulada o AM, es uno de los esquemas de modulación más sencillos que existen; haciendo referencia al proceso de modulación y demodulación de la onda, se tiene que ambas etapas pueden trabajarse con circuitos muy simples, por ende, económicos. En la modulación por amplitud, se deben tomar en cuenta las 3 ondas mencionadas: onda moduladora, visualizada gráficamente en la figura 1, onda portadora, visualizada gráficamente en la figura 2 y la onda modulada, visualizada gráficamente en la figura 3.

Figura 1. **Onda moduladora AM**



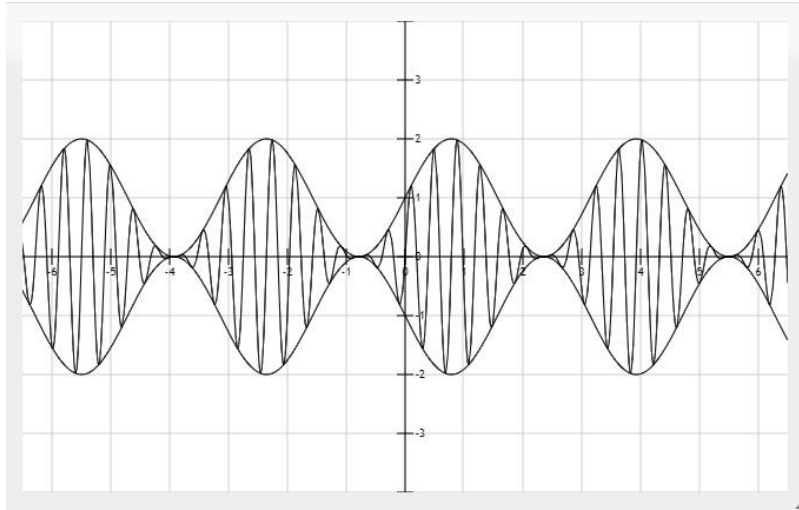
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 2. **Onda portadora AM**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>

Figura 3. **Onda modulada AM**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Matemáticamente se puede expresar la onda portadora como:

$$p(t) = A_p \text{Sen}(w_p t)$$

Y la onda moduladora puede ser representada como:

$$x(t) = A_x \text{Sen}(w_x t)$$

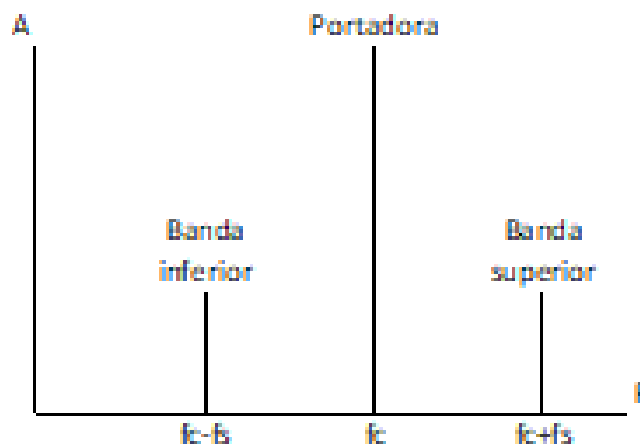
Para obtener como resultante la ecuación que representa a una señal modulada en amplitud, se expresa matemáticamente de la siguiente manera:

$$y(t) = A_p(1 + x(t))\text{Sen}(w_p t)$$

$$y(t) = A_p(1 + A_x \text{Sen}(w_x t)\text{Sen}(w_p t))$$

Al observar una onda modulada en amplitud en el dominio de la frecuencia, se puede advertir la distribución de las potencias y cómo es que aparecen dos bandas laterales, una superior y una inferior, conteniendo la información. Se observa también que la onda portadora disipa una gran cantidad de potencia, gracias a esto surgen dos esquemas: modulación en amplitud de banda lateral única con portadora suprimida o SSB SC y modulación en amplitud de doble banda lateral con portadora suprimida.

Figura 4. **Amplitud modulada en el dominio de la frecuencia**



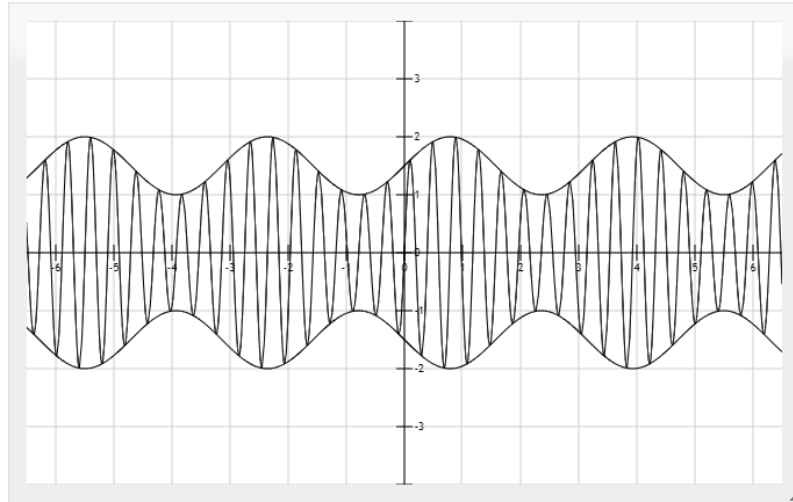
Fuente: elaboración propia, empleando Paint.

1.2.1.1. Índice de modulación

El índice de modulación en AM, es un parámetro que indica el porcentaje o grado en el que la señal moduladora afecta a la amplitud de la onda portadora. Matemáticamente, el índice de modulación en amplitud puede ser calculado con la siguiente relación:

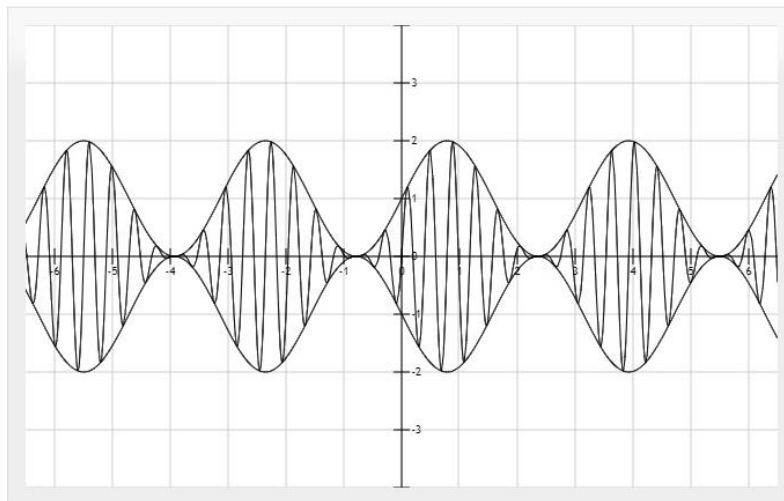
$$m = \frac{A_x}{A_p}$$

Figura 5. **50 % de índice de modulación AM**



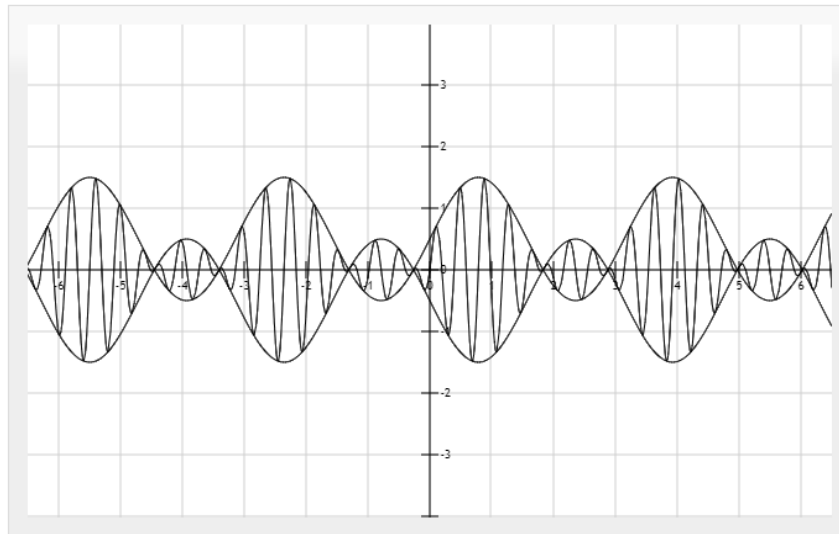
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 6. **100 % de índice de modulación AM**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 7. **150 % de índice de modulación AM**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

1.2.1.2. **Modulación banda lateral única SSB SC**

La modulación de banda lateral única con portadora suprimida o *simple side band supressed carrier*, por sus siglas en inglés SSB SC; al trabajar la transmisión de una onda modulada en amplitud se debe considerar la potencia necesaria para su transmisión; al evaluar la potencia en porcentaje, se tiene que el 50 % de la potencia total transmitida se encuentra en una onda senoidal con frecuencia constante, siendo esta la portadora; 25 % de la potencia es utilizada para transmitir la información que se desea enviar, conocida como onda moduladora; dicha onda es enviada dos veces, la primera en una banda de frecuencias ligeramente menor a la frecuencia de la onda portadora y la segunda, en una banda de frecuencias ligeramente mayor a la frecuencia de la onda portadora.

Por medio de una serie de filtros previos a la transmisión, un transmisor SSB puede eliminar una de las bandas, así como la portadora. Al trabajar la recepción y demodulación de una señal en SSB es necesario contar con un oscilador que opere a la frecuencia de la onda portadora, para ser agregada a la señal y reconstruir el mensaje a partir de una de las bandas recibidas.

1.2.1.3. Modulación doble banda lateral DSB SC

La modulación de banda lateral doble con portadora suprimida o *double side band suppressed carrier*, por sus siglas en inglés DSB SC, es similar al funcionamiento de la modulación de banda lateral simple, con la diferencia de que un transmisor DSB SC se encargará de eliminar solamente la portadora previamente a la transmisión, enviando solamente la onda modulada sin la onda portadora; de tal manera que se consigue optimizar la potencia del transmisor sin alterar la información. En la recepción de DSB SC, al igual en SSB SC, se debe trabajar con un oscilador, ya que la onda portadora, necesaria para la demodulación, no fue transmitida.

1.2.2. Modulación angular

Para mejorar la susceptibilidad al ruido de los sistemas de comunicaciones, se buscó procesar la información de tal manera que la señal modulada, al ser transmitida por medio de radio frecuencia, fuera menos afectada por condiciones ajenas al sistema; teniendo que al mantener una amplitud constante y variando la frecuencia o la fase de la onda portadora, se puede obtener una onda modulada prácticamente inmune al ruido, debido a que en la transmisión es muy difícil poder variar estos parámetros, mientras que la amplitud sí puede variar con facilidad.

1.2.2.1. Modulación de fase

Modulación de fase o *phase modulation*, también conocida como PM, consiste en la variación de fase de la onda portadora, variando de manera directamente proporcional a la onda moduladora. Este esquema de modulación es muy poco utilizado debido a la complejidad que representa la modulación y demodulación de la misma. Una onda modulada en fase puede ser representada matemáticamente por la siguiente ecuación:

$$y(t) = A_p \text{Sen}(w_p t + w_i t)$$

Donde:

$y(t)$ = onda modulada

A_p = amplitud onda portadora

$w_i t = N_p * x(t)$

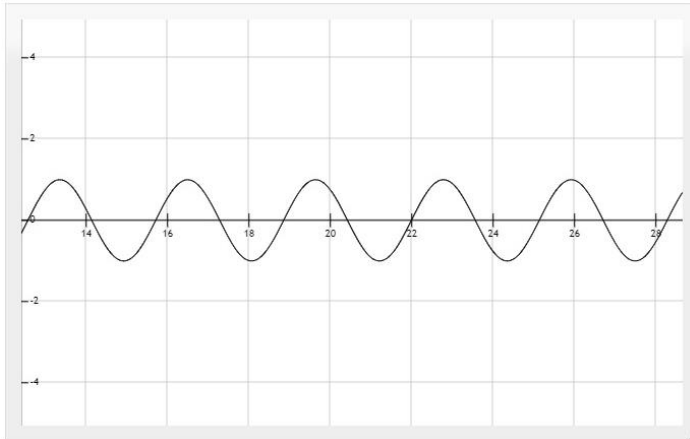
$x(t)$ = onda moduladora

N_p = índice de modulación de fase

1.2.2.1.1. Modulación de frecuencia

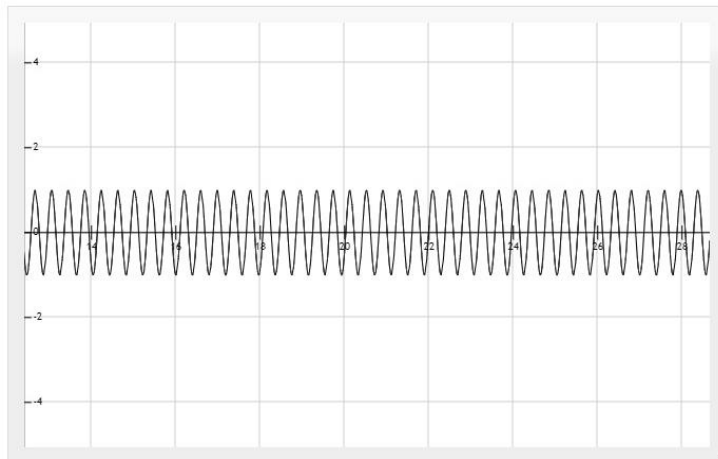
La modulación en frecuencia o *frequency modulation*, modulación angular, consiste en variar la frecuencia de la onda portadora, proporcionalmente a la amplitud de la onda moduladora. Su principal característica es una teórica inmunidad al ruido, ya que, al ser demodulada una onda modulada en frecuencia, se toman las variaciones en la frecuencia y no la amplitud de la onda, la cual es sumamente afectada por condiciones externas al sistema. La modulación en frecuencia se puede trabajar tanto en modulación en frecuencia de banda ancha, también conocida como WBFM, como en frecuencia de banda angosta, conocida como NBFM.

Figura 8. **Onda moduladora FM**



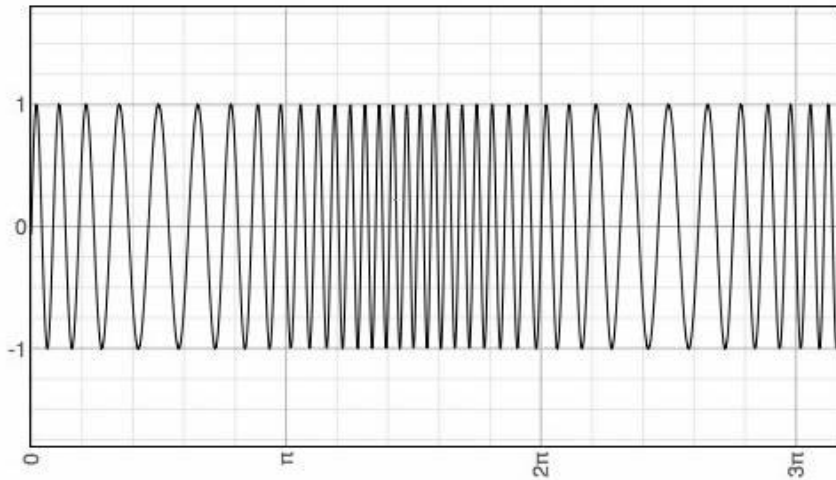
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 9. **Onda portadora FM**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 10. **Onda modulada FM**



Fuente: Textos científicos. *Modulación*.

<https://www.textoscientificos.com/redes/modulacion/frecuencia>. Consulta: mayo de 2020.

1.2.2.1.2. **Índice de modulación FM**

El índice de modulación FM depende de dos factores: la variación en frecuencia y la frecuencia de la onda portadora, siendo representado matemáticamente como:

$$m = \frac{\Delta f}{f}$$

1.3. **Modulación digital**

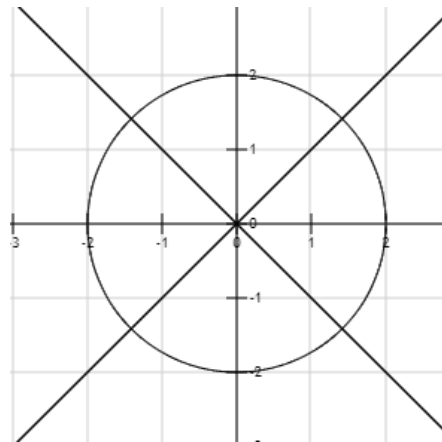
Consiste en procesar una onda discreta en el tiempo como onda moduladora, al igual que en la modulación analógica; se tiene tanto la modulación en amplitud como la modulación angular o una combinación de ambas, como es

el caso de QAM. A continuación, se presenta una breve descripción de los esquemas de modulación digitales más utilizados en la actualidad.

1.3.1. Modulación por desplazamiento de fase PSK

Phase shift key o modulación por desplazamiento de fase, es el esquema de modulación digital que consiste en la variación de la fase de la onda portadora; la modulación por desplazamiento de fase puede ser trabajada en dos casos distintos: BPSK y QPSK. La modulación por desplazamiento de fase se representa en un diagrama de constelaciones, que puede ser visualizado en la figura 11, en el cual se grafica la amplitud y la fase de una onda.

Figura 11. Diagrama de constelaciones

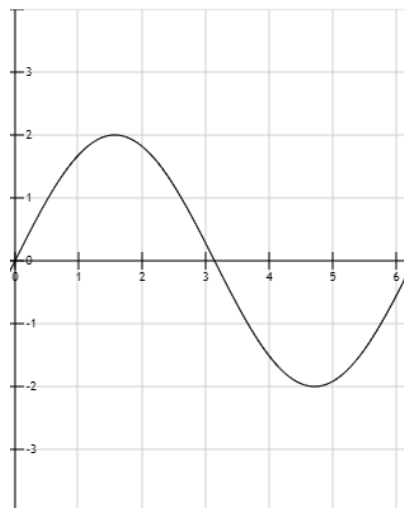


Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

1.3.1.1. Modulación por desplazamiento de fase binaria BPSK

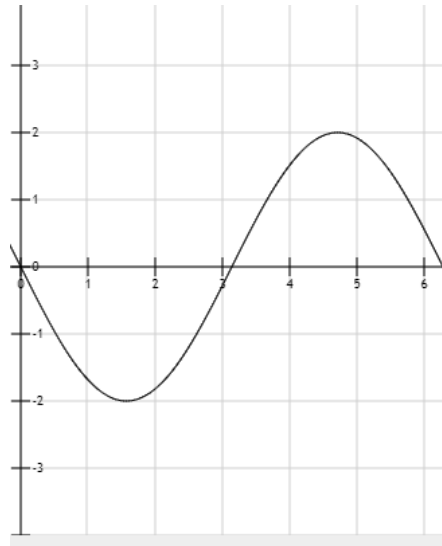
Binary phase shift key o modulación por desplazamiento de fase binaria, se basa en la variación de la fase de la onda portadora, pudiendo tomar 2 valores posibles, representados en número binario como 1 y 0. Los desfases trabajados en BPSK son 0° y 180° .

Figura 12. Onda portadora desfasada 0°



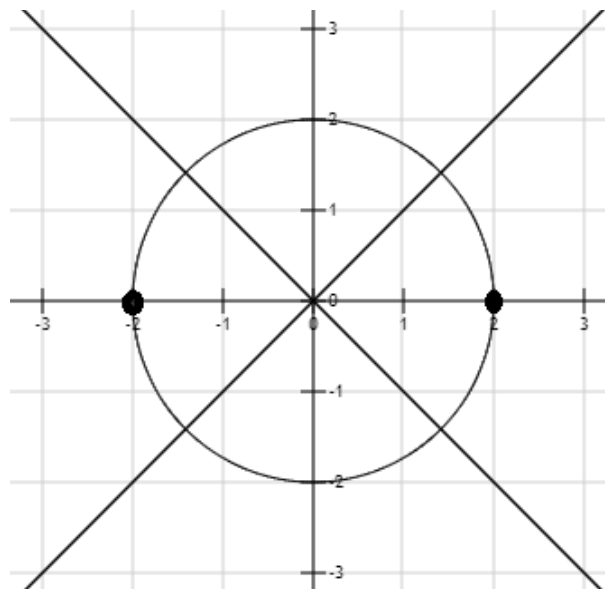
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 13. **Onda portadora desfasada 180°**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 14. **Diagrama de constelaciones BPSK 0° y 180°**

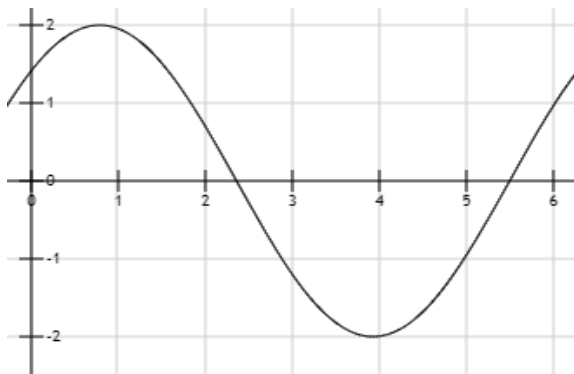


Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

1.3.1.2. Modulación por desplazamiento de fase en cuadratura QPSK

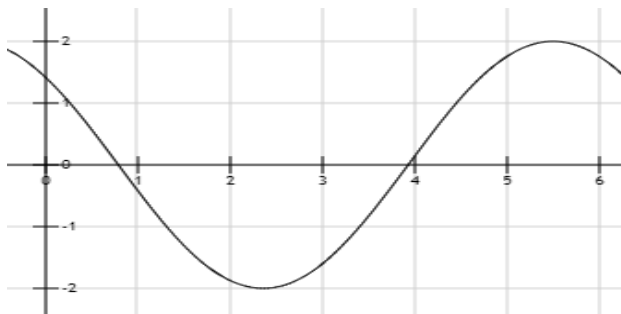
Quadrature phase shift key o modulación por desplazamiento de fase en cuadratura; este esquema se basa en la variación de la fase de la onda portadora, pudiendo tomar 4 valores posibles representados en un código binario de 2 dígitos: “00”, “01”, “10”, “11”.

Figura 15. Onda portadora desfasada 45°



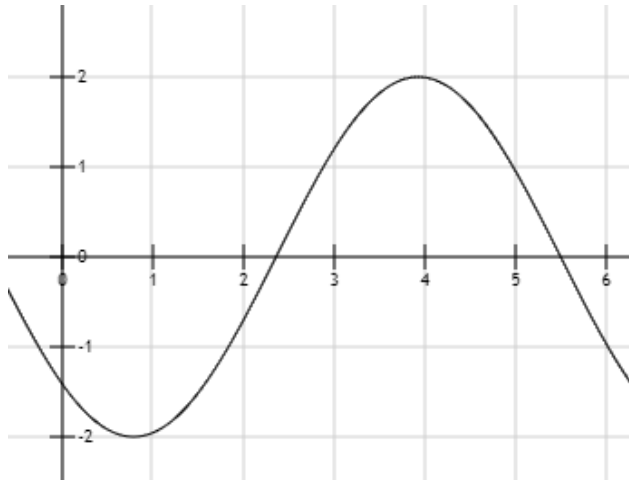
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 16. Onda portadora desfasada 135°



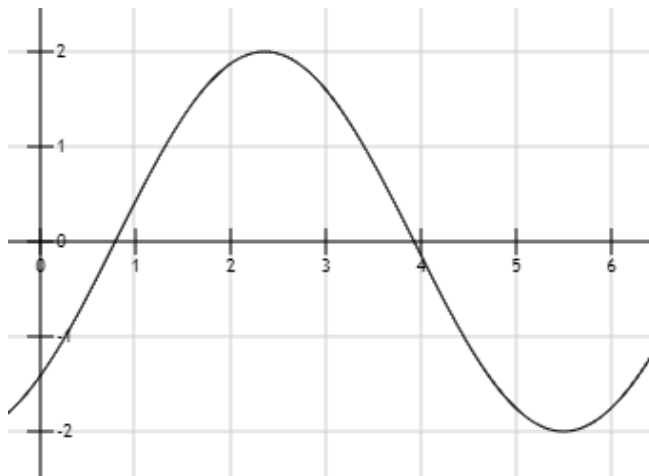
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 17. **Onda portadora desfasada 225°**



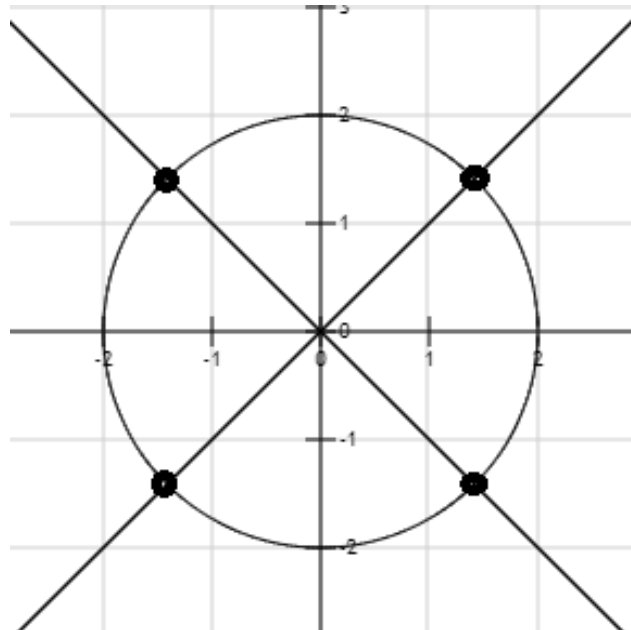
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 18. **Onda portadora desfasada 315°**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 19. **Diagrama de constelaciones BPSK 45° 135° 225° 315°**

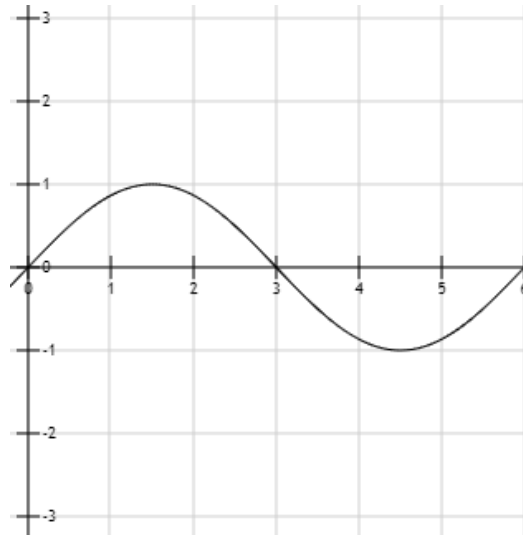


Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

1.3.2. **Modulación por desplazamiento de frecuencia FSK**

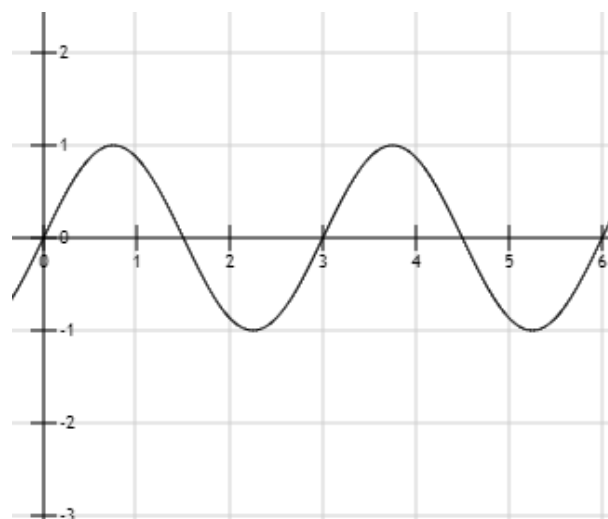
Frequency shift key o modulación por desplazamiento de frecuencia, consiste en la variación de la frecuencia de la onda portadora, tomando estos valores específicos en frecuencia, con los cuales se busca representar un símbolo (1 o 0), permitiendo de esta manera enviar en código binario.

Figura 20. **Ejemplo de gráfica representando un 1 lógico en FSK**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 21. **Ejemplo de gráfica representando un 0 lógico en FSK**

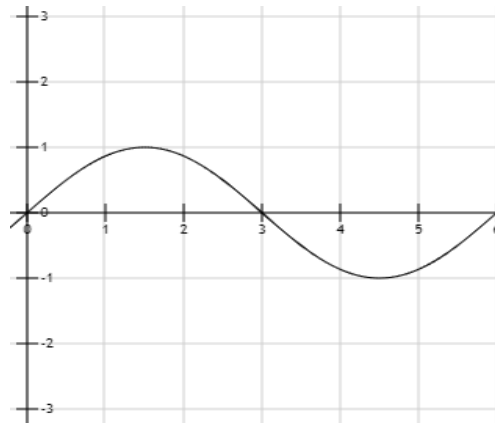


Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

1.3.3. Modulación por desplazamiento de amplitud ASK

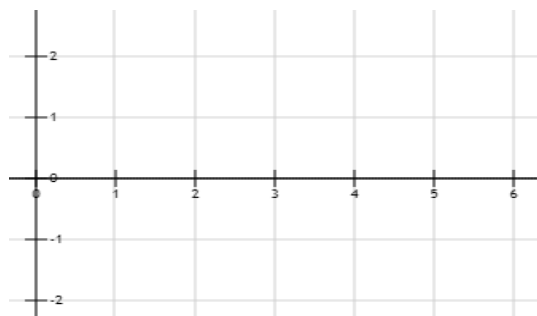
Amplitude shift key o modulación por desplazamiento de amplitud, similar a FSK, busca representar un símbolo (1 o 0), variando la amplitud de la onda moduladora, tomando un valor de 0 para representar un 0 lógico o la amplitud máxima de la onda portadora para representar un 1 lógico.

Figura 22. **Ejemplo de gráfica representando un 1 lógico en ASK**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 23. **Ejemplo de gráfica representando un 0 lógico en ASK**



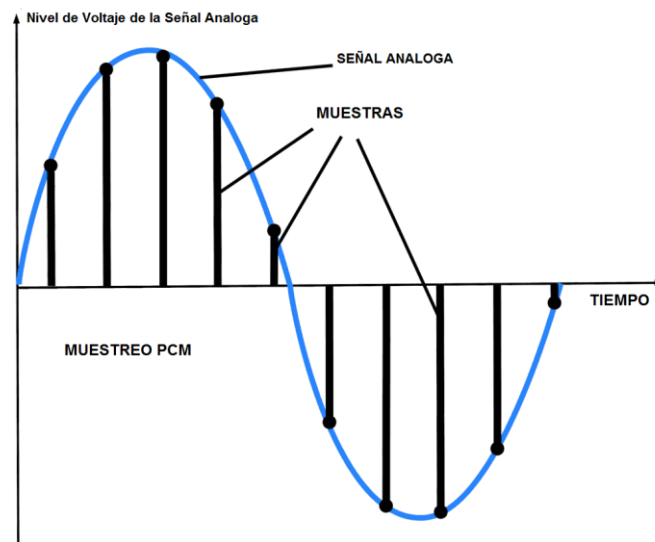
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

1.3.4. Modulación por pulso codificado PCM

Pulse code modulation o modulación por pulso codificado, utilizado como base en el conversor analógico-digital, representando valores de amplitud de una señal analógica como una secuencia de bits. Una señal analógica será muestreada periódicamente, y posteriormente dichos valores serán aproximados o cuantizados al valor más cercano a una serie de valores representados por un código binario.

En la modulación PCM, la cantidad de niveles o la resolución con la que se desea trabajar la señal dependerá de la cantidad de bits por muestra que se puedan tomar, siendo posible trabajar con señales de N cantidad de bits.

Figura 24. Modulación PCM



Fuente: *Equiposaudio. Modulación PCM.* <https://equiposaudio.com/blog/sistemas-teatro-en-casa/pcm-audio/> Consulta: junio 2020.

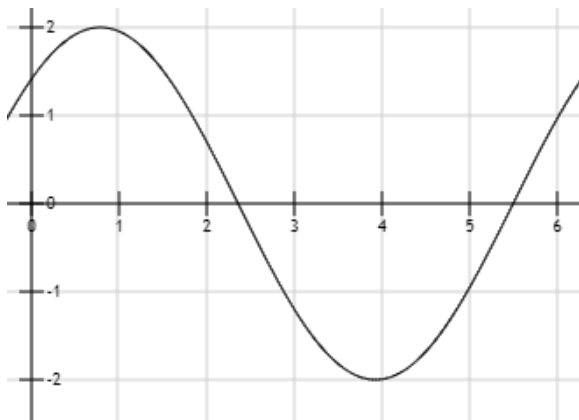
1.3.5. Modulación por amplitud en cuadratura QAM

Quadrature amplitude modulation o modulación por amplitud en cuadratura; esta trabaja una combinación entre la modulación de fase y modulación de amplitud, permitiendo así representar un código binario de N bits por medio de ondas desfasadas y una cantidad de grados específica, con variaciones de amplitud predeterminadas.

1.3.5.1. 4QAM

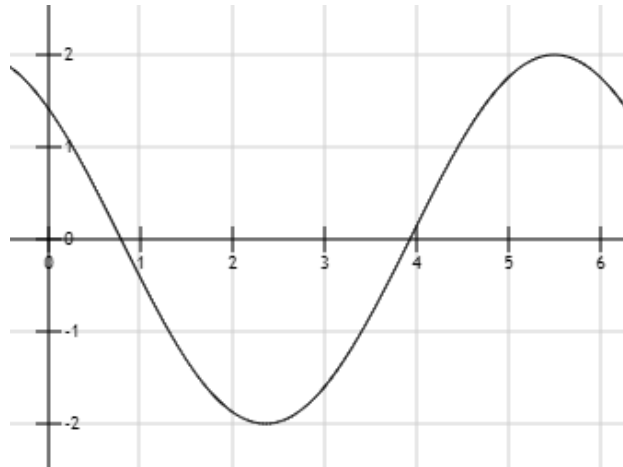
Similar a BPSQ, trabaja con un código binario de 2 bits, tomando 4 posibles valores: “00”, “01”, “10” y “11”; trabaja con desfases de 90° entre sí, partiendo de la primera pareja desfasada a 45° del origen.

Figura 25. Onda portadora desfasada 45°



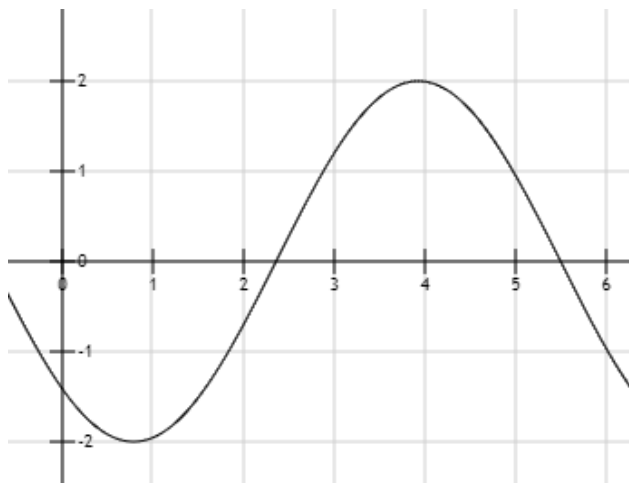
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 26. **Onda portadora desfasada 135°**



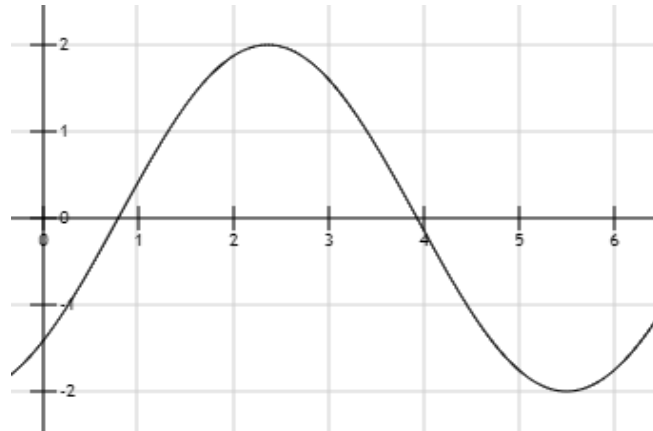
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 27. **Onda portadora desfasada 225°**



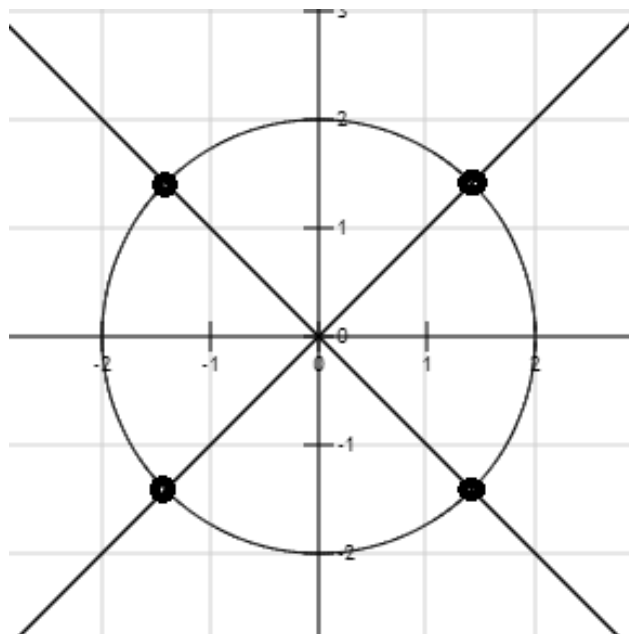
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 28. **Onda portadora desfasada 315°**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 29. **Diagrama de constelaciones QPSK 45°, 135°, 225° y 315°**

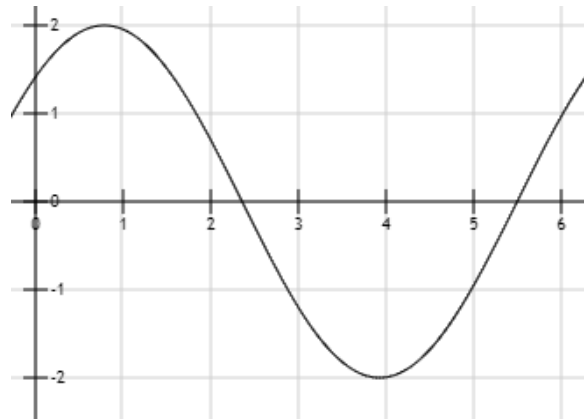


Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

1.3.5.2. 8QAM

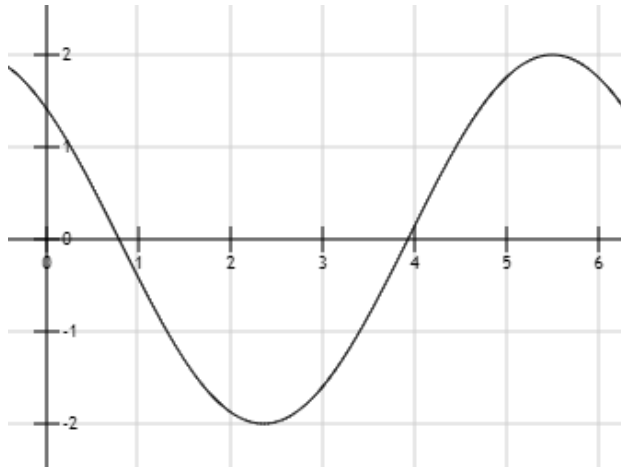
A diferencia de 4QAM, 8QAM, toma 8 valores, representados por un código binario de 3 bits, con valores de "000", "001", "010", "011", "100", "101", "110" y "111", en este caso se trabajará con desfases de 90° , partiendo de la primera onda desfasada 45° del origen, pero teniendo 2 valores de amplitud distintos, permitiendo de esta manera contar con 8 señales distintas para representar cada una de las posibles combinaciones.

Figura 30. **Onda portadora desfasada 45° con amplitud A**



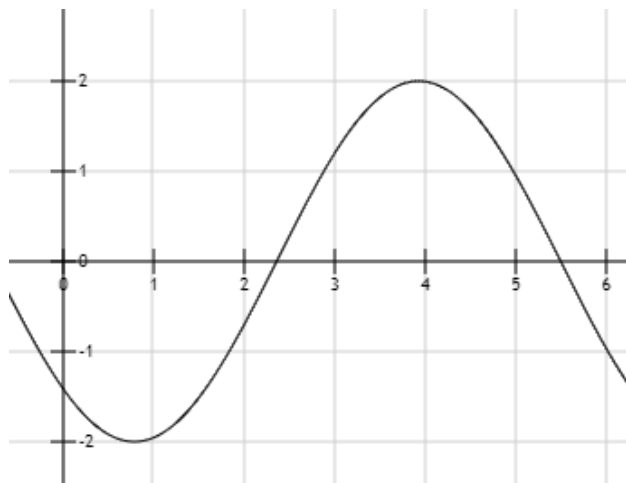
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 31. **Onda portadora desfasada 135° con amplitud A**



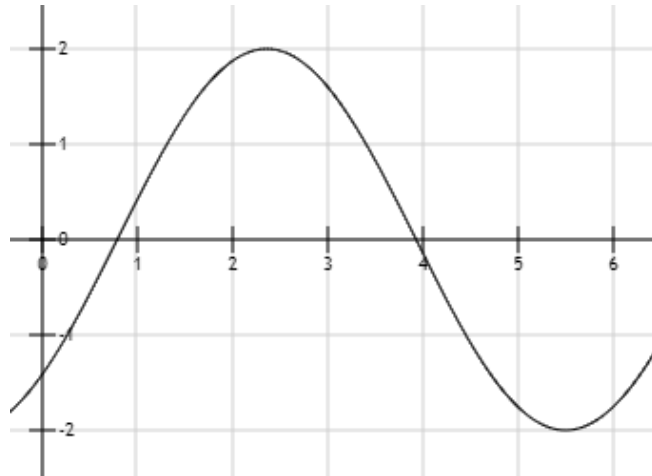
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 32. **Onda portadora desfasada 225° con amplitud A**



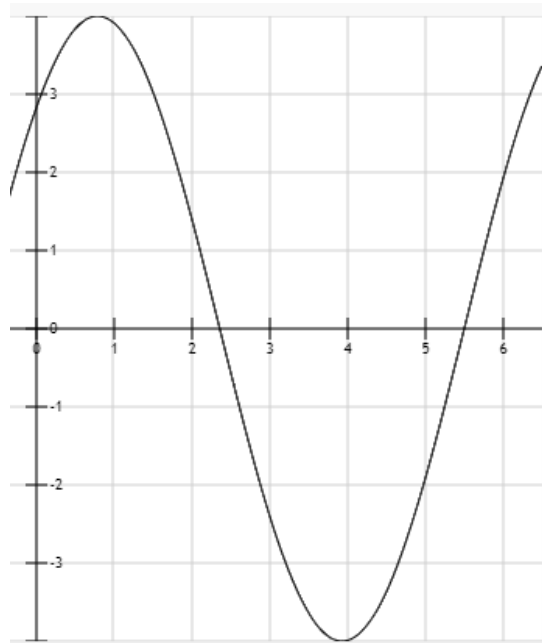
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 33. **Onda portadora desfasada 315° con amplitud A**



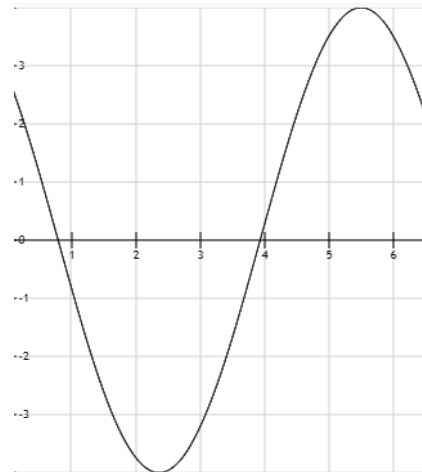
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 34. **Onda portadora desfasada 45° con amplitud $2A$**



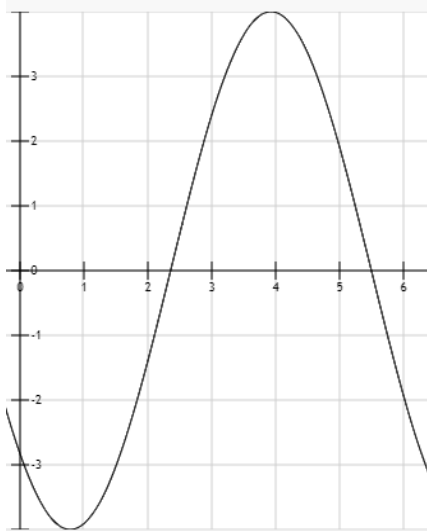
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 35. **Onda portadora desfasada 135° con amplitud $2A$**



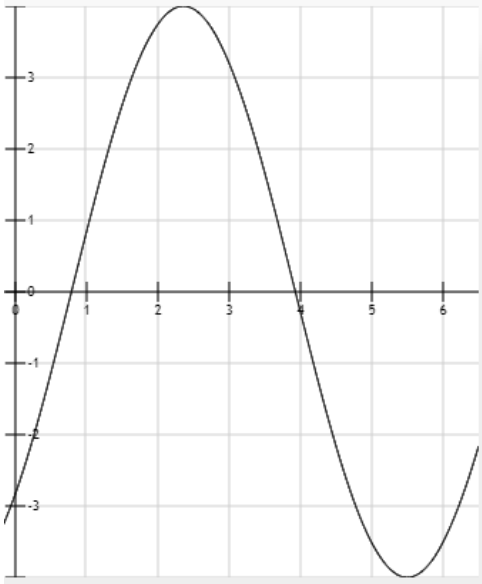
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 36. **Onda portadora desfasada 225° con amplitud $2A$**



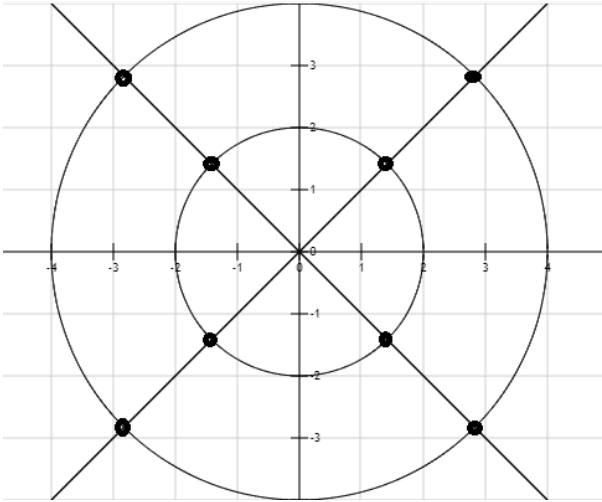
Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 37. **Onda portadora desfasada 315° con amplitud 2A**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

Figura 38. **Diagrama de constelaciones 8QAM**



Fuente: elaboración propia, empleando herramienta para graficar funciones <http://fooplot.com/>.

2. TECNOLOGÍA RADIO DEFINIDA POR SOFTWARE

La tecnología radio definida por software busca que en los sistemas de telecomunicaciones se reduzca al máximo la cantidad de dispositivos físicos, debido a que estos son, en la mayoría de los casos, más susceptibles de fallar, siendo substituidos en su mayoría por herramientas de software que permiten procesar las señales de manera digital en una computadora.

A continuación, se presentan las herramientas básicas, tanto de hardware como de software, que permiten trabajar un sistema de telecomunicaciones a nivel de laboratorio, aplicando la tecnología de radio definida por software.

2.1. Herramientas de hardware

A continuación, se presentan las herramientas de hardware o elementos físicos necesarios para trabajar la tecnología de radio definida por software.

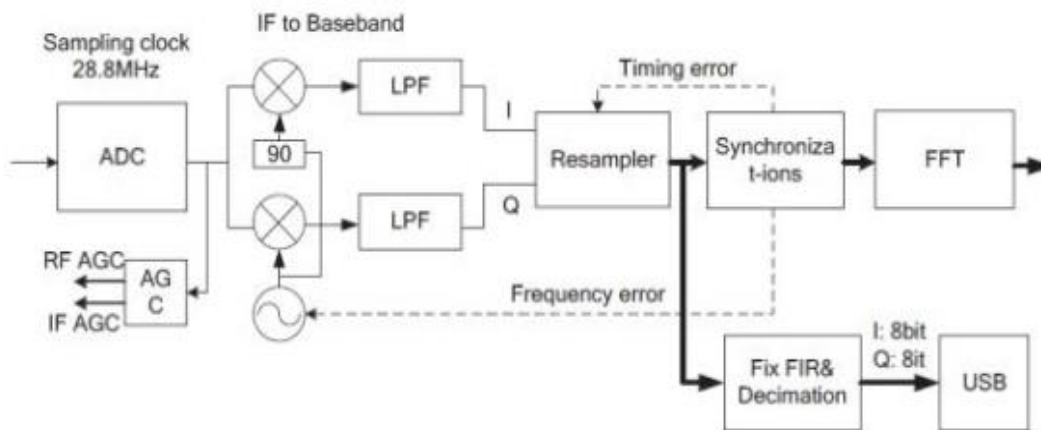
2.1.1. Módulo USB basado en el chip RTL2832u

El dispositivo USB fue diseñado originalmente para trabajar como un demodulador DVB-T COFDM; esto quiere decir que actúa como un receptor de “Difusión digital de video terrestre” (*Digital video broadcasting –terrestrial, DVB-T*) el cual usa una modulación por “Multiplexación por división de frecuencia ortogonal codificada” (*Coded Orthogonal Frequency Division Multiplexin CODFMI*).

Se descubrió que, modificando el driver original del dispositivo, es posible utilizar un módulo DVB-T COFDM, basado en el chip RTL2832U, como un

receptor de radio frecuencias, el cual tiene un ancho de banda de recepción que inicia desde los 50 Mhz hasta los 2 200 Mhz. En la figura 39 se puede observar el diagrama esquemático de un módulo receptor USB DVB-T.

Figura 39. **Diagrama esquemático módulo USB DVB-T basado en el chip RTL2832u**



Fuente: *Datasheetcafe. Esquemático modulo USB DVB-T.*

<http://www.datasheetcafe.com/rtl2832-datasheet-pdf/>. Consulta: mayo de 2020.

Figura 40. **Módulo USB DVB-T basado en el chip RTL2832u**



Fuente: *Globalsources. Modulo USB DVB-T* <https://www.globalsources.com/gsol/IDVB-T-receiver/p/sm/1151297158.htm#1151297158>. Consulta: mayo de 2020.

2.1.2. Raspberry pi versión 3

Una Raspberry pi es una computadora de tamaño reducido, basada en un procesador ARM, capaz de utilizar una gran variedad de sistemas operativos, entre los cuales se pueden encontrar:

- Raspbian, versión adaptada de Linux
- Noobs
- *Ubuntu mate*
- *Ubuntu core*
- *Ubuntu server*
- Windows 10 IoT Core
- OSMC
- LibreELEC
- Mozilla WebThings
- PiNet
- RISC OS
- *Weather Station*

2.1.2.1. Características Raspberry pi versión 3B+

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- 1GB LPDDR2 SDRAM
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- 40 pines GPIO
- Puerto HDMI
- 4 puertos USB 2.0
- Puerto CSI para cámara de Raspberry Pi

- Puerto DSI para *display touch*
- Salida de audio de 4 polos
- Puerto para micro SD
- Puerto de alimentación de 5V/2.5A DC

Figura 41. **Raspberry pi 3B+**



Fuente: *Raspberrypi. Raspberrypi 3B+* <https://www.raspberrypi.org/>. Consulta: junio de 2020.

2.2. Herramientas de software

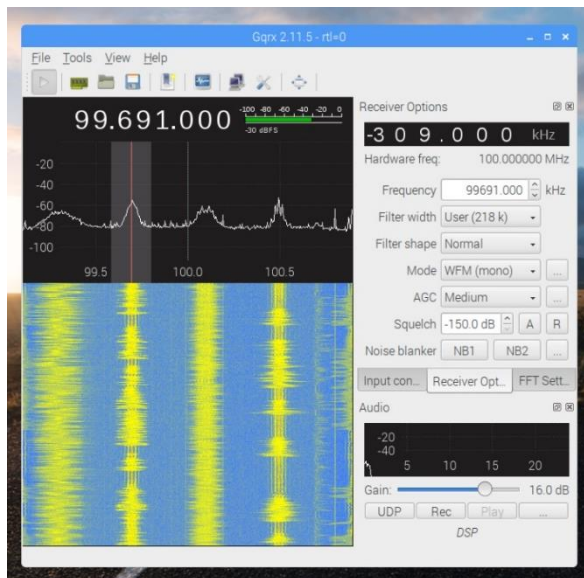
Para la implementación de la tecnología de radio definida por software, es necesario conocer algunas herramientas de software como: GQRX, GNU Radio, SDR#, RPITX, entre otros.

2.2.1. GQRX (Ubuntu)

Programa escrito en C++; permite utilizar el módulo como un analizador de espectro; GQRX posee algunas funciones predeterminadas que permiten recibir y demodular señales moduladas en amplitud y frecuencia, como por ejemplo una emisora de radio.

GQRX está diseñado para ser ejecutado en el sistema operativo Ubuntu, por lo que si se desea utilizar Windows, será necesario aplicar un programa distinto.

Figura 42. **Interfaz GQRX Ubuntu**

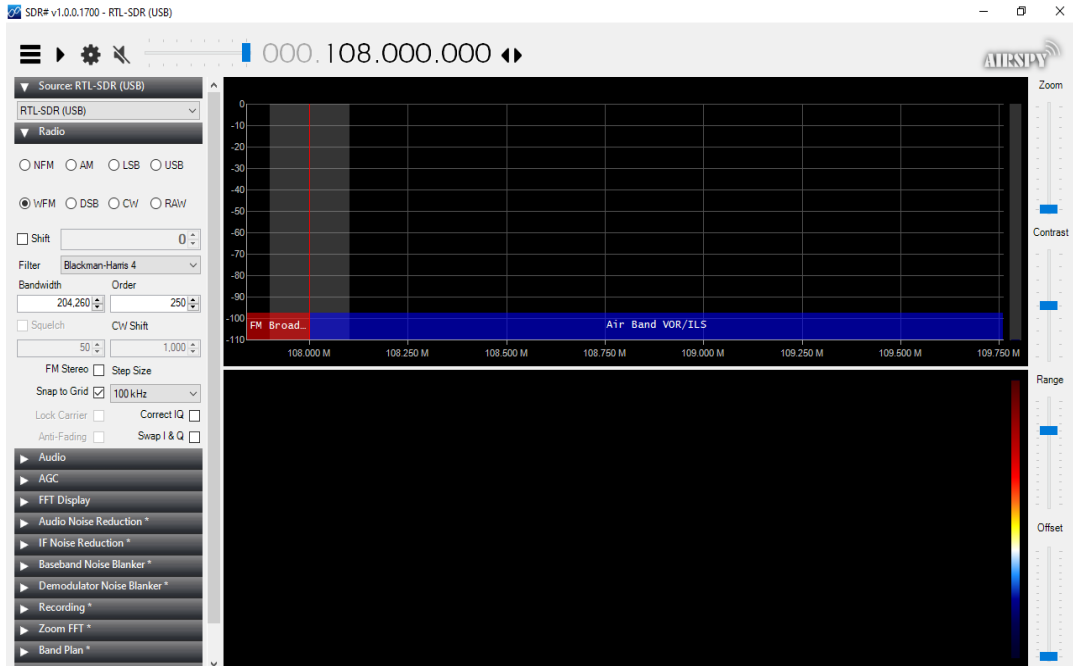


Fuente: elaboración propia, empleando GQRX.

2.2.2. **SDR# (Windows)**

Programa escrito en C#, con función similar a GQRX, que permite utilizar el módulo RTL como un analizador de espectro, con funciones de demodulación de algunos de los esquemas básicos. A diferencia de GQRX, SDR# está diseñado para trabajar sobre el sistema operativo Windows.

Figura 43. Interfaz SDR#

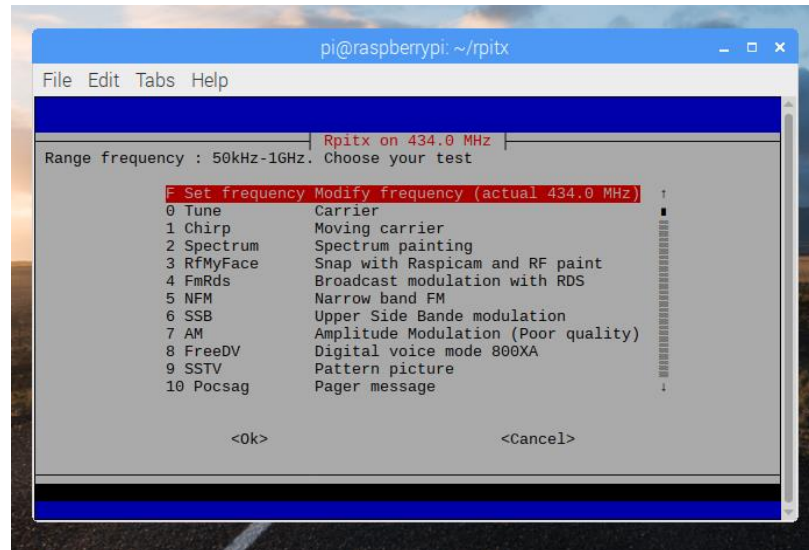


Fuente: elaboración propia, empleando SDR#.

2.2.3. Librería RPITX para transmisión en Raspberry pi

Software para la Raspberry pi, que permite transformarla en un transmisor de ondas de radio, operando desde los 5 kHz hasta los 1 500 MHz, sin la necesidad de hardware especializado para realizar la transmisión de la información.

Figura 44. Interfaz RPITX Raspberry pi

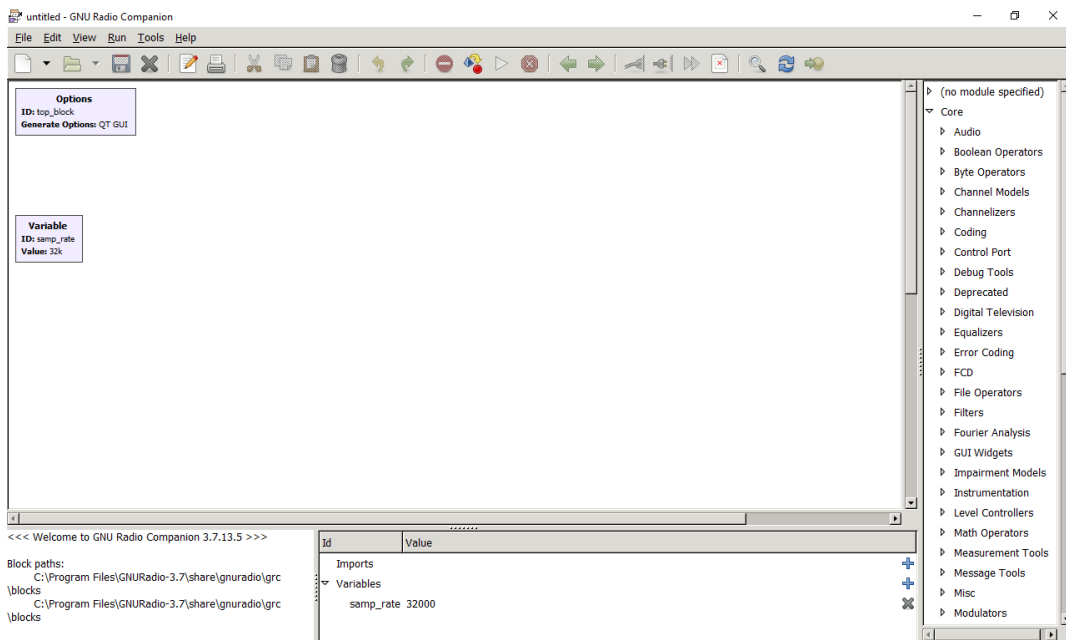


Fuente: elaboración propia, empleando *easytest* de RPITX

2.2.4. GNU Radio

Herramienta de software libre que permite utilizar una serie bloques de procesamiento digital de señales, utilizados para implementar la tecnología de radio definida por software.

Figura 45. Interfaz GNU Radio

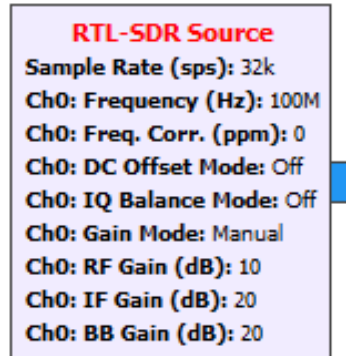


Fuente: elaboración propia, empleando GNU Radio.

2.2.4.1. Bloques

A continuación, se presentan los bloques de uso general utilizados en GNU Radio para trabajar la recepción, modulación, demodulación o transmisión de ondas de radio.

Figura 46. **Bloque fuente RTL-SDR**



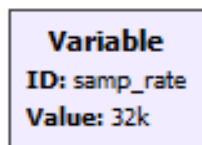
Fuente: elaboración propia, empleando GNU Radio.

Descripción: bloque utilizado para habilitar el módulo RTL como receptor.

Parámetros:

- *Sample Rate*: frecuencia con la que se obtendrán los datos del módulo RTL.
- *Ch0 frequency*: frecuencia que se desea sintonizar.

Figura 47. **Variable**



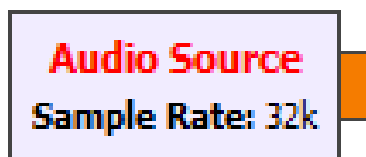
Fuente: elaboración propia, empleando GNU Radio.

Descripción: bloque que permite crear una variable para ser utilizada en uno o más bloques, por ejemplo, la frecuencia de muestreo a la que se desea trabajar.

Parámetros:

- ID: nombre asignado a la variable
- *Value*: valor asignado a la variable

Figura 48. ***Audio source***



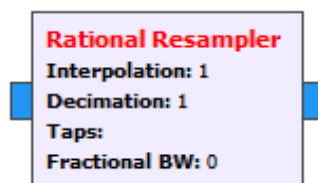
Fuente: elaboración propia, empleando GNU Radio.

Descripción: bloque el cual permite utilizar una pista de audio como señal a procesar.

Parámetros:

Sample Rate es la frecuencia a la cual está muestreada la pista de audio que será utilizada como fuente.

Figura 49. ***Rational resampler***



Fuente: elaboración propia, empleando GNU Radio.

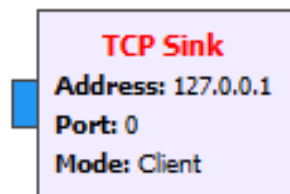
Descripción: bloque que permite modificar la frecuencia de muestreo de la señal; es usado por ejemplo para utilizar la tarjeta de sonido de la computadora como salida y así escuchar lo que se está sintonizando; se trabaja con la siguiente ecuación:

$$Sample_Rate_Out = \frac{Interpolation}{Decimation} Sample_Rate_In$$

Parámetros:

- *Interpolation*
- *Decimation*

Figura 50. **TCP sink**



Fuente: elaboración propia, empleando GNU Radio.

Descripción: bloque que permite sincronizar GNU Radio y la librería RPITX por medio de un *socket* que permite realizar la transmisión de la señal modulada trabajada con GNU Radio.

Parámetros:

- *Address*: dirección IP de la Raspberry que se utilizará como transmisor
- *Port*: puerto de la Raspberry con el que se desea trabajar

Figura 51. **Audio sink**

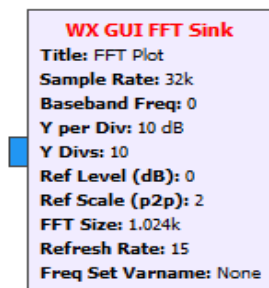


Fuente: elaboración propia, empleando GNU Radio.

Descripción: bloque con el cual se conecta GNU Radio y la tarjeta de sonido de la computadora, permitiendo escuchar lo que se está sintonizando.

Parámetros: *Sample rate*: frecuencia de muestreo en la que trabaja la tarjeta de sonido de la computadora, usualmente se trabajan valores de 48 kHz y 44,1 kHz, siendo estos valores estándar para las tarjetas de sonido de las computadoras.

Figura 52. **WX GUI FFT sink**



Fuente: elaboración propia empleando GNU Radio.

Descripción: permite graficar la transformada rápida de Fourier de la señal que se está sintonizando. Se observa su comportamiento en función de la frecuencia y su potencia. Parámetros: en *Sample Rate* se debe colocar la frecuencia de muestreo a la que se esté trabajando en el bloque anterior a este.

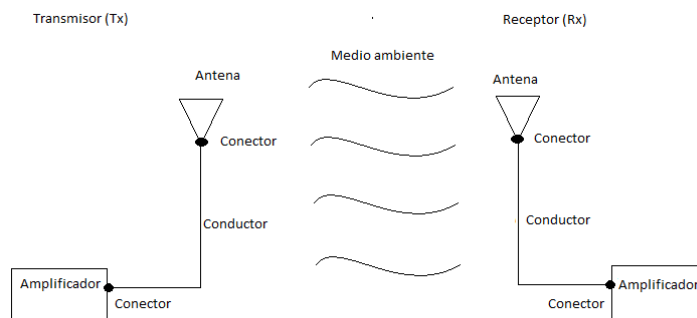
3. ENLACE EN SISTEMAS DE TELECOMUNICACIONES UTILIZANDO EL MÓDULO RTL 2832U Y RASPBERRY PI 3

En los sistemas de comunicaciones existen diversos elementos que deben ser estudiados, los cuales se presentan a continuación.

3.1. Radio enlace

Un radio enlace está conformado por una serie de elementos que permiten establecer comunicaciones de manera inalámbrica entre dos puntos, con el objetivo de conseguir un intercambio de información entre ellos. La base de un radio enlace lo constituyen los distintos tipos de antenas; sin embargo, existen diversos elementos que interactúan en el sistema que deben conocerse, ya que algunos interactúan generando pérdidas y otros, ganancia; además, en un radio enlace se debe cumplir con lo que se conoce como las zonas de Fresnell, para poder tener un radio enlace funcional y confiable.

Figura 53. Radio enlace



Fuente: elaboración propia, empelando Paint.

En un radio enlace se pueden separar los elementos en dos categorías: los que representan una pérdida al sistema y los que significan una ganancia; estos se dividen de la siguiente manera:

Tabla I. **Ganancias y pérdidas en un radio enlace**

Ganancia	Pérdida
Antena transmisora Tx (ganancia de la antena).	Conectores
Amplificador transmisor Tx (ganancia del amplificador).	Conductores
Antena receptora Rx (ganancia de la antena).	Medio de transmisión (es afectado por condiciones atmosféricas, efectos del suelo y elementos que varíen las zonas de Fresnell).
Amplificador receptor (ganancia del amplificador).	

Fuente: elaboración propia.

El espectro electromagnético se puede dividir en las siguientes bandas:

Tabla II. **División del espectro electromagnético**

Banda	Frecuencia inicial	Frecuencia final
<i>Extremely low frequency</i> (ELF)	3 Hz	30 Hz
<i>Super low frequency</i> (SLF)	30 Hz	300 Hz
<i>Ultra low frequency</i> (ULF)	300 Hz	3 kHz
<i>Very low frequency</i> (VLF)	3 kHz	30 kHz

Continuación de la tabla II.

<i>Low frequency</i> (LF)	30 kHz	300 kHz
<i>Medium frequency</i> (MF)	300 kHz	3 MHz
<i>High frequency</i> (HF)	3 MHz	30 MHz
<i>Very high frequency</i> (VHF)	30 MHz	300 MHz
<i>Ultra high frequency</i> (UHF)	300 MHz	3 GHz
<i>Super high frequency</i> (SHF)	3 GHz	30 GHz
<i>Extremely high frequency</i> (EHF)	30 GHz	300 GHz

Fuente: elaboración propia.

3.1.1. Antenas

Una antena es un dispositivo normalmente construido a base de un material conductor, con el objetivo de permitir la transmisión y recepción de ondas electromagnéticas, y así lograr el intercambio de información de un punto a otro en un radio enlace.

Las antenas se pueden dividir en: elementales, arreglos de antenas, de onda progresiva y parabólicas; cada una de estas con parámetros específicos de los cuales dependerá su aplicación.

3.1.1.1. Parámetros de antenas

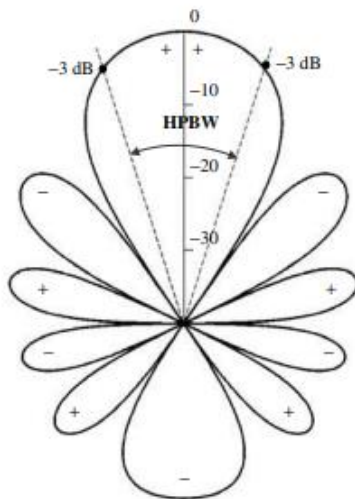
Entre los principales parámetros de las antenas, se puede encontrar el diagrama o patrón de radiación, ancho de banda, directividad, adaptación, eficiencia, impedancia y polarización.

Diagrama de radiación: en una antena, el diagrama o patrón de radiación representa de manera gráfica el comportamiento de las ondas electromagnéticas irradiadas, normalmente representa la densidad de potencia radiada.

Según el comportamiento del patrón de radiación, las antenas se pueden clasificar en: direccionales, omnidireccionales e isotrópicas.

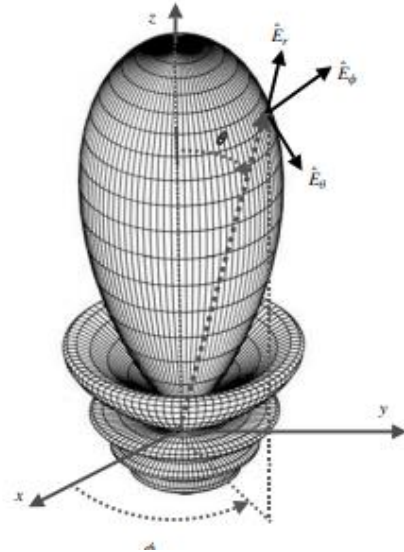
Una antena con patrón de radiación direccional o unidireccional es aquella que enfoca la mayor cantidad de potencia radiada en una dirección, logrando así una mayor distancia de transmisión. En la siguiente gráfica se observa el patrón de radiación de una antena direccional.

Figura 54. **Patrón de radiación direccional 2 dimensiones**



Fuente: BALANIS, Constantine. *Antenna Theory third edition analysis and design*. p. 29.

Figura 55. **Patrón de radiación direccional 3 dimensiones**

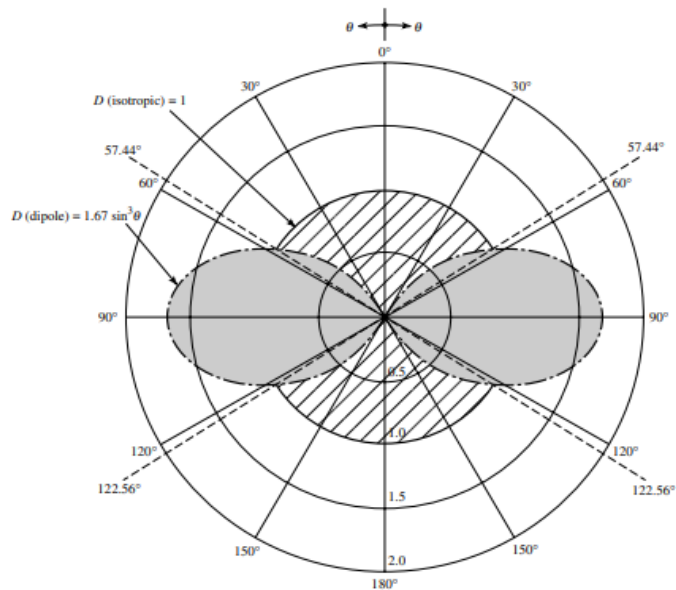


Fuente: BALANIS, Constantine. *Antenna Theory third edition analysis and design*. p. 31.

Una antena con patrón de radiación de una antena omnidireccional es aquella capaz de irradiar de manera uniforme en todas direcciones; sin embargo, la antena capaz de radiar de manera uniforme en 3 dimensiones es la isotrópica.

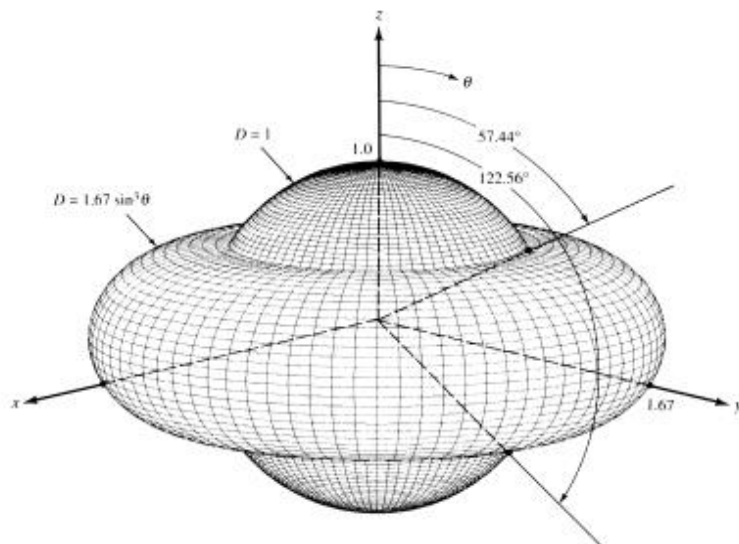
En la siguiente gráfica se observa el patrón de radiación de un dipolo, categorizado como antena omnidireccional.

Figura 56. **Patrón de radiación antena omnidireccional 2 dimensiones**



Fuente: BALANIS, Constantine. *Antenna Theory third edition analysis and design*. p. 49.

Figura 57. **Patrón de radiación antena omnidireccional 3 dimensiones**



Fuente: BALANIS, Constantine. *Antenna Theory third edition analysis and design*. p. 29.

Ancho de banda: se conoce como el rango de frecuencias en el cual la antena puede trabajar de manera óptima; está delimitado por las frecuencias en las cuales la intensidad de la señal que recibe la antena decrece 3 dB respecto de la intensidad con la que recibe las señales a la frecuencia para la que fue diseñada.

Directividad: es la relación de la potencia radiada en una dirección con la potencia radiada de forma isotrópica. Se puede calcular con la siguiente ecuación:

$$D_{(\theta,\varphi)} = \frac{P_{(\theta,\varphi)}}{\frac{N_r}{4\pi r}}$$

Donde la expresión $P_{(\theta,\varphi)}$ representa la potencia radiada en una dirección y la expresión $\frac{N_r}{4\pi r}$ la potencia radiada de forma isotrópica.

Adaptación: consiste en aplicar el teorema de la máxima transferencia de potencia, de tal manera que en el sistema la impedancia de entrada sea igual a la de salida. Si una antena no está adaptada, parte de la potencia será reflejada, siendo esto una pérdida para el sistema.

Eficiencia: es la relación que existe entre la ganancia y la directividad, representada con el símbolo η en la siguiente ecuación:

$$\eta = \frac{\text{Ganancia}}{\text{Directividad}}$$

La impedancia se puede descomponer en varios elementos:

- -Ra: resistencia física de la antena; esta depende del material utilizado para la construcción de la antena, así como el tipo de antena.
- -Rr: resistencia de radiación.
- -X: reactancia (este valor se encuentra en función de la frecuencia de operación).

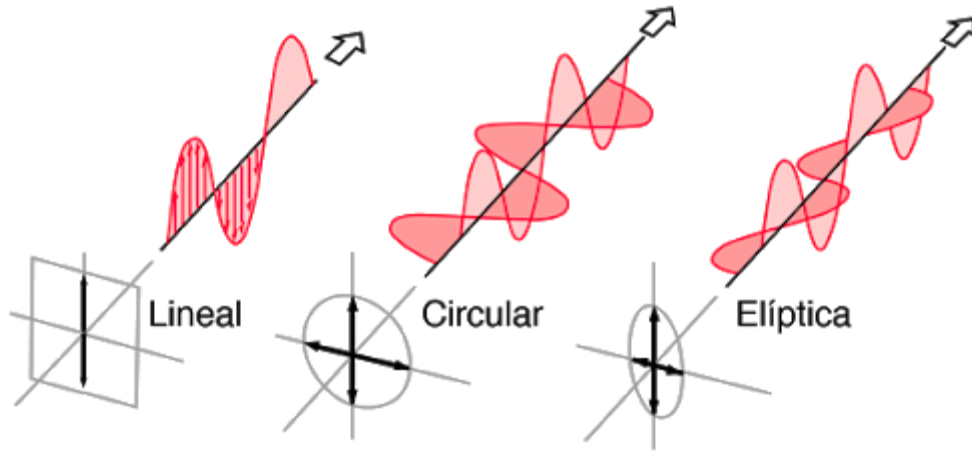
La impedancia de una antena se puede obtener a partir de la siguiente ecuación:

$$Z = (Ra + Rr) + jX$$

Se tiene que la componente real de la impedancia se obtiene por la suma de la resistencia física y la radiación; la componente compleja está dada por la reactancia de la antena.

Polarización: la polarización en las ondas electromagnéticas sirve para conocer el comportamiento de los campos eléctricos y magnéticos en una onda transversal; existe la polarización lineal, circular y elíptica.

Figura 58. **Tipos de polarización**



Fuente: *Hyperphysics. Tipos de polarización.* <http://hyperphysics.phy-astr.gsu.edu/hbasees/phyopt/polclas.html> Consulta: junio 2020.

3.1.1.2. Tipos de antenas

Según las características de las antenas y su aplicación, es posible dividir las en: elementales, como el dipolo y la espira; de onda progresiva como las de apertura y de bocina; los arreglos de antenas, como las antenas Yagi; para finalizar con las antenas parabólicas o tipo plato.

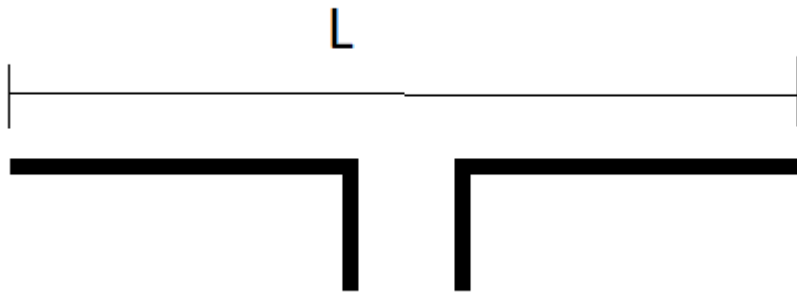
Tabla III. **Rango de operación antenas**

Tipo de antena	Rango de operación
Elemental	10 kHz – 1 GHz
Onda progresiva	1 MHz – 10 GHz
Arreglo de antenas	10 MHz – 10 GHz
Parabólicas	1 GHz – 100 GHz

Fuente: elaboración propia.

- Dipolo elemental: es un conductor de corriente con longitud “L”, el cual es recorrido por una corriente uniforme, cuyas dimensiones son pequeñas comparadas con su longitud de onda; la mayor parte de las antenas que trabajan en frecuencias inferiores a 1MHz se comportan como dipolos elementales.

Figura 59. **Dipolo**



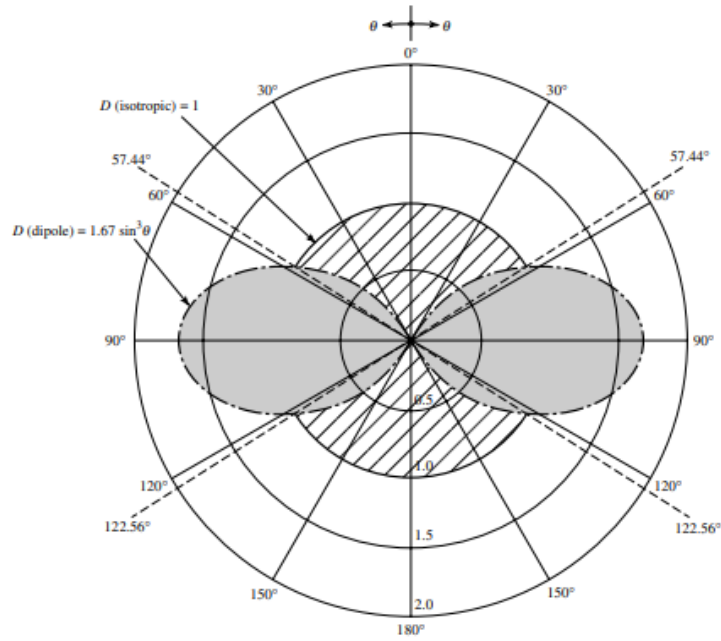
Fuente: elaboración propia, empleando Paint.

En un dipolo, la longitud “L” está dada por la siguiente ecuación:

$$L = \frac{\lambda}{2}$$

Donde λ es la longitud de onda de la frecuencia con la que trabajará el sistema.

Figura 60. **Patrón de radiación dipolo**



Fuente: BALANIS, Constantine. *Antenna Theory third edition analysis and design*. p. 49.

Tabla IV. **Parámetros dipolo elemental**

Directividad	1.5
Patrón de radiación	Omnidireccional
Longitud	$\frac{\lambda}{2}$

Fuente: elaboración propia.

- Arreglo de antenas: consisten en la agrupación de uno o más tipos de antenas simples unidas; se pueden tener arreglos de antenas lineales, de antenas planas, y cilíndricos; su utilización dependerá de la aplicación.

- Antenas lineales: utilizadas normalmente para la transmisión de emisoras de radio y televisión, entre ellas se puede encontrar la antena Yagi, que es una antena direccional compuesta de una serie de dipolos colocados uno tras otro.

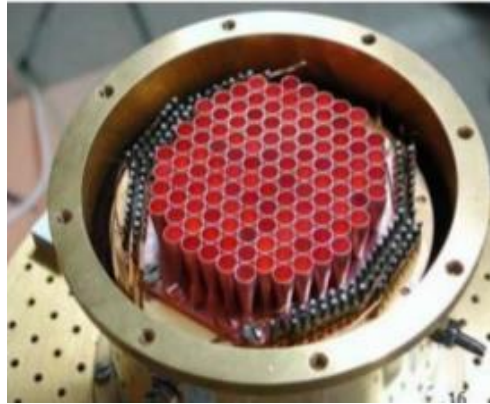
Figura 61. **Antena Yagi**



Fuente: *Directindustry.Antena Yagi*. <https://www.directindustry.es/prod/cstel/product-188490-1953235.html>. Consulta: junio de 2020.

- Arreglo de antenas planas: son utilizados normalmente en las comunicaciones satelitales; este tipo de arreglo se consigue colocando todos los elementos sobre un plano, prácticamente de 2 dimensiones; consiguiendo así una mayor directividad y control del patrón de radiación; los arreglos de antenas planas más comunes son los reticulares y los circulares.

Figura 62. **Arreglo de antenas planas circular**

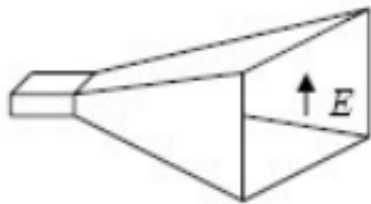


Fuente: *Slideshare. Arreglo de antenas planas circular.*

<https://es.slideshare.net/ErickPereiraPolo/teora-array-de-antenas>. Consulta: junio de 2020.

- Antenas de apertura: son aquellas que buscan concentrar la radiación electromagnética en un punto, de tal manera que consiguen concentrar las ondas de radio en una dirección, para conseguir una mayor directividad. Entre estas se pueden mencionar las antenas de bocina y los reflectores parabólicos.
- Antenas de bocina: son utilizadas, en la mayoría de los casos, para trabajar con microondas; tienden a tener un ancho de banda bastante amplio; las antenas tipo bocina consisten en una guía de onda cuya área va en incremento hasta un extremo abierto en el cual ingresan las ondas electromagnéticas.

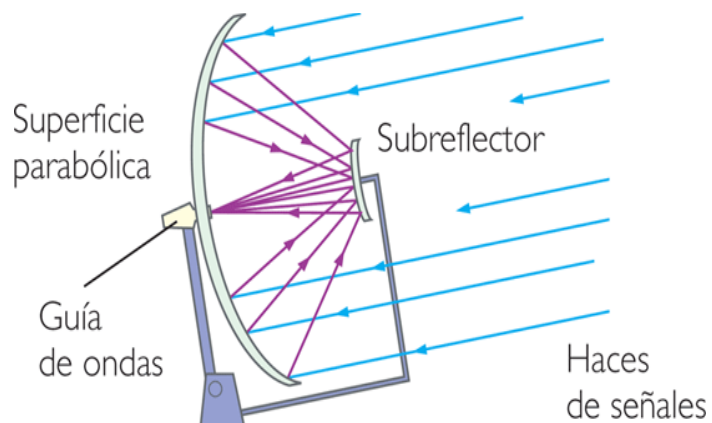
Figura 63. **Antena tipo bocina**



Fuente: VELMA, Mario. *Introducción a las antenas*. p. 31.

- Reflector parabólico: este concentra la energía en un punto conocido como foco, al hacer rebotar las ondas electromagnéticas en una superficie reflectora.

Figura 64. **Antena parabólica**



Fuente: Oscar-lunaceti. *Antena parabólica*. http://oscar-lunaceti.mex.tl/1912102_Antenas.html.

Consulta: mayo de 2020.

Tabla V. **Comparación de antenas**

Tipo antena	Patrón de radiación	Ganancia	Directividad	Polarización
Dipolo	Amplio	Baja	Baja	Lineal
Yagi	<i>Endfire</i>	Media/alta	Media/alta	Lineal
Parabólica	Amplio	Alta	Alta	Lineal/circular
Ranura	<i>Endfire</i>	Media	Media	Lineal

Fuente: elaboración propia.

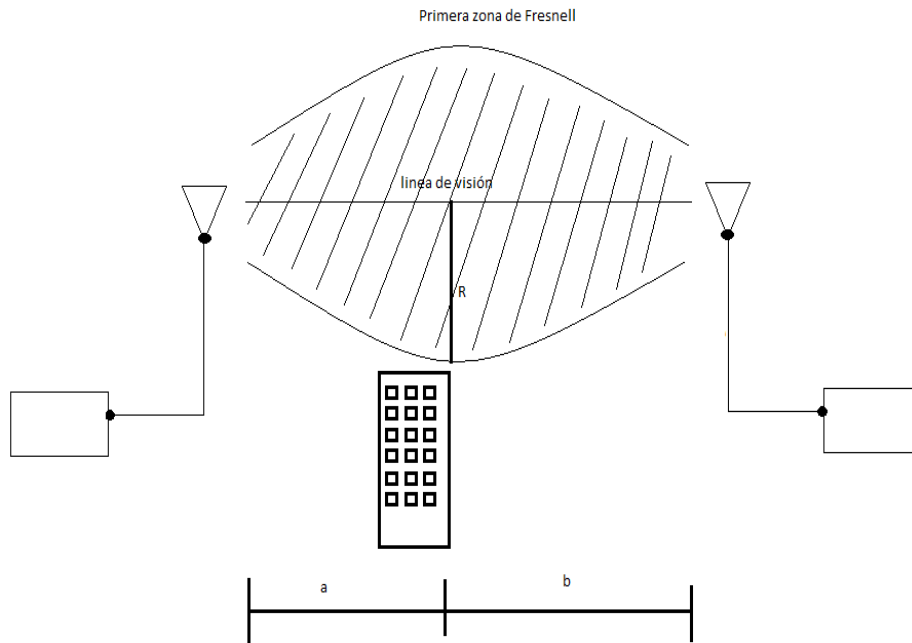
3.1.2. **Zonas de Fresnell**

Se conoce con este nombre al volumen de espacio entre el emisor y el receptor; generalmente, se busca que la primera zona de Fresnell tenga un 80 % de su volumen libre, de cualquier elemento que pueda interferir.

Para determinar o calcular la distancia que se debe tener entre algún obstáculo se utiliza la siguiente ecuación:

$$R = \sqrt{\frac{\lambda ab}{a + b}}$$

Figura 65. Zonas de Fresnell

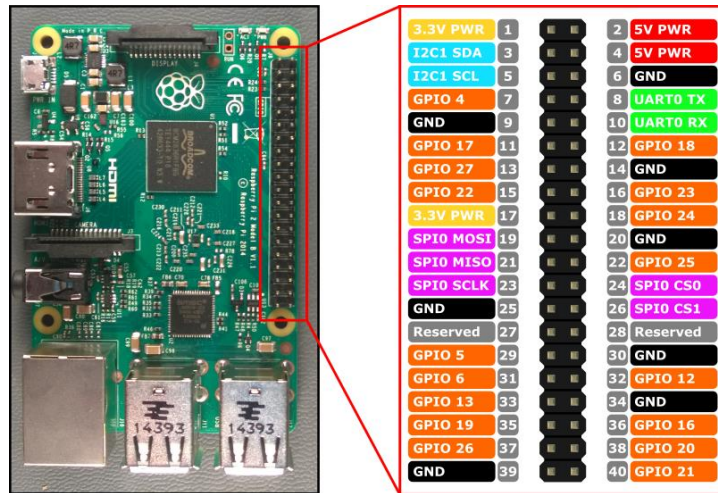


Fuente: elaboración propia, empleando Paint.

3.2. Raspberry pi 3 como transmisor de señales de radio frecuencia

La librería RPITX permite utilizar una Raspberry pi como transmisor de radio frecuencia sin la necesidad de contar con hardware especial, ya que utiliza uno de los GPIOs (*general purpose input/output*, entrada/salida de propósito general) como transmisor; necesita solamente un pequeño cable, que se debe conectar al pin #7 de la Raspberry pi, con el fin de ser utilizado como antena.

Figura 66. GPIO pinout Raspberry pi 3



Fuente: Prometec. GPIO pinout Raspberry pi 3 <https://www.prometec.net/rpi-gpio/>. Consulta: julio de 2020.

Para trabajar un transmisor de radio con GNU Radio y Rpitx, es necesario realizar el proceso de modulación en GNU Radio y para finalizar se envían los datos a un *socket* dentro de la Raspberry pi, antes de accionar el programa de GNU Radio se debe ejecutar un comando en consola, el cual habilita el *socket* y se establece la frecuencia en la que se desea transmitir.

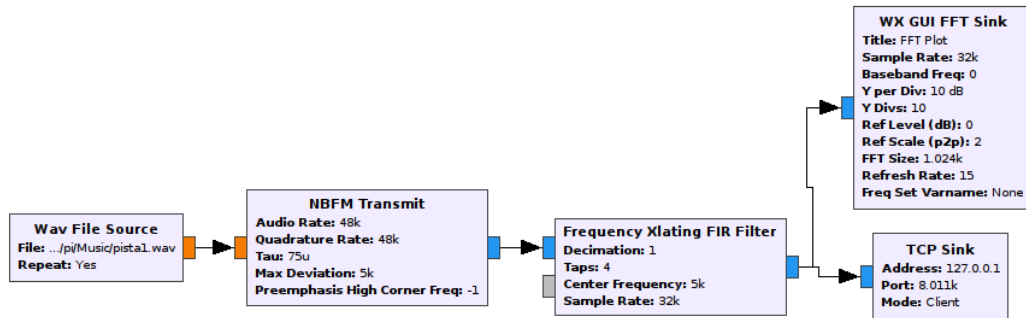
El comando que se debe ejecutar es el siguiente:

```
nc -l 9999 | sudo rpitx -i- -m IQFLOAT -f 96900
```

nc -l 9999: indica que el puerto que está siendo utilizado es el 9999.

-f 96900: indica que se estará trabajando a una frecuencia de 96,9 MHz.

Figura 67. **Diagrama de bloques transmisor NBFM con GNU Radio**



Fuente: elaboración propia, empleando GNU Radio.

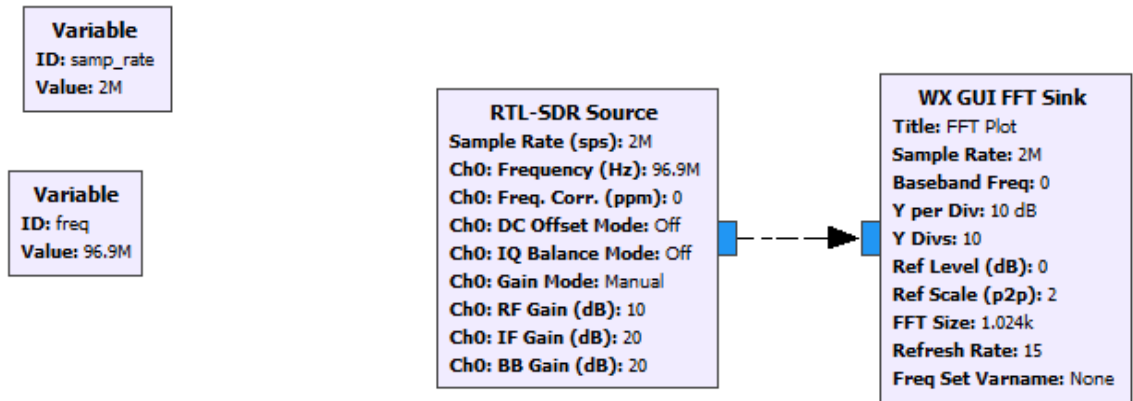
3.3. Receptor de señales de radio frecuencias con GNU Radio y módulo RTL 2832u

Empezando con la implementación de la tecnología radio definida por software, se muestran a continuación los bloques utilizados en GNU Radio, los cuales permiten habilitar un módulo USB basado en el chip de Realtek 2832u como un receptor de radio frecuencias, desplegando una gráfica que muestra la transformada rápida de Fourier.

Para la elaboración del programa que funcionará como un receptor de radio frecuencias en GNU Radio serán necesarios los siguientes bloques:

- Variable
- Rtl-SDR Source
- WX GUI FFT Sink

Figura 68. Diagrama de bloques receptor de señales de radio frecuencias con GNU Radio



Fuente: elaboración propia, empleando GNU Radio.

4. PRÁCTICAS APLICABLES AL LABORATORIO DE COMUNICACIONES 1

A continuación, se presenta una serie de prácticas aplicables al laboratorio de Comunicaciones 1, de la carrera de Ingeniería Eléctrica de la Facultad de ingeniería de la Universidad de San Carlos de Guatemala.

4.1. Práctica 1: instalación de *drivers* y software en sistema operativo Ubuntu

La primera práctica aplicable al laboratorio de Comunicaciones 1 consiste en la instalación del software necesario, así como los *drivers* y librerías que se utilizarán en la implementación de la tecnología radio definida por software.

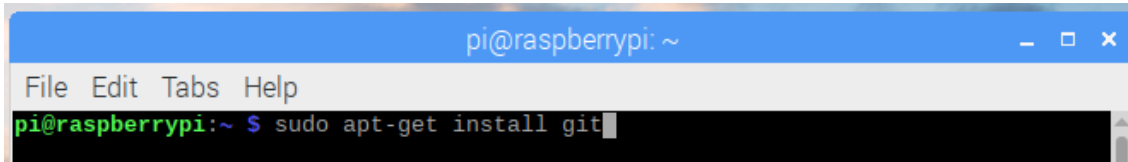
4.1.1. Driver módulo RTL

Para la utilización del módulo USB como receptor, ya sea en GNU Radio, GQRX o cualquier otro software que permita trabajar la tecnología radio definida por software, es necesario seguir una serie de pasos en el orden que se presenta a continuación.

4.1.1.1. Instalación *git*

Es necesario descargar los *drivers* de un repositorio en GitHub, por lo que el primer paso debe ser instalar la librería *git*, la cual permite acceder y descargar los repositorios; esto se realiza con el siguiente comando: *Sudo apt-get install git*.

Figura 69. **Instalación librería *git***

A screenshot of a terminal window on a Raspberry Pi. The window title is 'pi@raspberrypi: ~'. The menu bar shows 'File Edit Tabs Help'. The terminal prompt is 'pi@raspberrypi:~ \$' and the command 'sudo apt-get install git' is being entered, with a cursor at the end of the line.

Fuente: elaboración propia, empleando consola Raspberry pi.

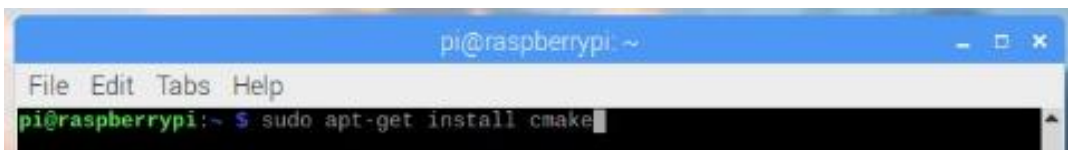
4.1.1.2. **Instalación *Cmake***

Con la librería *git* instalada, es necesario contar con la librería *cmake*, perteneciente al grupo de herramientas conocidas como “*Autotools*”, las cuales facilitan la compilación de proyectos en plataformas tipo Unix, Mac OSX y Windows.

Para instalar la librería *CMake*, se utiliza el siguiente código en consola:

Sudo apt-get install cmake

Figura 70. **Instalación *CMake***

A screenshot of a terminal window on a Raspberry Pi. The window title is 'pi@raspberrypi: ~'. The menu bar shows 'File Edit Tabs Help'. The terminal prompt is 'pi@raspberrypi:~ \$' and the command 'sudo apt-get install cmake' is being entered, with a cursor at the end of the line.

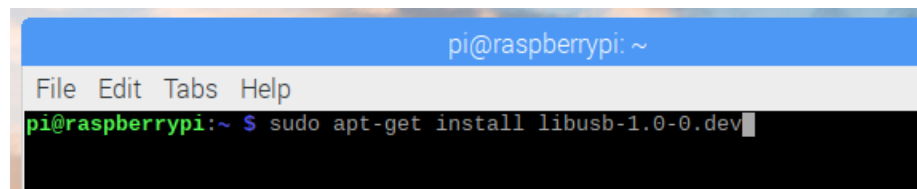
Fuente: elaboración propia, empleando consola Raspberry pi.

4.1.1.3. Instalación *libusb*

Para la utilización del módulo es necesario contar con la librería “*libusb*”, la cual permite acceso a los dispositivos USB conectados; para la instalación se utiliza el siguiente código:

```
Sudo apt-get install libusb-1.0-0.dev
```

Figura 71. Instalación *libusb*

A screenshot of a terminal window on a Raspberry Pi. The window title is 'pi@raspberrypi: ~'. The terminal shows a menu bar with 'File Edit Tabs Help'. The prompt is 'pi@raspberrypi:~ \$' and the command 'sudo apt-get install libusb-1.0-0.dev' is being typed, with a cursor at the end of the line.

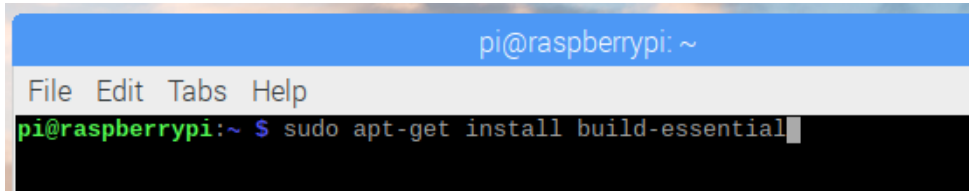
Fuente: elaboración propia, empleando consola Raspberry pi.

4.1.1.4. Instalación *build-essential*

Otra librería de suma importancia para la instalación es “*build-essential*”, la cual se instala con el comando:

```
Sudo apt-get install build-essential
```

Figura 72. Instalación *build-essential*



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ sudo apt-get install build-essential
```

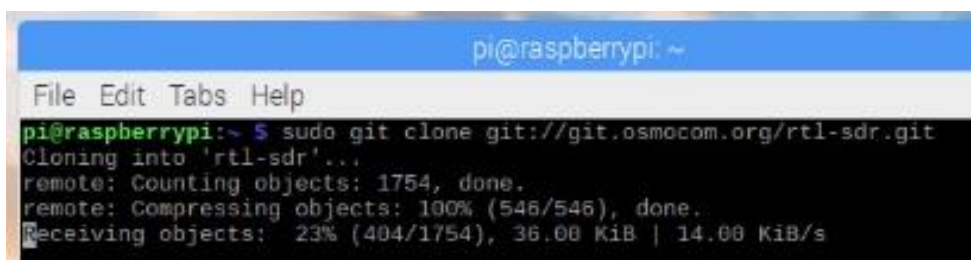
Fuente: elaboración propia, empleando consola Raspberry pi.

4.1.1.5. Descarga de repositorio *driver* RTL-SDR

Se debe obtener el *driver* del módulo USB desde un repositorio de *git*, para la descarga se utilizará la librería *gitclone* con el siguiente comando; esto creará una carpeta llamada “RTL-SDR” en la cual se descargará toda la información dentro del repositorio.

```
Sudo git clone git://git.osmocom.org/rtl-sdr.git
```

Figura 73. *Gitclone*



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ sudo git clone git://git.osmocom.org/rtl-sdr.git  
Cloning into 'rtl-sdr'...  
remote: Counting objects: 1754, done.  
remote: Compressing objects: 100% (546/546), done.  
Receiving objects: 23% (404/1754), 36.00 KiB | 14.00 KiB/s
```

Fuente: elaboración propia, empleando consola Raspberry pi.

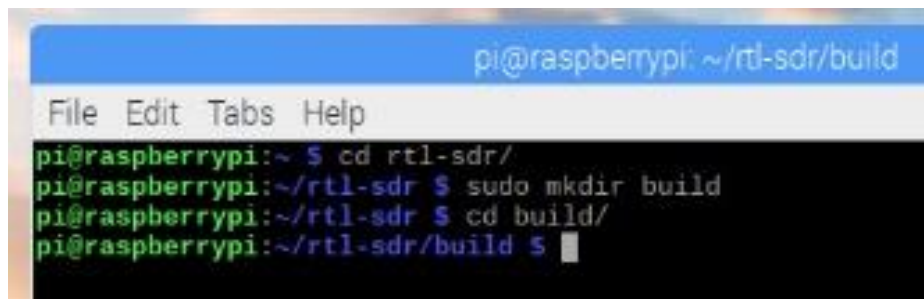
4.1.1.6. Configuración e instalación

Para la instalación del *driver* es necesario seguir los siguientes pasos:

- Paso 1: dentro de la carpeta en la cual se descargó el repositorio de *git* (RTL-SDR) se debe crear un directorio con el nombre de “*build*”; para eso se utilizará el comando “*mkdir*” de la siguiente forma:

Sudo mkdir build

Figura 74. ***Mkdir build***



```
pi@raspberrypi: ~/rtl-sdr/build
File Edit Tabs Help
pi@raspberrypi:~ $ cd rtl-sdr/
pi@raspberrypi:~/rtl-sdr $ sudo mkdir build
pi@raspberrypi:~/rtl-sdr $ cd build/
pi@raspberrypi:~/rtl-sdr/build $
```

Fuente: elaboración propia, empleando consola Raspberry pi.

- Paso 2: dentro de la carpeta *build*, es necesario ejecutar el comando “*cmake*”; para compilar los paquetes debe ejecutarse de la siguiente forma:

Sudo cmake ../-DINSTALL_UDEV_RULES=ON

Figura 75. **CMAKE**



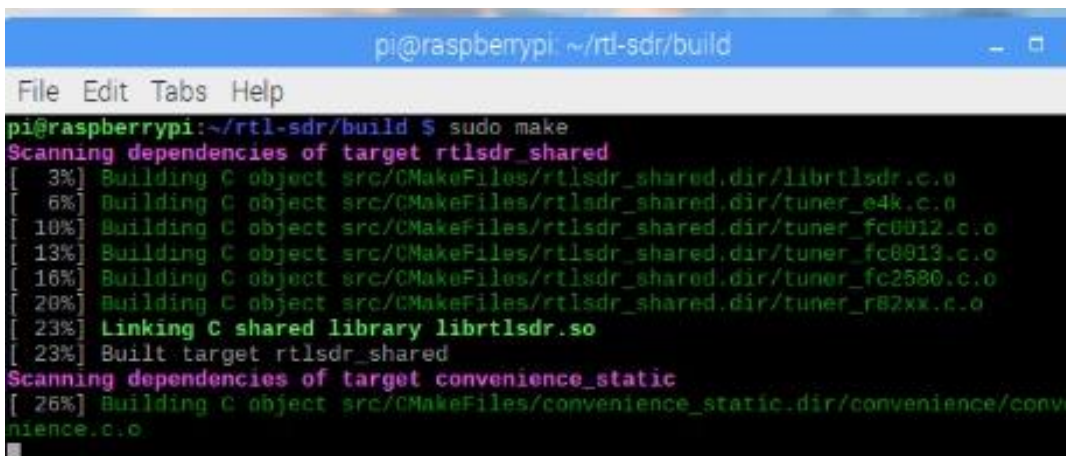
```
pi@raspberrypi: ~/rtl-sdr/build
File Edit Tabs Help
pi@raspberrypi:~/rtl-sdr/build $ sudo cmake ../ -DINSTALL_UDEV_RULES=ON
```

Fuente: elaboración propia, empleando consola Rasberry pi.

- Paso 3: se ejecuta el comando “make” para la compilación de las piezas que necesiten ser recompiladas; se ingresará el siguiente comando en consola:

Sudo make

Figura 76. **Make**



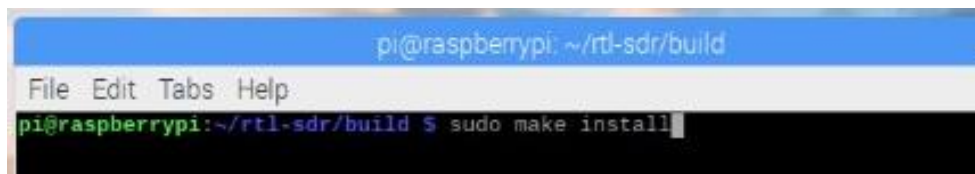
```
pi@raspberrypi: ~/rtl-sdr/build
File Edit Tabs Help
pi@raspberrypi:~/rtl-sdr/build $ sudo make
Scanning dependencies of target rtl_sdr_shared
[ 3%] Building C object src/CMakeFiles/rtl_sdr_shared.dir/librtlsdr.c.o
[ 6%] Building C object src/CMakeFiles/rtl_sdr_shared.dir/tuner_e4k.c.o
[ 10%] Building C object src/CMakeFiles/rtl_sdr_shared.dir/tuner_fc0012.c.o
[ 13%] Building C object src/CMakeFiles/rtl_sdr_shared.dir/tuner_fc0013.c.o
[ 16%] Building C object src/CMakeFiles/rtl_sdr_shared.dir/tuner_fc2580.c.o
[ 20%] Building C object src/CMakeFiles/rtl_sdr_shared.dir/tuner_r82xx.c.o
[ 23%] Linking C shared library librtlsdr.so
[ 23%] Built target rtl_sdr_shared
Scanning dependencies of target convenience_static
[ 26%] Building C object src/CMakeFiles/convenience_static.dir/convenience/conve
```

Fuente: elaboración propia, empleando consola Rasberry pi.

- Paso 4: con todos los componentes compilados, es necesario ejecutar el comando *make install* para finalizar el proceso de instalación del *driver*, para proceder con la configuración. Para ejecutar se ingresará el siguiente comando en consola:

Sudo make install

Figura 77. **Make Install**



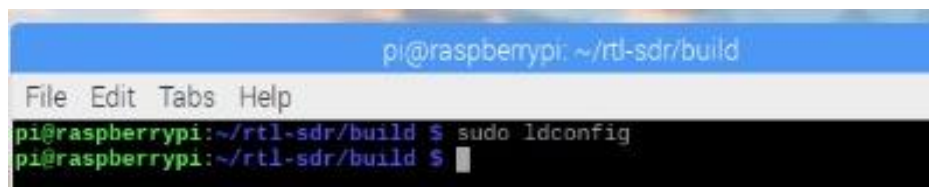
```
pi@raspberrypi: ~/rtl-sdr/build
File Edit Tabs Help
pi@raspberrypi:~/rtl-sdr/build $ sudo make install
```

Fuente: elaboración propia, empleando consola Raspberry pi.

- Paso 5: es necesario crear los vínculos a las bibliotecas en el archivos */etc/ld.so.conf* y en los directorios */lib* y */usr/lib*; para eso se utilizará el siguiente comando:

Sudo ldconfig

Figura 78. **ldconfig**



```
pi@raspberrypi: ~/rtl-sdr/build
File Edit Tabs Help
pi@raspberrypi:~/rtl-sdr/build $ sudo ldconfig
pi@raspberrypi:~/rtl-sdr/build $
```

Fuente: elaboración propia, empleando consola Raspberry pi.

- Paso 6: se debe copiar el archivo *rtl-sdr.rules* dentro del directorio */rules.d*; para esto se utilizará el comando *cp* de la siguiente manera:

```
Sudo cp ../rtl-sdr.rules /etc/udev/rules.d/
```

Figura 79. **Rtl-sdr.rules**



Fuente: elaboración propia, empleando consola Raspberry pi.

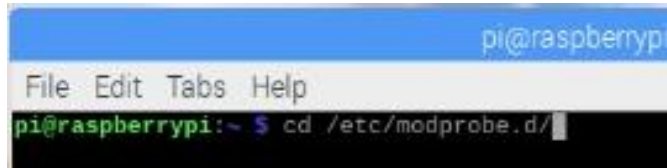
4.1.1.7. Creación *blacklist*

Para indicarle al sistema operativo que debe utilizar el nuevo *driver* y no el que tiene el sistema operativo, es necesario agregarlos en un archivo llamado *Blacklist-rtl.conf*; para esto se deben seguir los siguientes pasos:

- Paso 1: el archivo *blacklist* debe ser creado en el directorio *modprobe.d* del sistema operativo; para esto se utilizará el siguiente comando para acceder a dicho directorio:

```
cd /etc/modprobe.d/
```

Figura 80. Directorio *modprobe.d*



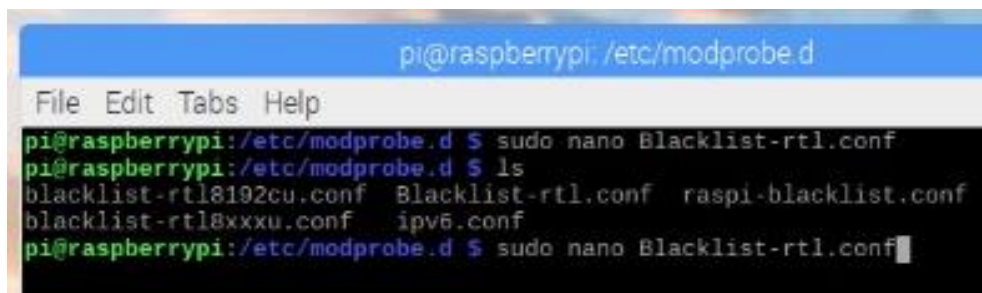
```
pi@raspberrypi:~$ cd /etc/modprobe.d/
```

Fuente: elaboración propia, empleando consola Raspberry pi.

- Paso 2: dentro del directorio se creará el archivo *Blacklist-rtl.conf* utilizando el siguiente comando; el cual además de crear el archivo lo abrirá para su edición:

Sudo nano Blacklist-rtl.conf

Figura 81. *Blacklist-rtl.conf*



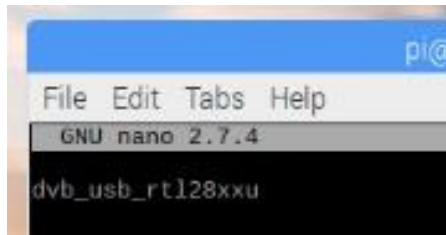
```
pi@raspberrypi:~$ sudo nano Blacklist-rtl.conf
pi@raspberrypi:~$ cd /etc/modprobe.d/
pi@raspberrypi:/etc/modprobe.d$ ls
blacklist-rtl8192cu.conf  Blacklist-rtl.conf  raspi-blacklist.conf
blacklist-rtl8xxxu.conf  ipv6.conf
pi@raspberrypi:/etc/modprobe.d$ sudo nano Blacklist-rtl.conf
```

Fuente: elaboración propia, empleando consola Raspberry pi.

- Paso 3: dentro del archivo se deberá agregar la siguiente línea y se procederá a guardar y cerrarlo.

dvb_usb_rtl28xxu

Figura 82. **Contenido *Blacklist-rtl.conf***

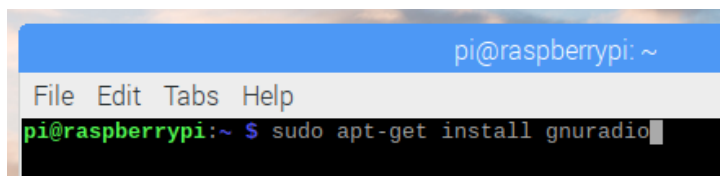


Fuente: elaboración propia, empleando consola Raspberry pi.

4.1.2. **GNU Radio**

Para la instalación del programa GNU Radio utilizado para la recepción por medio de bloques, basta con ingresar el siguiente comando en la consola para realizar la instalación: *Sudo apt-get install gnuradio*.

Figura 83. **Instalación GNU Radio**



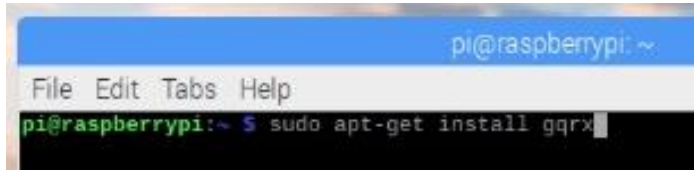
Fuente: elaboración propia, empleando consola Raspberry pi.

4.1.3. **GQRX**

Para la instalación del programa GQRX utilizado para la recepción, basta con ingresar el siguiente comando en la consola para realizar la instalación:

Sudo apt-get install gqrx

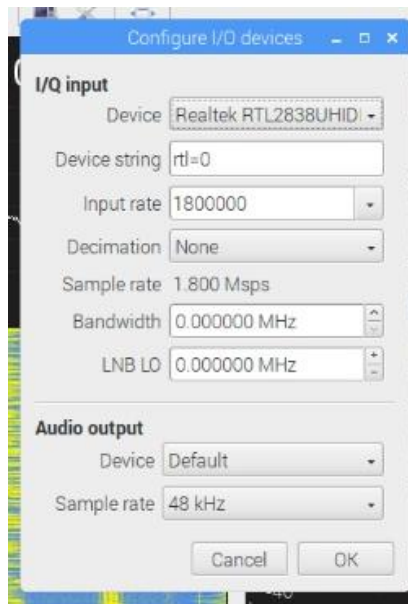
Figura 84. **Instalación GQRX**



Fuente: elaboración propia, empleando consola Rasberry pi.

Una vez instalado GQRX se debe verificar la configuración; se selecciona el dispositivo utilizado para la recepción; en este caso el módulo Realtek RTL2838u, con una tasa de datos (*input rate*) de 1800000.

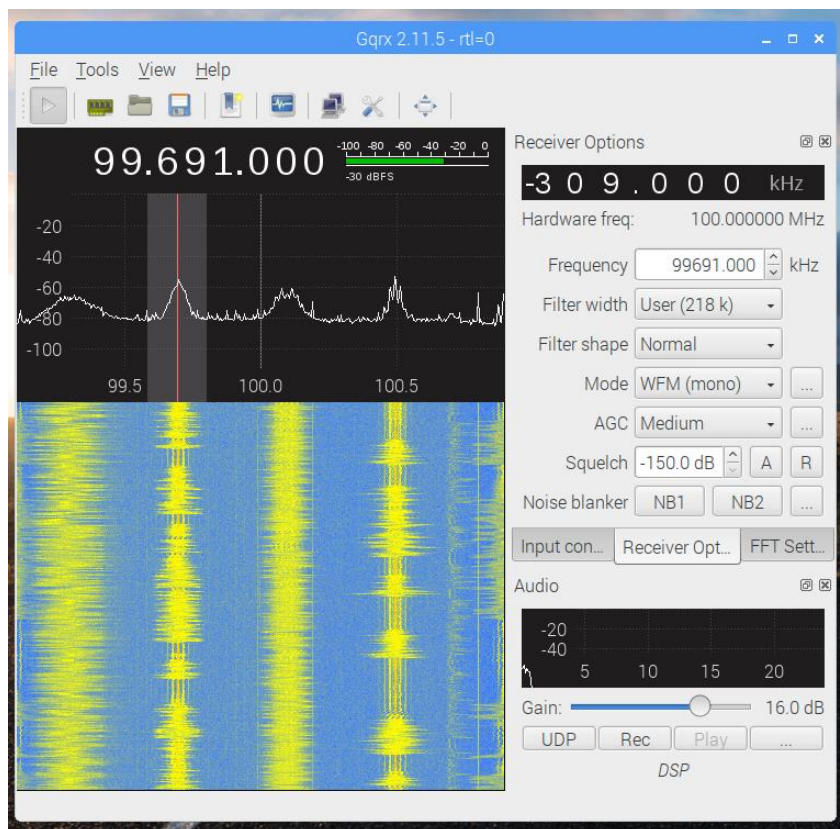
Figura 85. **Configuración GQRX**



Fuente: elaboracion propia, empleando GQRX.

Con los parámetros del módulo USB configurados se procede a realizar la prueba del funcionamiento del software y el módulo RTL; al sintonizar en el rango de FM comercial, se debe observar algo similar a lo desplegado a continuación:

Figura 86. Prueba GQRX



Fuente: elaboración propia, empleando GQRX.

4.1.4. RPITX

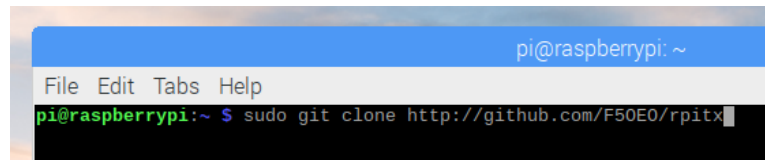
La librería RPITX permite utilizar la Raspberry como transmisor; debe ser instalada utilizando el siguiente procedimiento, en el cual además de realizar la instalación, se llevará a cabo la prueba de funcionamiento de la misma.

4.1.4.1. Repositorio RPITX

Para la instalación de la librería RPITX se debe clonar el repositorio de la librería a la computadora con el siguiente comando:

```
Sudo git clone http://github.com/F50E0/rpitx
```

Figura 87. **Gitclone RPITX**



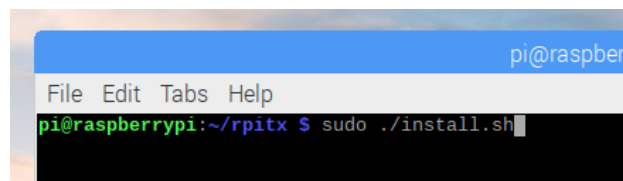
```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ sudo git clone http://github.com/F50E0/rpitx
```

Fuente: elaboración propia, empleando consola Raspberry pi.

Con la librería clonada del repositorio de git, se ejecuta el script que instalará la librería, primero ingresando al directorio y luego ejecutando la instalación con los siguientes comandos:

```
cd rpitxsudo .install.sh
```

Figura 88. **Instalación RPITX**



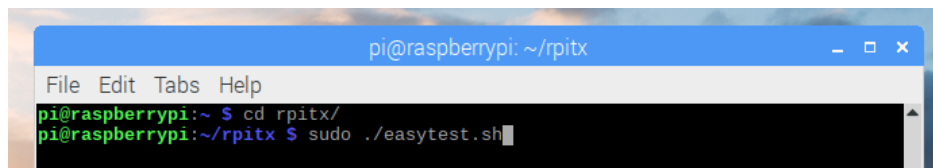
```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~/rpitx $ sudo ./install.sh
```

Fuente: elaboración propia, empleando consola Raspberry pi.

Luego de los pasos anteriores se procede a reiniciar la Raspberry, para luego realizar el test que permite verificar la correcta instalación; se ingresa al directorio en el que se instaló la librería RPITX y se procede a ejecutar el “easytest”, el cual permitirá confirmar la instalación y el correcto funcionamiento de la librería; este se ejecutará ingresando el siguiente comando en la consola:

```
sudo ./easytest.sh
```

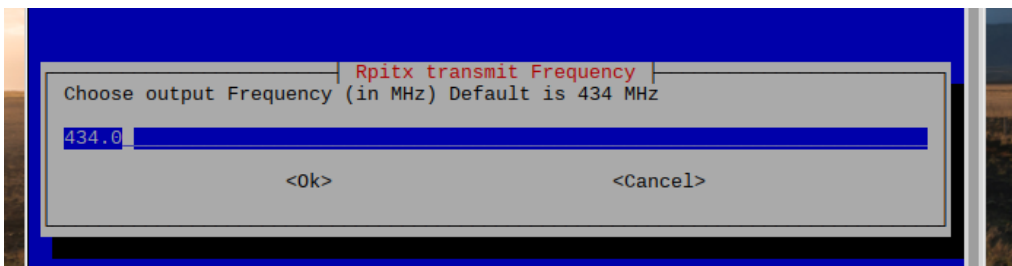
Figura 89. **Easytest RPITX 1**



Fuente: elaboración propia, empleando consola Raspberry pi.

Una vez ejecutado el comando anterior, se abrirá una ventana en la cual se debe ingresar la frecuencia en la que se desea trabajar.

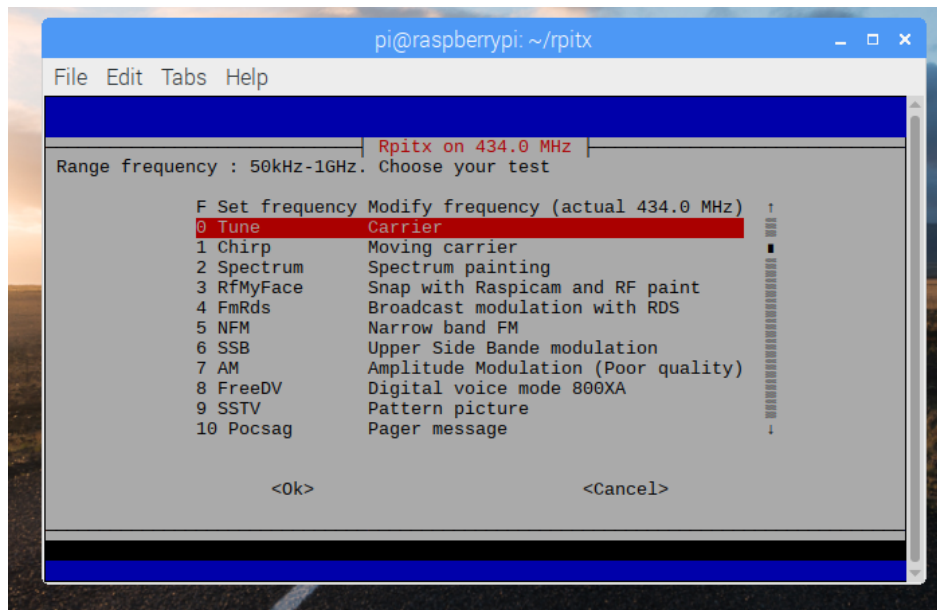
Figura 90. **Easytest RPITX 2**



Fuente: elaboración propia, empleando consola Raspberry pi.

A continuación, será desplegado un menú de opciones, el cual permite escoger el tipo de prueba o modulación con la que se desea trabajar; se debe sintonizar con el módulo USB y GQRX para la validación de su funcionamiento.

Figura 91. **Easytest RPITX 3**

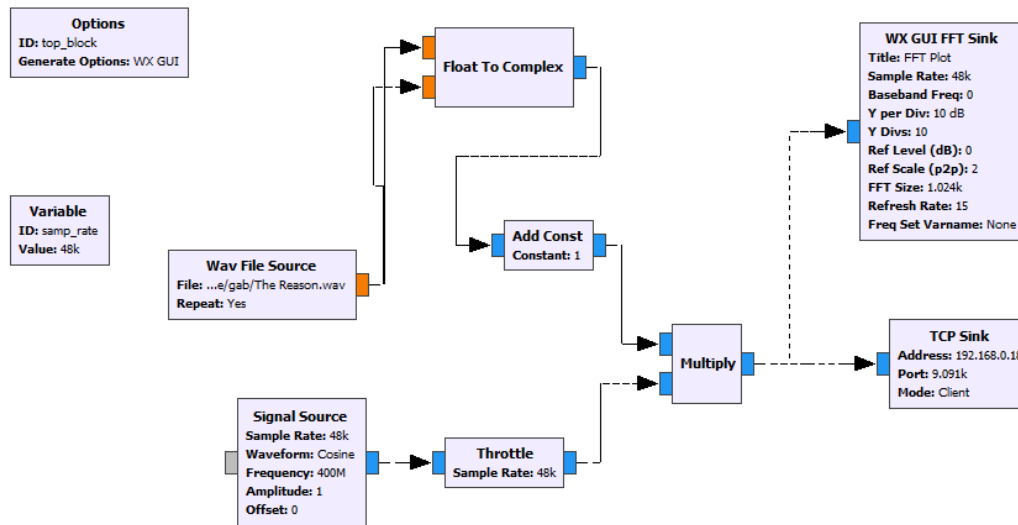


Fuente: elaboración propia, empleando consola Rasberry pi.

4.2. **Práctica 2: amplitud modulada, transmisión y recepción GNU Radio**

La segunda práctica aplicable al laboratorio de Comunicaciones 1 consiste en la realización de un modulador/transmisor y un receptor/demodulador, trabajando con la modulación en amplitud.

Figura 92. Diagrama de bloques transmisor amplitud modulada



Fuente: elaboración propia, empleando GNU Radio.

4.2.1. Transmisor amplitud modulada

Los bloques utilizados para trabajar la modulación en amplitud, así como la transmisión de una pista de audio deben ser los siguientes:

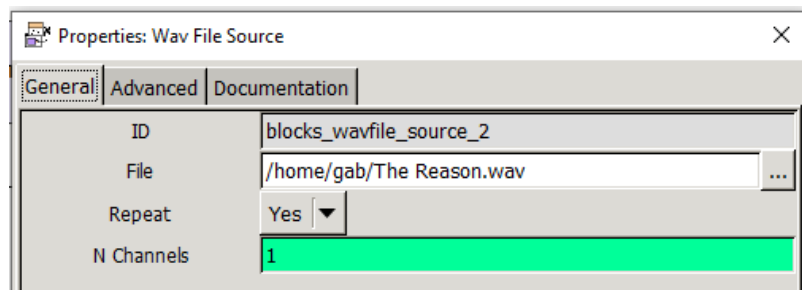
- *Wav file source*
- *Float to complex*
- *Add constant*
- *Signal source*
- *Trottle*
- *Multiply*
- *TCP sink*
- *WX GUI FFT sink*
- *Variable*

A continuación, se explicará la función de los bloques, así como los parámetros de cada uno de ellos.

4.2.1.1. Wav file source transmisor AM

El bloque “*Wav file source*” es el encargado de cargar el archivo de audio utilizado como señal moduladora; los parámetros a configurar son los siguientes:

Figura 93. Parámetros *wav file source* transmisor AM

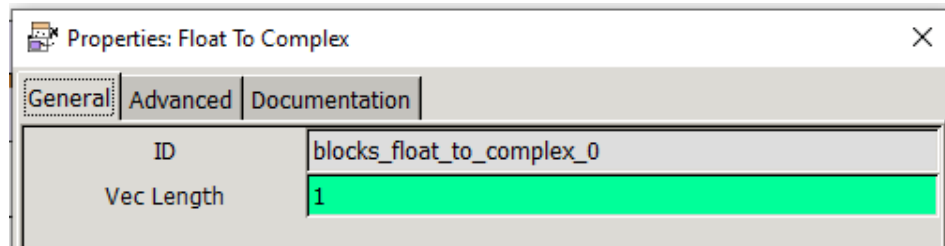


Fuente: elaboración propia, empleando GNU radio.

4.2.1.2. Float to complex transmisor AM

Este bloque permite la conversión de los datos tipo flotante de la pista de audio a tipo complejo, necesario para la compatibilidad con los demás bloques a utilizar; los parámetros son los siguientes:

Figura 94. **Parámetros *float to complex* transmisor AM**

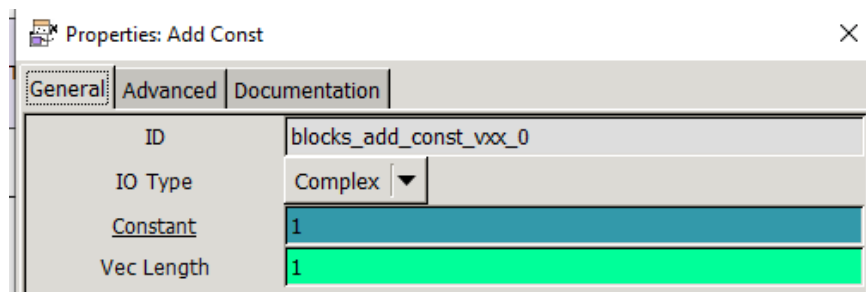


Fuente: elaboración propia, empleando GNU Radio.

4.2.1.3. **Add constant transmisor AM**

El bloque *Add constant* es utilizado para agregar una componente DC a la señal moduladora (pista de audio), necesaria para trabajar con la modulación en amplitud; los parámetros de este bloque son los siguientes:

Figura 95. **Parámetros *Add constant* transmisor AM**

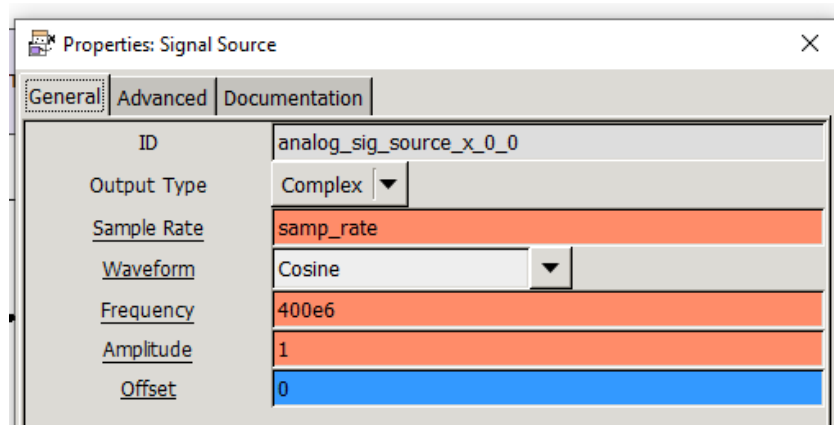


Fuente: elaboración propia, empleando GNU Radio.

4.2.1.4. *Signal source* transmisor AM

El bloque *Signal source* será utilizado para la creación de una onda cosenoidal, la cual funcionará como onda portadora con una frecuencia de 400 MHz; los parámetros del bloque *Signal source* deben ser los siguientes:

Figura 96. **Parámetros *signal source* transmisor AM**

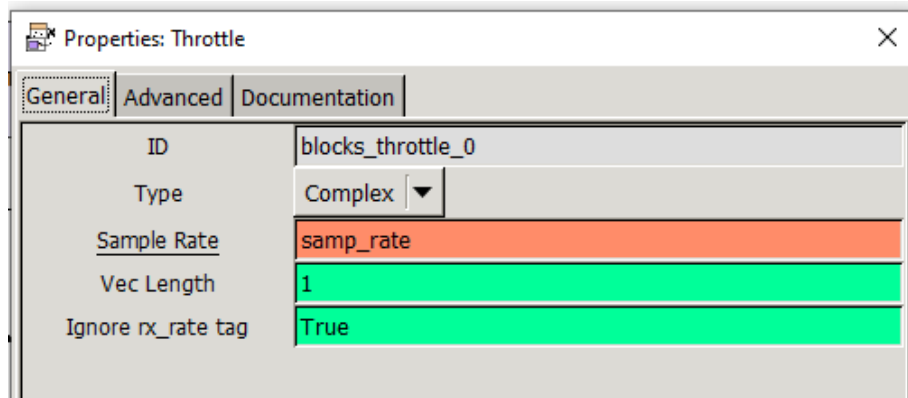


Fuente: elaboración propia, empleando GNU Radio.

4.2.1.5. *Trottle* transmisor AM

El bloque "*Trottle*" se encarga de regular el flujo de datos, manteniéndolo constante a su salida; los parámetros deben ser los siguientes:

Figura 97. **Parámetros *trottle* transmisor AM**

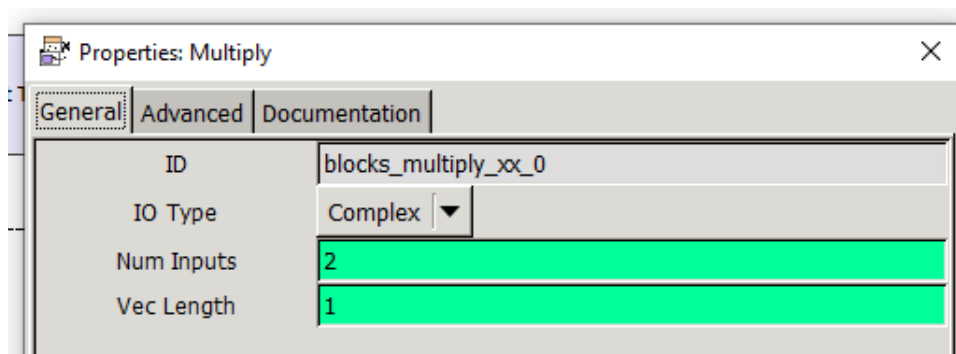


Fuente: elaboración propia, empleando GNU Radio.

4.2.1.6. ***Multiply* transmisor AM**

El multiplicador será utilizado para efectuar el producto entre la señal moduladora y la portadora; los parámetros deben ser los siguientes:

Figura 98. **Parámetros *multiply* transmisor AM**

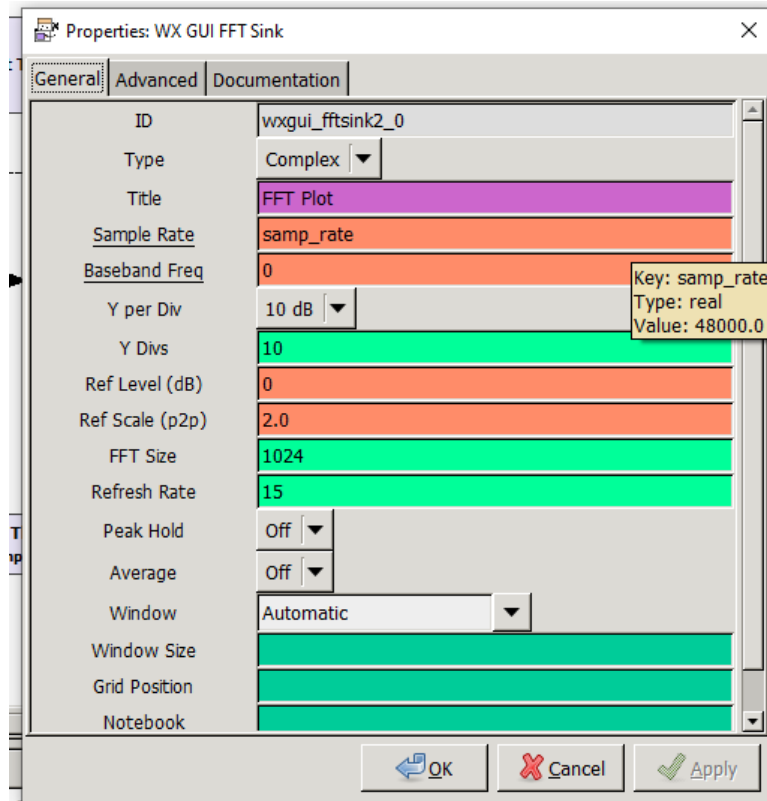


Fuente: elaboración propia, empleando GNU Radio.

4.2.1.7. WX GUI FFT *sink* transmisor AM

El bloque WX GUI FFT *sink* es utilizado para desplegar la gráfica de la transformada rápida de Fourier de la onda modulada; los parámetros deben ser los siguientes:

Figura 99. Parámetros WX GUI FFT *sink* transmisor AM

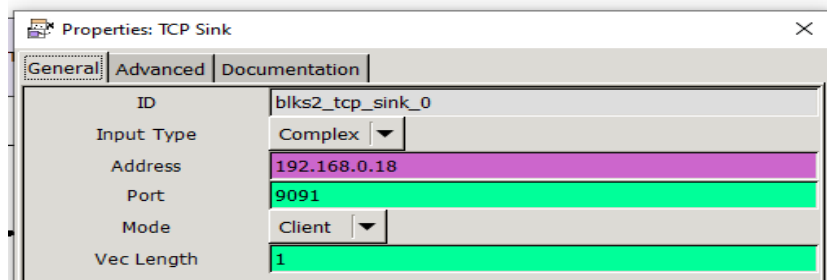


Fuente: elaboración propia, empleando GNU Radio.

4.2.1.8. TCP *sink* transmisor AM

El bloque TCP *sink* será utilizado para la comunicación Raspberry; es necesario colocar la dirección ip de la Raspberry y el puerto que se utilizará con la librería RPITX para la transmisión; los parámetros deben estar de la siguiente manera:

Figura 100. **Parámetros TCP *sink* transmisor AM**

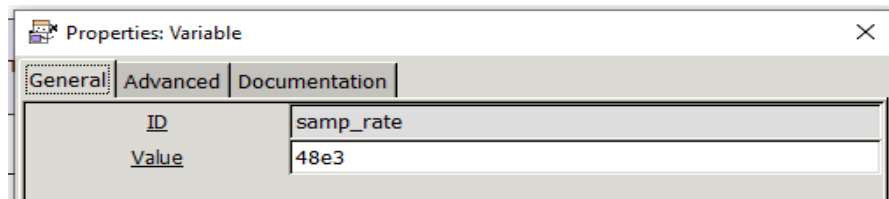


Fuente: elaboración propia, empleando GNU Radio.

4.2.1.9. Variable transmisor AM

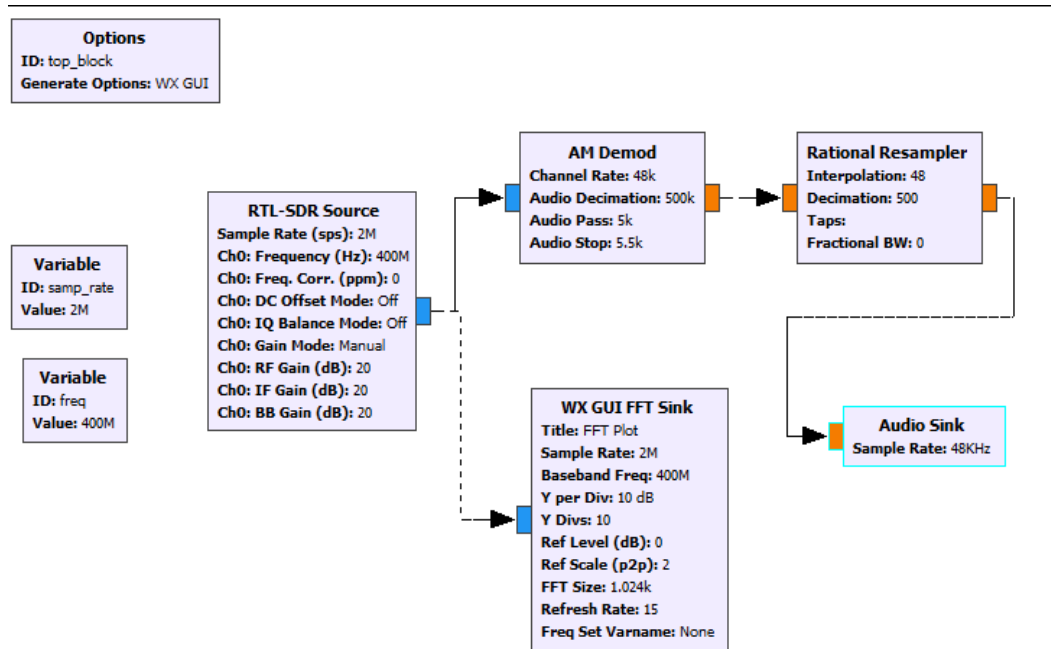
Se utiliza una variable para especificar la frecuencia de muestreo; los parámetros serán los siguientes:

Figura 101. **Parámetros variable transmisor AM**



Fuente: elaboración propia, empleando GNU Radio.

Figura 102. Diagrama de bloques de receptor modulación en amplitud



Fuente: elaboración propia, empleando GNU Radio.

4.2.2. Receptor amplitud modulada

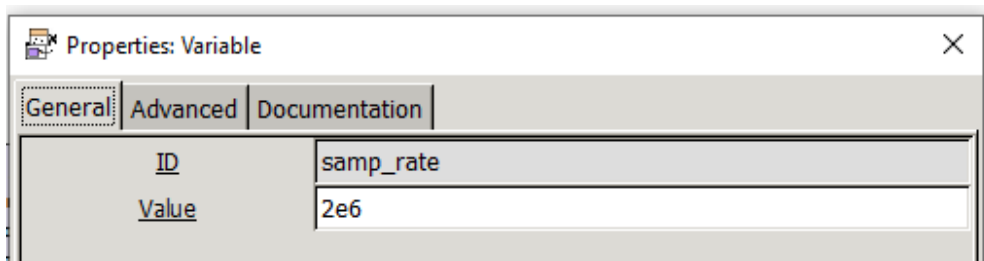
Los bloques utilizados para trabajar la recepción y demodulación en amplitud deben ser los siguientes:

- Variables
- RTL-SDR *source*
- AM *demod*
- Rational *resampler*
- Audio *sink*
- WX GUI FFT *sink*

4.2.2.1. Variables receptor AM

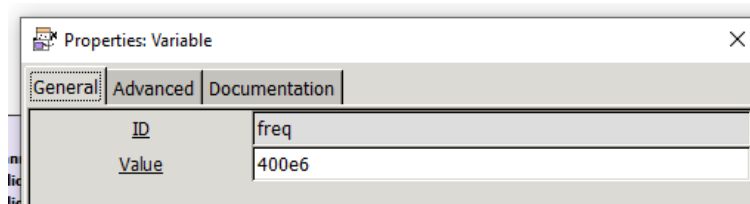
Se utilizarán dos variables para la recepción; la primera con la frecuencia de muestreo y la segunda, con la frecuencia que se sintonizará:

Figura 103. **Parámetros variable 1, receptor AM**



Fuente: elaboración propia, empleando GNU Radio.

Figura 104. **Parámetros variable 2, receptor AM**



Fuente: elaboración propia, empleando GNU Radio.

4.2.2.2. RTL-SDR *source* receptor AM

El bloque RTL-SDR *source* es utilizado para habilitar el módulo USB basado en el chip RTL2832u; los parámetros que deben emplearse en este bloque son los siguientes:

Figura 105. **Parámetros RTL-SDR source receptor AM**

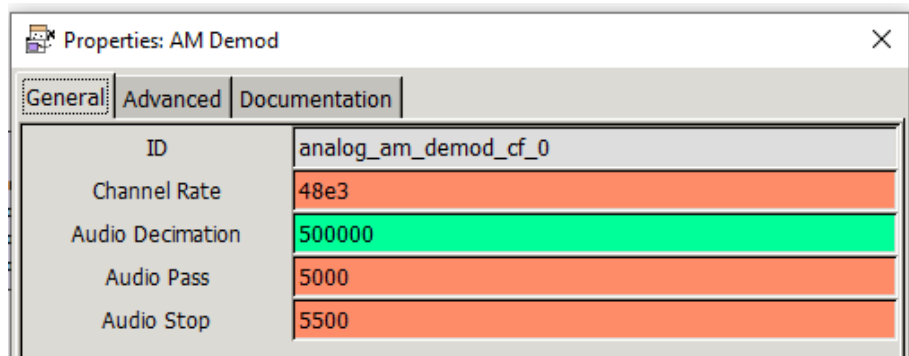
<u>ID</u>	rtlsdr_source_0
Output Type	Complex float32 ▾
Device Arguments	
Sync	don't sync ▾
Num Mboards	1 ▾
Mb0: Clock Source	Default ▾
Mb0: Time Source	Default ▾
Num Channels	1 ▾
<u>Sample Rate (sps)</u>	samp_rate
<u>Ch0: Frequency (Hz)</u>	freq
<u>Ch0: Freq. Corr. (ppm)</u>	0
<u>Ch0: DC Offset Mode</u>	Off ▾
<u>Ch0: IQ Balance Mode</u>	Off ▾
<u>Ch0: Gain Mode</u>	Manual ▾
<u>Ch0: RF Gain (dB)</u>	20
<u>Ch0: IF Gain (dB)</u>	20
<u>Ch0: BB Gain (dB)</u>	20
<u>Ch0: Antenna</u>	
<u>Ch0: Bandwidth (Hz)</u>	0

Fuente: elaboración propia, empleando GNU Radio.

4.2.2.3. **AM demod receptor AM**

El bloque AM *demod*, es el encargado de realizar la demodulación de la señal captada por el bloque RTL-SDR *source*; los parámetros a utilizar deben ser los siguientes:

Figura 106. **Parámetros AM demod receptor AM**

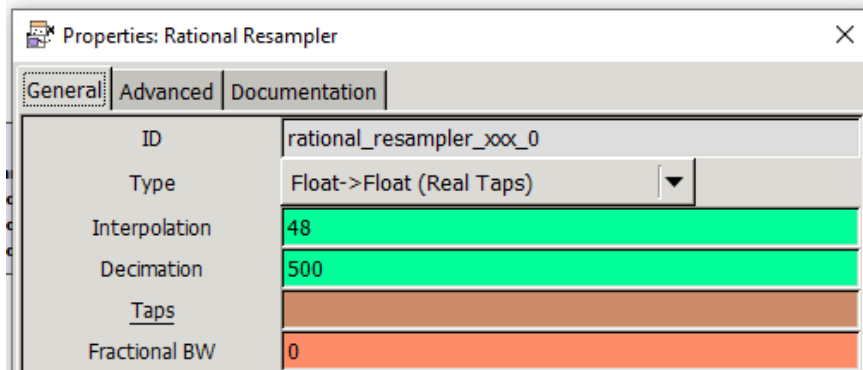


Fuente: elaboración propia, empleando GNU Radio.

4.2.2.4. Rational resampler receptor AM

Para la utilización del bloque *audio sink*, es necesario acoplar la señal a la frecuencia de muestreo con la que trabaja la tarjeta de audio de la computadora, siendo el bloque *rational resampler* el encargado de trabajar dicha modificación; para esto se deben colocar los siguientes parámetros:

Figura 107. **Parámetros rational resampler receptor AM**

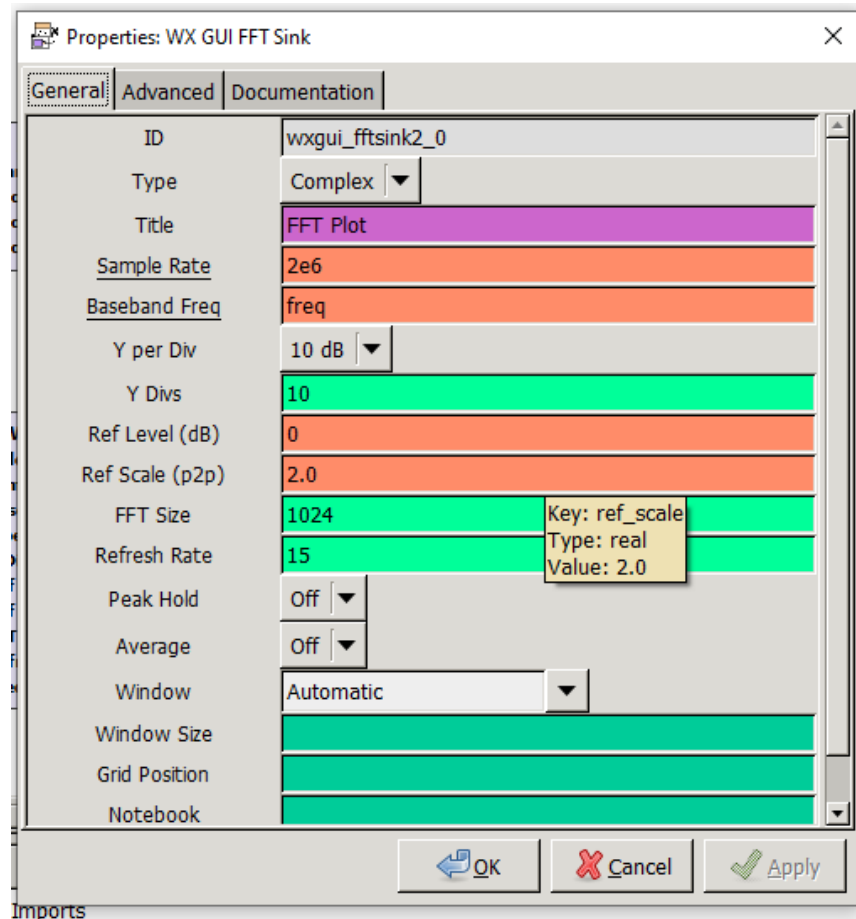


Fuente: elaboración propia, empleando GNU Radio.

4.2.2.5. WX GUI FFT *sink* receptor AM

El bloque WX GUI FFT *sink* es utilizado para desplegar la gráfica de la transformada rápida de Fourier de la señal recibida por el módulo USB; los parámetros deben ser los siguientes:

Figura 108. **Parámetros WX GUI FFT *sink* receptor AM**

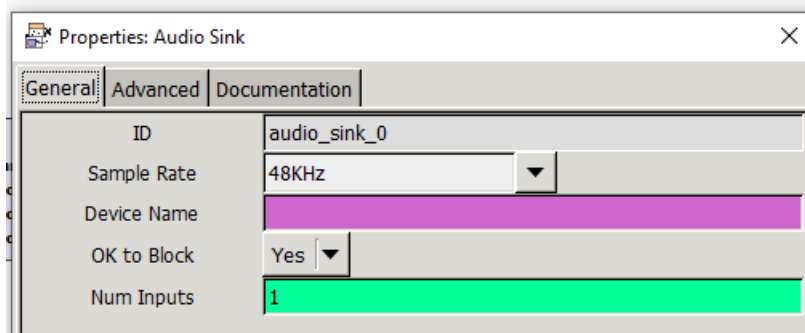


Fuente: elaboración propia, empleando GNU Radio.

4.2.2.6. *Audio sink* receptor AM

El bloque *Audio sink* es utilizado para habilitar la tarjeta de sonido de la computadora como dispositivo de salida, y poder escuchar la pista de audio que se está recibiendo; para esto se deben colocar los siguientes parámetros:

Figura 109. **Parámetros *Audio sink* receptor AM**

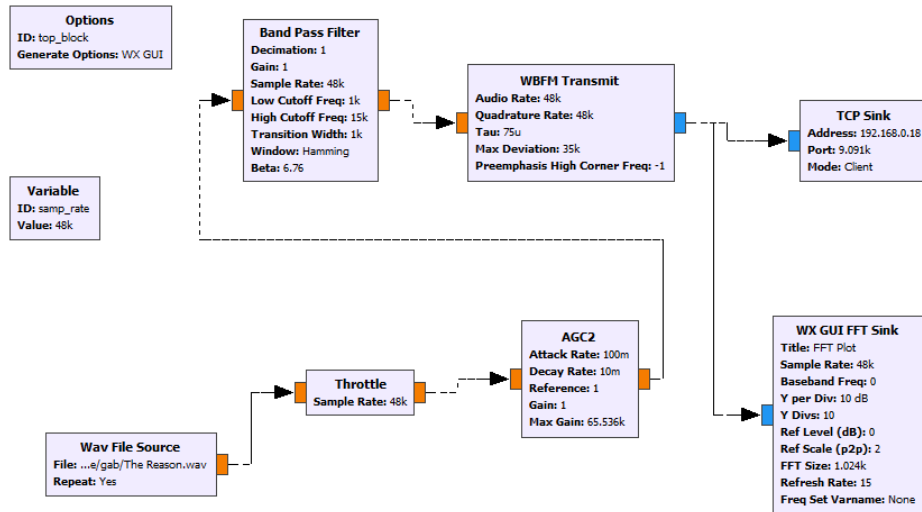


Fuente: elaboración propia, empleando GNU Radio.

4.3. **Práctica 3: frecuencia modulada, transmisión y recepción GNU Radio**

La tercera práctica aplicable al laboratorio de Comunicaciones 1, consiste en la realización de un modulador/transmisor y un receptor/demodulador, trabajando con la modulación en frecuencia.

Figura 110. Diagrama de bloques transmisor WBFM



Fuente: elaboración propia, empleando GNU Radio.

4.3.1. Transmisor frecuencia modulada banda ancha

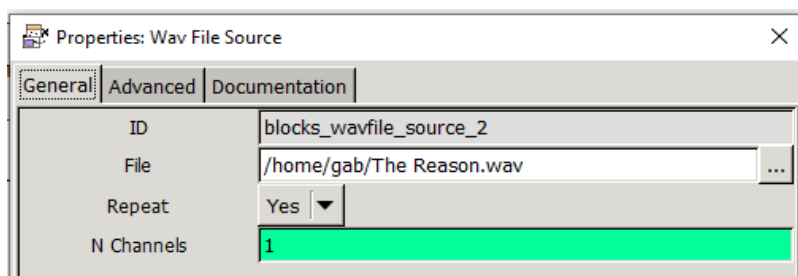
Los bloques utilizados para trabajar la modulación en frecuencia de banda ancha y su transmisión son los siguientes:

- *Wav file source*
- *Trottle*
- *AGC2*
- *Band pass filter*
- *WBFM transmit*
- *TCP sink*
- *WX GUI FFT sink*
- *Variable*

4.3.1.1. *Wav file source* transmisor WBFM

El bloque *Wav file source* es el encargado de cargar el archivo de audio utilizado como señal moduladora; los parámetros a configurar son los siguientes:

Figura 111. **Parámetros *wav file source* transmisor WBFM**

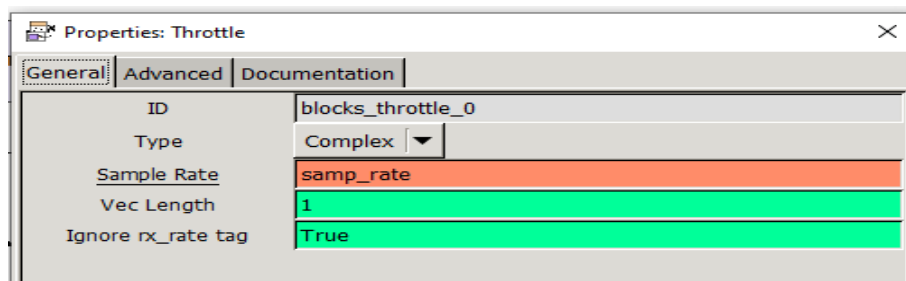


Fuente: elaboración propia, empleando GNU Radio.

4.3.1.2. *Trottle* transmisor WBFM

El bloque *Trottle* se encarga de regular el flujo de datos manteniéndolo constante a su salida; los parámetros deben ser los siguientes:

Figura 112. **Parámetros *Trottle* transmisor WBFM**

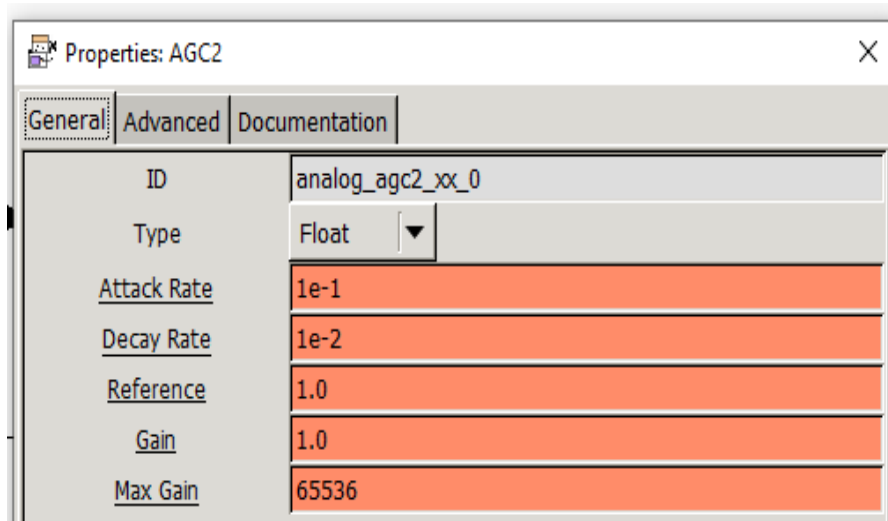


Fuente: elaboración propia, empleando GNU Radio.

4.3.1.3. AGC2 transmisor WBFM

Control automático de ganancia, bloque encargado de variar la ganancia de la señal recibida; los parámetros deben ser los siguientes:

Figura 113. Parámetros AGC2 transmisor WBFM

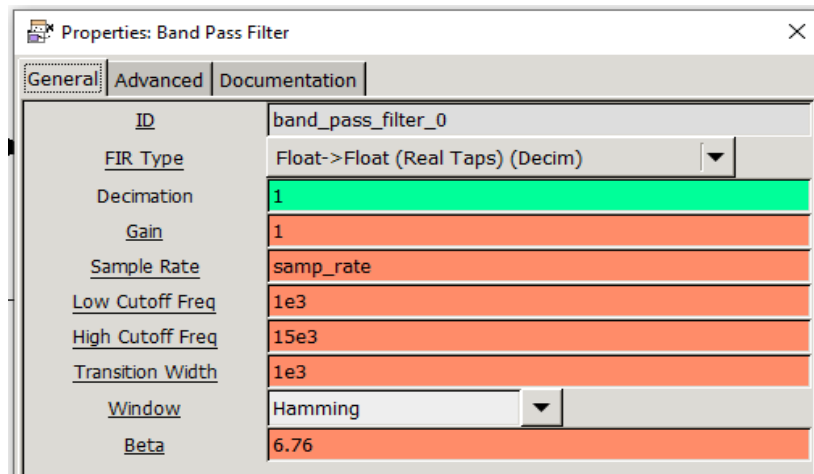


Fuente: elaboración propia, empleando GNU Radio.

4.3.1.4. Band pass filter transmisor WBFM

El filtro pasabanda debe tener los siguientes parámetros:

Figura 114. **Parámetros *band pass filter* transmisor WBFM**

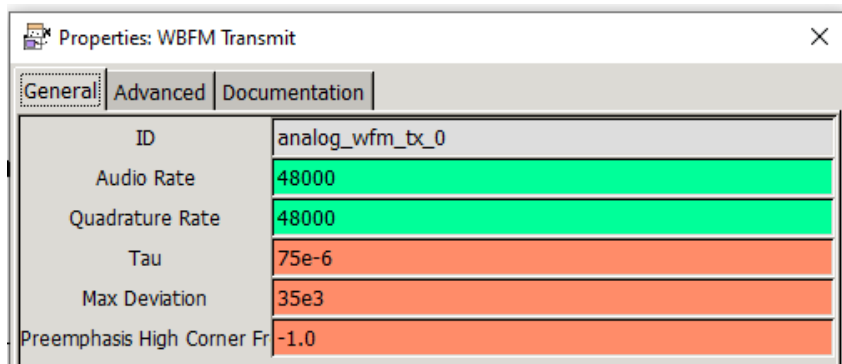


Fuente: elaboración propia, empleando GNU Radio.

4.3.1.5. **WBFM *transmit* transmisor WBFM**

Bloque encargado de realizar la modulación en frecuencia de banda ancha; los parámetros deben ser los siguientes:

Figura 115. **Parámetros WBFM *transmit* transmisor WBFM**

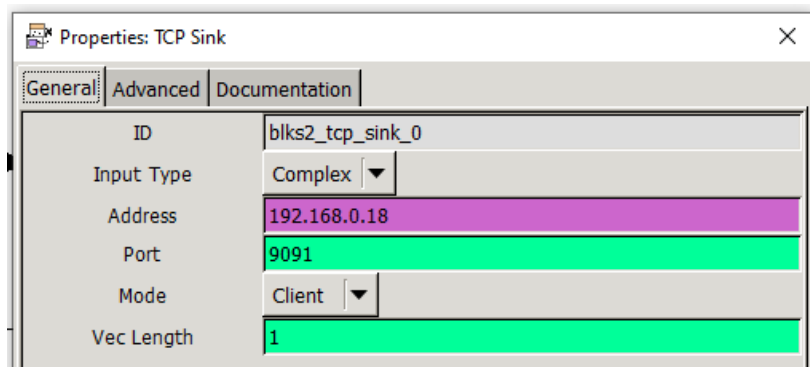


Fuente: elaboración propia, empleando GNU Radio.

4.3.1.6. TCP *sink* transmisor WBFM

El bloque TCP *sink*, será utilizado para la comunicación la Rasperry; es necesario colocar la dirección ip de la Rasperry y el puerto que se utilizará con la librería RPITX para la transmisión; los parámetros deben estar de la siguiente manera:

Figura 116. Parámetros TCP *sink* transmisor WBFM

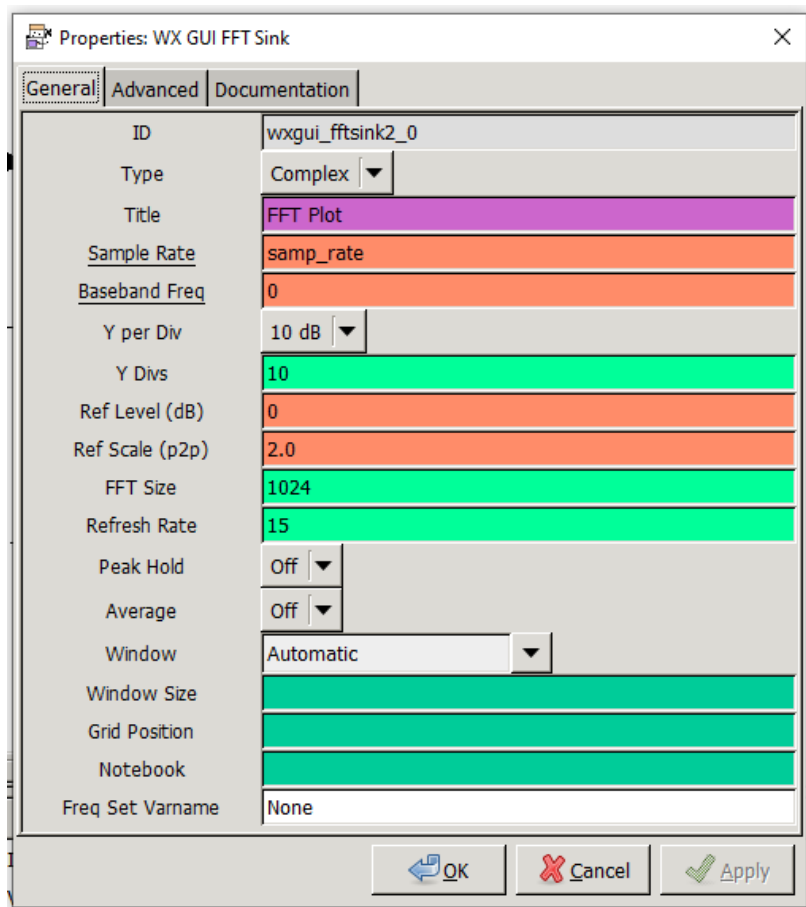


Fuente: elaboración propia, empleando GNU Radio.

4.3.1.7. WX GUI FFT *sink* transmisor WBFM

El bloque WX GUI FFT *sink* es utilizado para desplegar la gráfica de la transformada rápida de Fourier de la señal recibida por el módulo USB; los parámetros deben ser los siguientes:

Figura 117. **Parámetros WX GUI FFT *sink* transmisor WBFM**

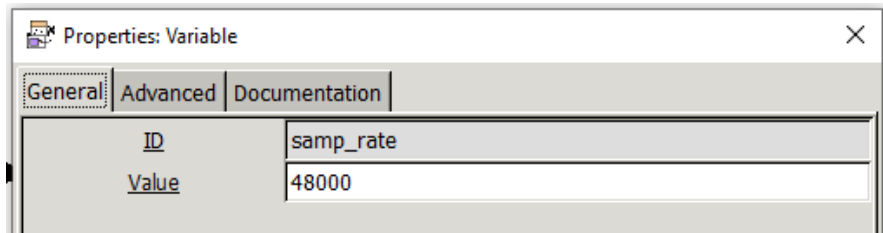


Fuente: elaboración propia, empleando GNU Radio.

4.3.1.8. **Variable transmisor WBFM**

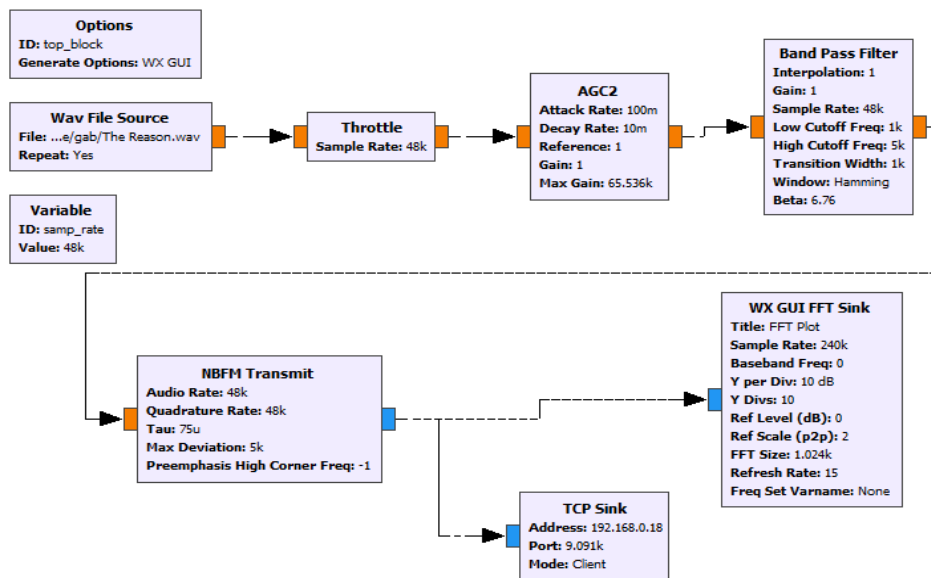
Variable utilizada para establecer la frecuencia de muestreo del sistema; los parámetros deben ser los siguientes:

Figura 118. **Parámetro variable transmisor WBFM**



Fuente: elaboración propia, empleando GNU Radio.

Figura 119. **Diagrama de bloques transmisor NBFM**



Fuente: elaboración propia, empleando GNU Radio.

4.3.2. Transmisor frecuencia modulada banda angosta

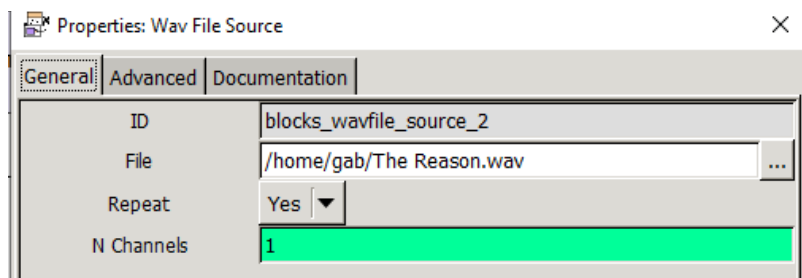
Los bloques utilizados para trabajar la modulación en frecuencia de banda angosta y su transmisión son los siguientes:

- *Wav file source*
- *Trottle*
- *AGC2*
- *Band Pass Filter*
- *NBFM Transmit*
- *TCP Sink*
- *WX GUI FFT Sink*
- *Variable*

4.3.2.1. *Wav file source* transmisor NBFM

El bloque *Wav file source* es el encargado de cargar el archivo de audio, utilizado la señal moduladora; los parámetros a configurar son los siguientes:

Figura 120. **Parámetros *wav file source* transmisor NBFM**

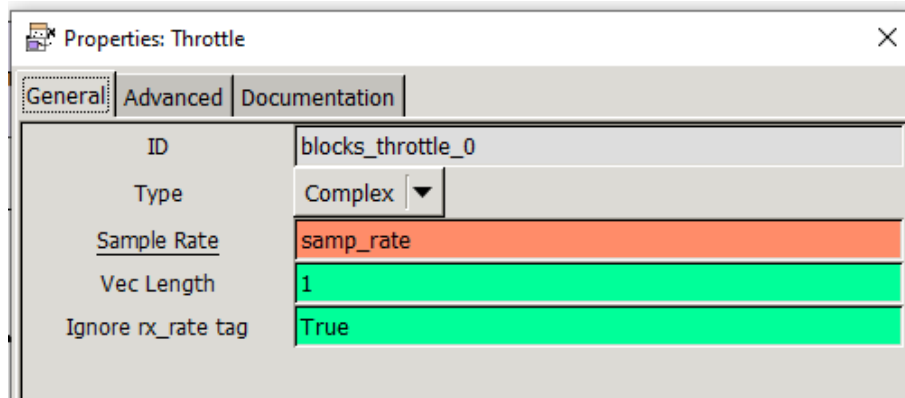


Fuente: elaboración propia, empleando GNU Radio.

4.3.2.2. *Trottle* transmisor NBFM

El bloque *Trottle* se encarga de regular el flujo de datos, manteniéndolo constante a su salida; los parámetros deben ser los siguientes:

Figura 121. **Parámetros *trottle* transmisor NBFM**

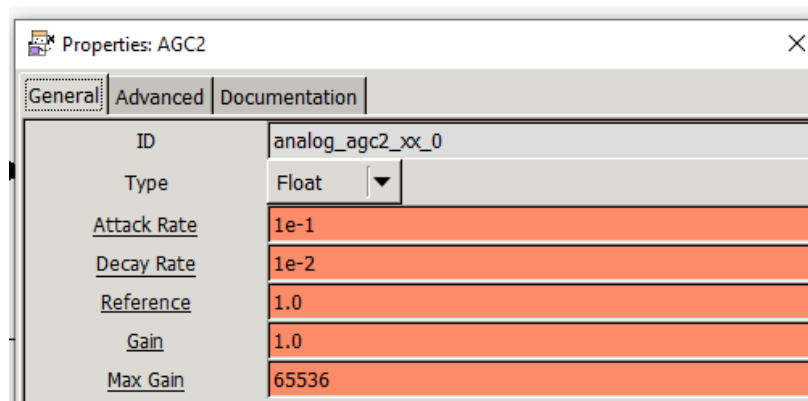


Fuente: elaboración propia, empleando GNU Radio.

4.3.2.3. **AGC2 transmisor NBFM**

Control automático de ganancia, bloque encargado de variar la ganancia de la señal recibida; los parámetros deben ser los siguientes:

Figura 122. **Parámetros AGC2 transmisor NBFM**

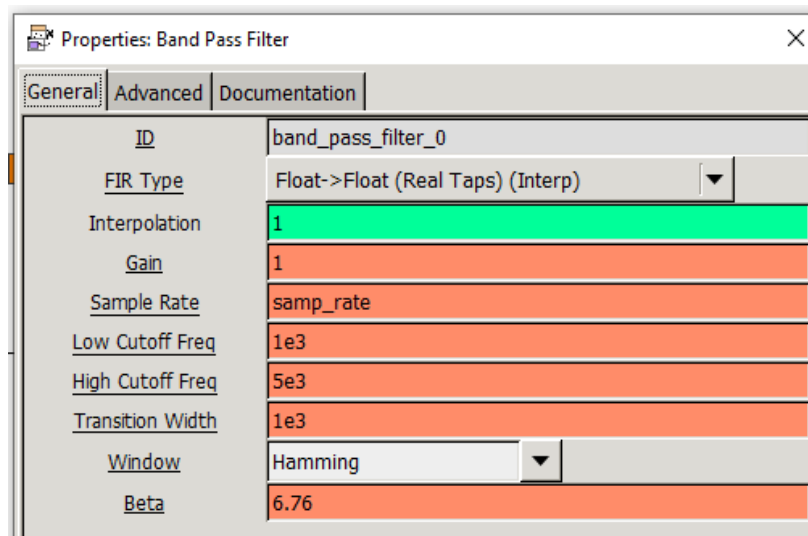


Fuente: elaboración propia, empleando GNU Radio.

4.3.2.4. *Band pass filter* transmisor NBFM

El filtro pasabanda debe tener los siguientes parámetros:

Figura 123. **Parámetros *band pass filter* transmisor NBFM**

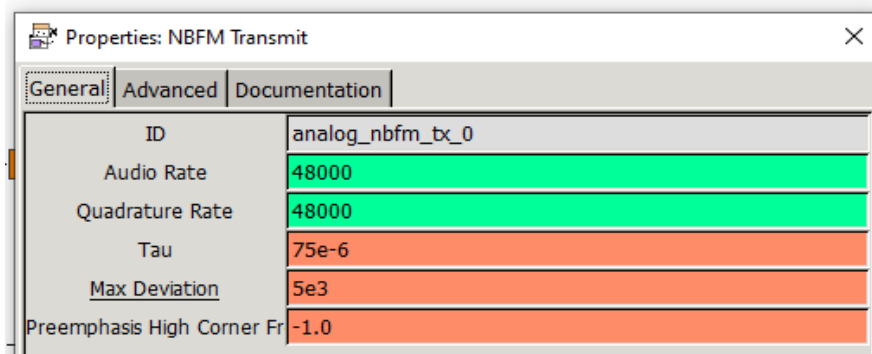


Fuente: elaboración propia, empleando GNU Radio.

4.3.2.5. *NBFM transmit* transmisor NBFM

Bloque encargado de realizar la modulación en frecuencia de banda angosta; los parámetros deben ser los siguientes:

Figura 124. **Parámetros NBFM *transmit* transmisor NBFM**

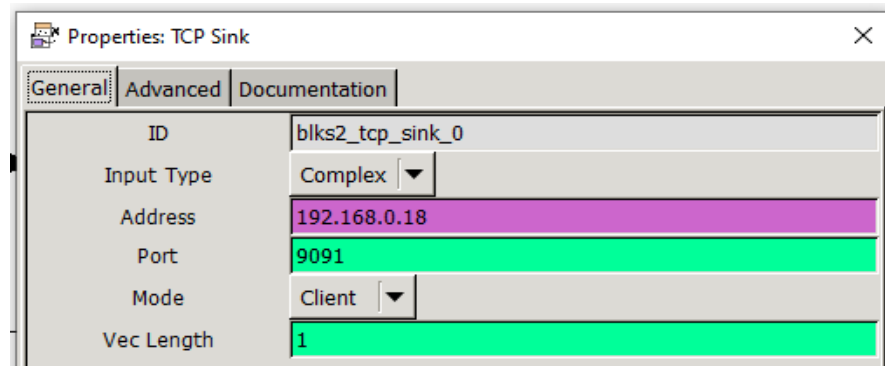


Fuente: elaboración propia, empleando GNU Radio.

4.3.2.6. **TCP *sink* transmisor NBFM**

El bloque TCP *sink*, será utilizado para la comunicación la Raspberry, es necesario colocar la dirección ip de la Raspberry y el puerto que se utilizará con la librería RPITX para la transmisión; los parámetros deben estar de la siguiente manera:

Figura 125. **Parámetros TCP *sink* transmisor NBFM**

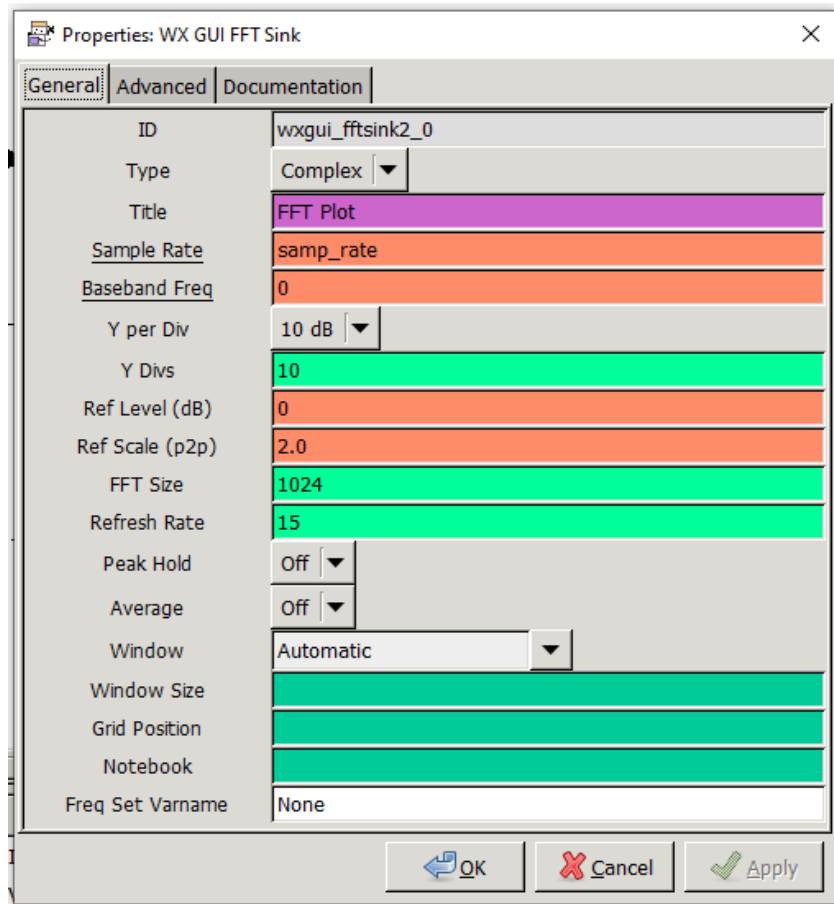


Fuente: elaboración propia, empleando GNU Radio.

4.3.2.7. WX GUI FFT *sink* transmisor NBFM

El bloque WX GUI FFT *sink* es utilizado para desplegar la gráfica de la transformada rápida de Fourier de la señal recibida por el módulo USB; los parámetros deben ser los siguientes:

Figura 126. Parámetros WX GUI FFT *sink* transmisor NBFM

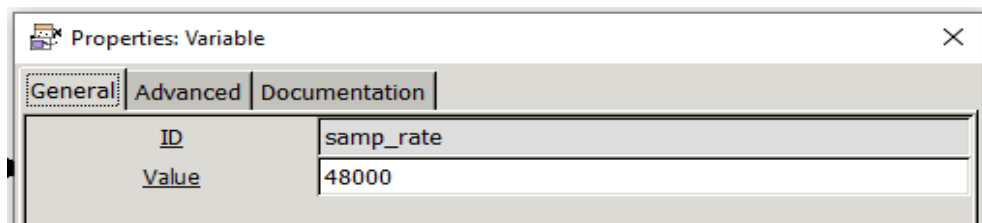


Fuente: elaboración propia, empleando GNU Radio.

4.3.2.8. Variable transmisor NBFM

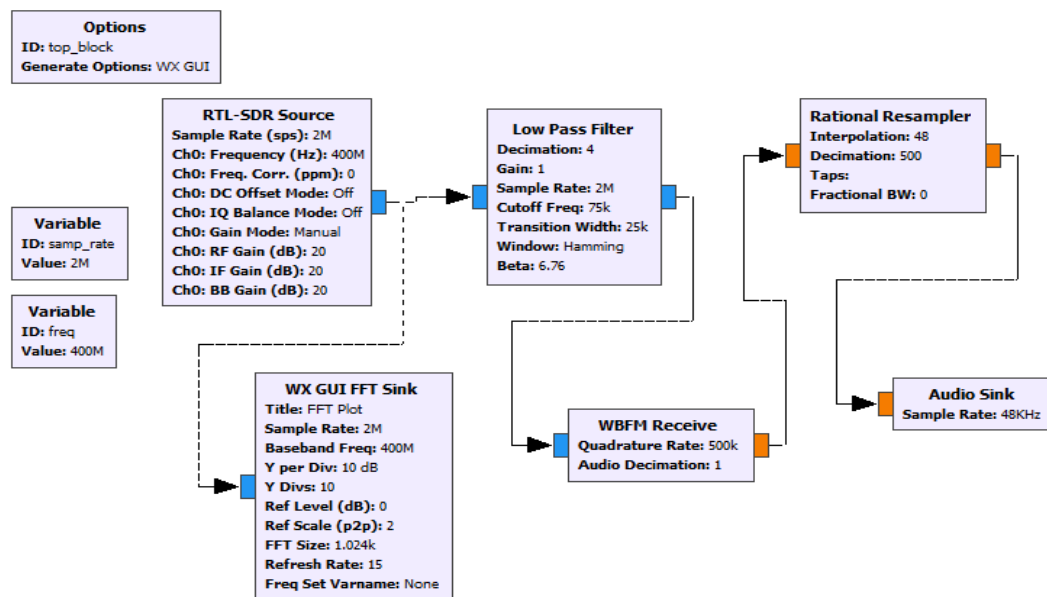
Variable utilizada para establecer la frecuencia de muestreo del sistema; los parámetros deben ser los siguientes:

Figura 127. Parámetros variable transmisor NBFM



Fuente: elaboración propia, empleando GNU Radio.

Figura 128. Diagrama de bloques receptor FM



Fuente: elaboración propia, empleando GNU Radio.

4.3.3. Receptor frecuencia modulada

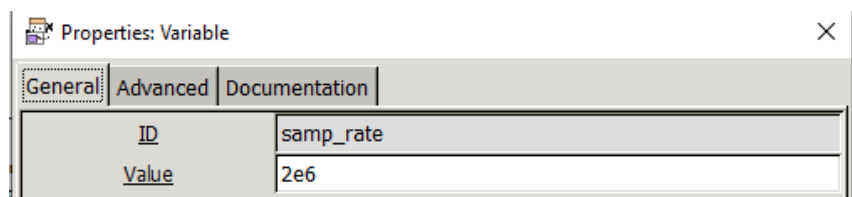
Los bloques utilizados para trabajar la recepción y demodulación en frecuencia, tanto para banda ancha como para banda angosta, deben ser los siguientes:

- *Variables*
- *RTL-SDR source*
- *Low pass filter*
- *WBFM demod*
- *Rational resampler*
- *Audio sink*
- *WX GUI FFT sink*

4.3.3.1. Variables receptor FM

Se utilizarán dos variables para la recepción: la primera con la frecuencia de muestreo y la segunda con la frecuencia que se sintonizará:

Figura 129. **Parámetros variable 1 receptor FM**



Fuente: elaboración propia, empleando GNU Radio.

Figura 130. **Parámetros variable 2 receptor FM**

General		Advanced	Documentation
ID	freq		
Value	400e6		

Fuente: elaboración propia, empleando GNU Radio.

4.3.3.2. RTL-SDR *source* receptor FM

El bloque RTL-SDR *source* es utilizado para habilitar el módulo USB basado en el chip RTL2832u; los parámetros a utilizar en este bloque son los siguientes:

Figura 131. **Parámetros RTL-SDR *source* receptor AM**

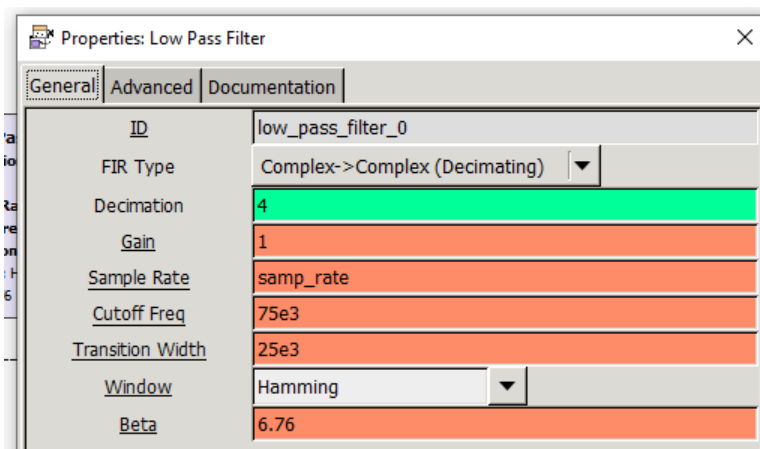
ID	rtlsdr_source_0		
Output Type	Complex float32		
Device Arguments			
Sync	don't sync		
Num Mboards	1		
Mb0: Clock Source	Default		
Mb0: Time Source	Default		
Num Channels	1		
Sample Rate (sps)	samp_rate		
Ch0: Frequency (Hz)	freq		
Ch0: Freq. Corr. (ppm)	0		
Ch0: DC Offset Mode	Off		
Ch0: IQ Balance Mode	Off		
Ch0: Gain Mode	Manual		
Ch0: RF Gain (dB)	20		
Ch0: IF Gain (dB)	20		
Ch0: BB Gain (dB)	20		
Ch0: Antenna			
Ch0: Bandwidth (Hz)	0		

Fuente: elaboración propia, empleando GNU Radio.

4.3.3.3. Low pass filter receptor FM

Filtro pasabajo utilizado para disminuir el ruido en la recepción; los parámetros del bloque deben ser los siguientes:

Figura 132. Parámetros *low pass filter* receptor FM

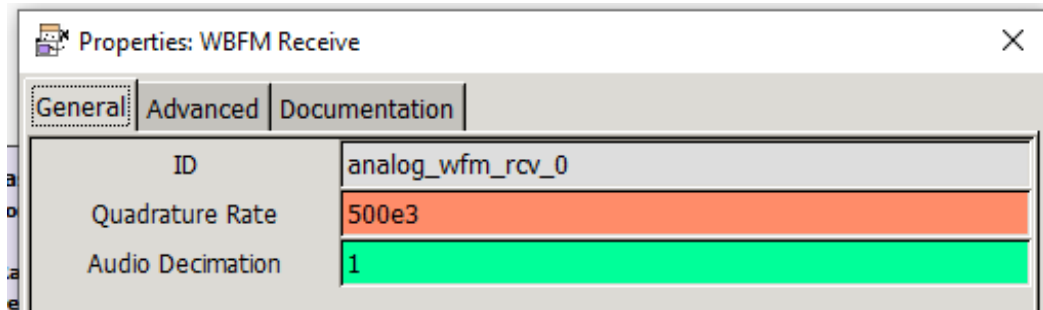


Fuente: elaboración propia, empleando GNU Radio.

4.3.3.4. WBFM *demod* receptor FM

El bloque WBFM *demod*, es el encargado de realizar la demodulación de la señal captada por el bloque RTL-SDR *source*, para esto; los parámetros a utilizar deben ser los siguientes:

Figura 133. **Parámetros WBFM demod receptor FM**

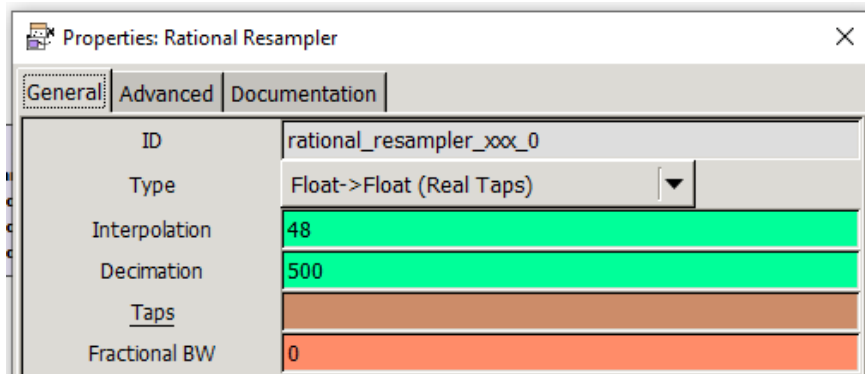


Fuente: elaboración propia, empleando GNU Radio.

4.3.3.5. Rational resampler receptor FM

Para la utilización del bloque *audio sink* es necesario acoplar la señal a la frecuencia de muestreo con la que trabaja la tarjeta de audio de la computadora, siendo el bloque *rational resampler* el encargado de trabajar dicha modificación; para esto se deben colocar los siguientes parámetros:

Figura 134. **Parámetros *rational resampler* receptor FM**

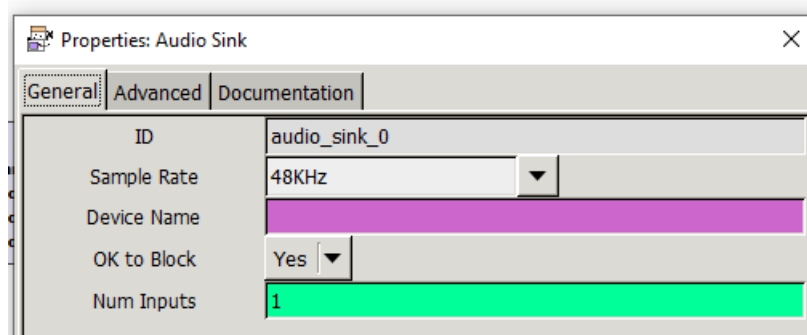


Fuente: elaboración propia, empleando GNU Radio.

4.3.3.6. **Audio sink receptor FM**

El bloque *Audio sink* es utilizado para habilitar la tarjeta de sonido de la computadora como dispositivo de salida y poder escuchar la pista de audio que se está recibiendo; para esto se deben colocar los siguientes parámetros:

Figura 135. **Parámetros *audio sink* receptor FM**

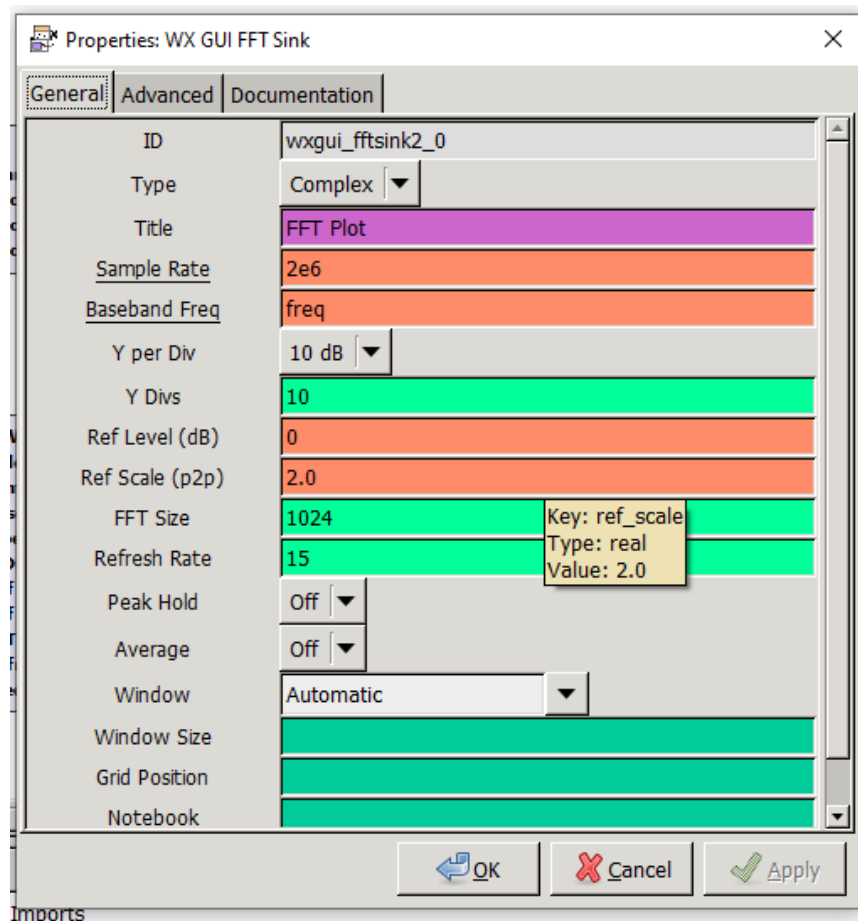


Fuente: elaboración propia, empleando GNU Radio.

4.3.3.7. **WX GUI FFT *sink* receptor FM**

El bloque *WX GUI FFT sink* es utilizado para desplegar la gráfica de la transformada rápida de Fourier de la señal recibida por el módulo USB; los parámetros deben ser los siguientes:

Figura 136. **Parámetros WX GUI FFT *sink* receptor FM**

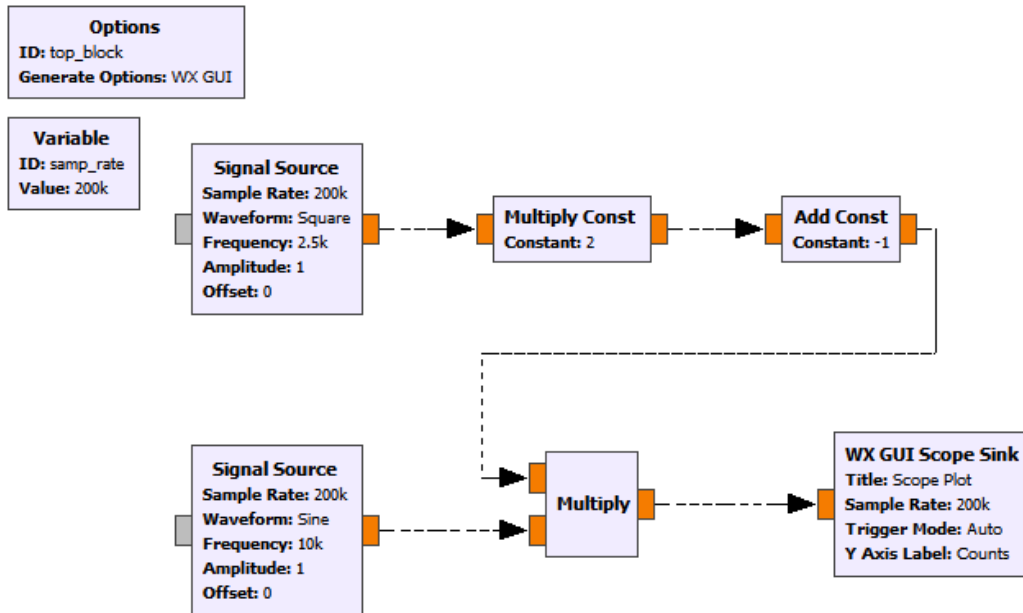


Fuente: elaboración propia, empleando GNU Radio.

4.4. **Práctica 4: modulación BPSK**

La cuarta práctica aplicable al laboratorio de Comunicaciones 1, consiste en trabajar la modulación por desplazamiento de fase binaria, utilizando como señal moduladora una onda cuadrada.

Figura 137. Diagrama de bloques modulador BPSK



Fuente: elaboración propia, empleando GNU Radio.

4.4.1. Modulador BPSK

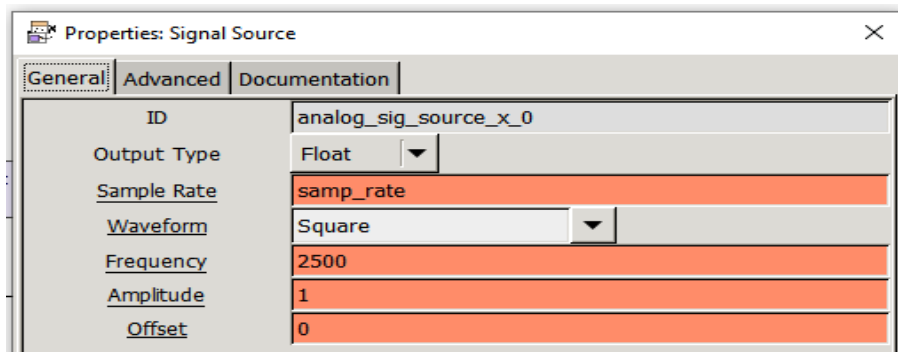
Los bloques utilizados para trabajar la modulación en frecuencia de banda angosta y su transmisión son los siguientes:

- *Signal source*
- *Multiply const*
- *Add const*
- *Multiply*
- *WX scope sink*
- *Variable*

4.4.1.1. *Signal source* modulador BPSK

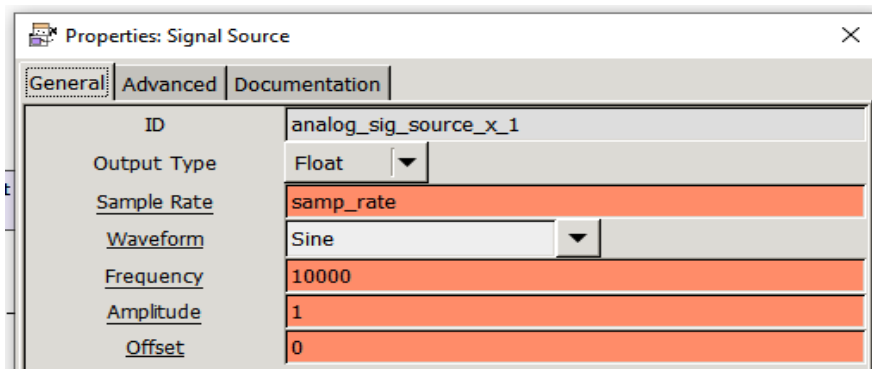
Los bloques *Signal source* serán utilizados para la creación de dos ondas, de las cuales, el primer bloque será utilizado para crear la onda moduladora y el segundo será una onda senoidal que funcionará como onda portadora; los parámetros de los bloques *Signal source* deben ser los siguientes:

Figura 138. **Parámetros *signal source* 1 modulador BPSK**



Fuente: elaboración propia, empleando GNU Radio.

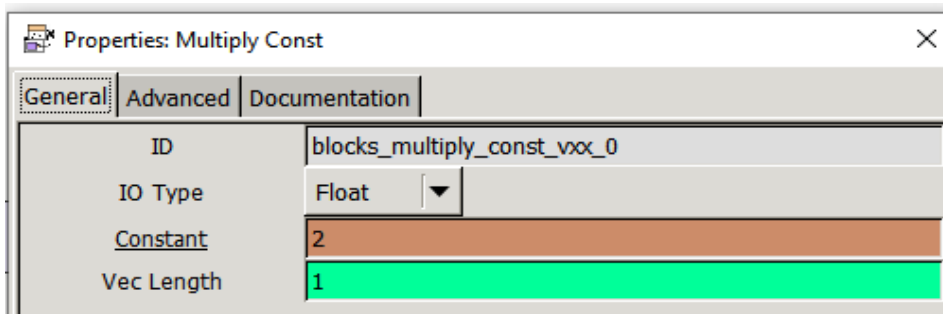
Figura 139. **Parámetros *signal source* 2 modulador BPSK**



Fuente: elaboración propia, empleando GNU Radio.

El bloque *multiply constant* modulador BPSK es utilizado para variar la amplitud de la onda cuadrada utilizada como onda moduladora; los parámetros deben ser los siguientes:

Figura 140. **Parámetros *multiply constant* modulador BPSK**

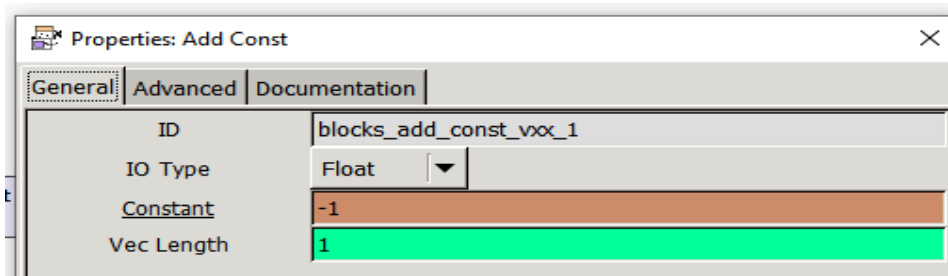


Fuente: elaboración propia, empleando GNU Radio.

4.4.1.2. **Add constant modulador BPSK**

El bloque *Add constant* es utilizado para desfasar la onda en el eje x; los parámetros de este bloque son los siguientes:

Figura 141. **Parámetros *Add constant* modulador BPSK**

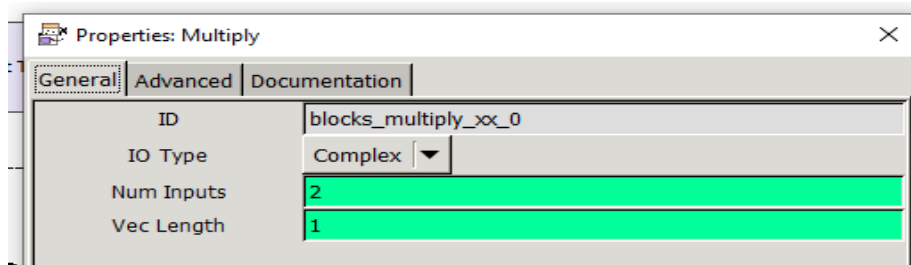


Fuente: elaboración propia, empleando GNU Radio.

4.4.1.3. *Multiply* modulador BPSK

Los multiplicadores serán utilizados para efectuar el producto entre las señales para crear onda la modulada; los parámetros deben ser los siguientes para ambos bloques:

Figura 142. **Parámetros *multiply* modulador BPSK**

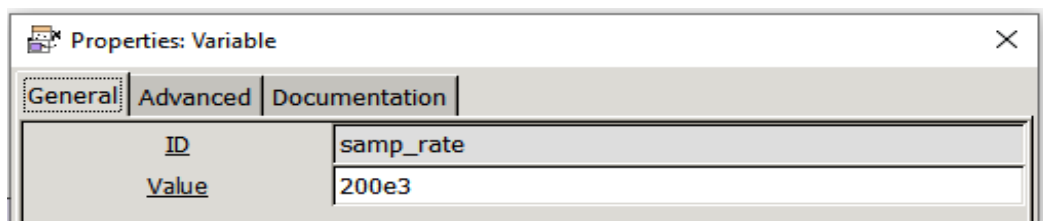


Fuente: elaboración propia, empleando GNU Radio.

4.4.1.4. *Variable* modulador BPSK

Variable utilizada para establecer la frecuencia de muestreo del sistema; los parámetros deben ser los siguientes:

Figura 143. **Parámetros *variable* modulador BPSK**

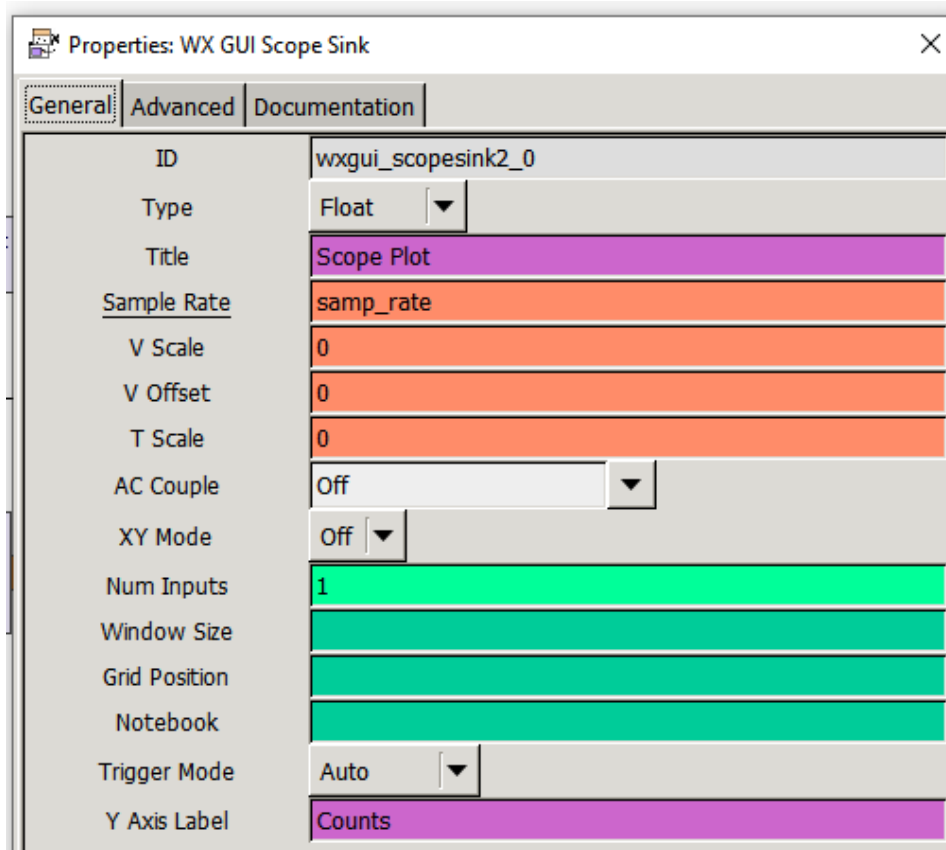


Fuente: elaboración propia, empleando GNU Radio.

4.4.1.5. WX GUI *scope sink* modulador BPSK

Bloque utilizado para la visualización de la forma de onda de la señal modulada y poder observar las variaciones en la fase; los parámetros deben ser los siguientes:

Figura 144. Parámetros WX GUI *scope sink* modulador BPSK

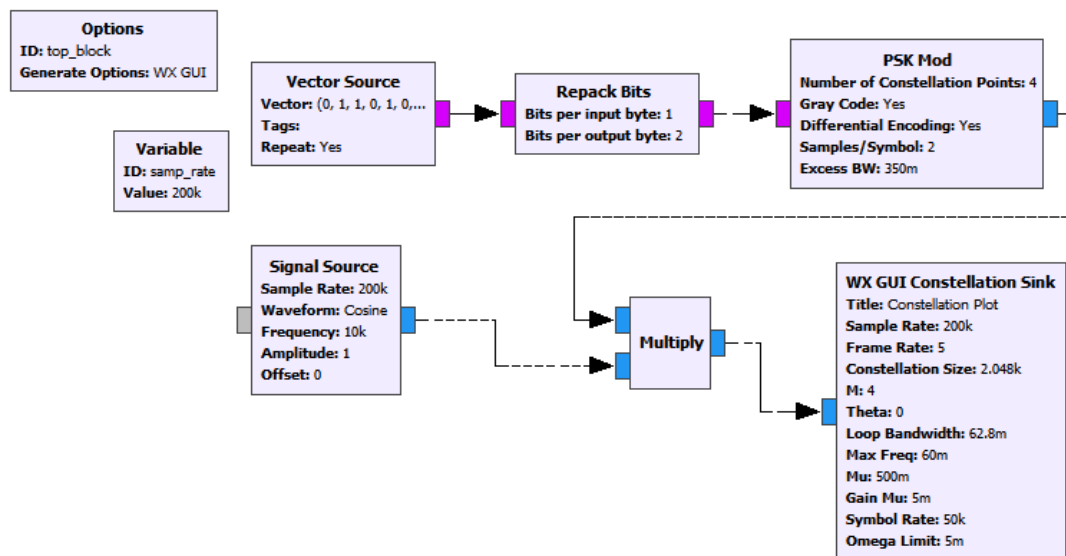


Fuente: elaboración propia, empleando GNU Radio.

4.5. Práctica 5: modulación QPSK

La quinta práctica aplicable al laboratorio de Comunicaciones 1 consiste en trabajar la modulación por desplazamiento de fase en cuadratura, utilizando como señal moduladora un vector de bits.

Figura 145. Diagrama de bloques modulador QPSK



Fuente: elaboración propia, empleando GNU Radio.

4.5.1. Modulación QPSK

Los bloques utilizados para trabajar la modulación por desplazamiento de fase en cuadratura son:

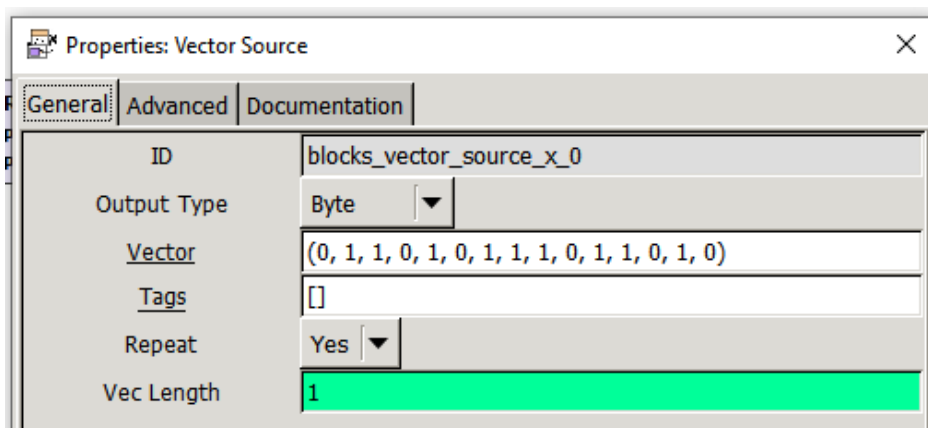
- *Vector source*
- *Repack bits*
- *PSK mod*

- *Signal source*
- *Multiply*
- *WX GUI constellation sink*
- *Variable*

4.5.1.1. **Vector source modulador QPSK**

Bloque utilizado como fuente de información para obtener la señal moduladora; los parámetros del bloque deben ser los siguientes:

Figura 146. **Parámetros vector source modulador QPSK**

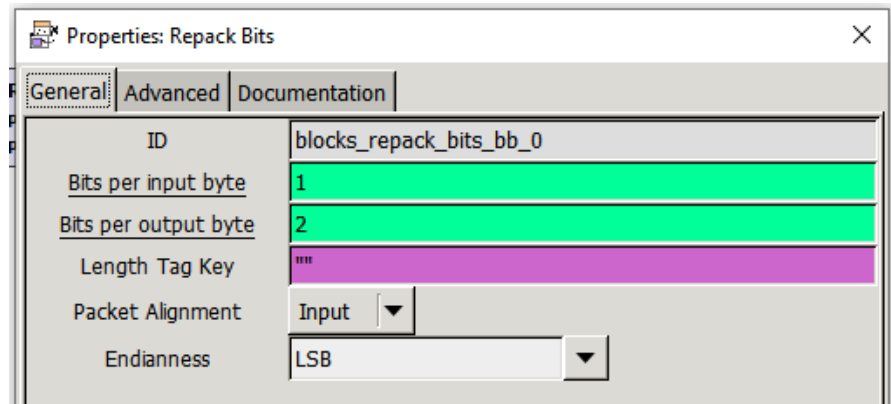


Fuente: elaboración propia, empleando GNU Radio.

4.5.1.2. **Repack bits modulador QPSK**

Bloque utilizado para empaquetar los bits en parejas necesario para la modulación por desplazamiento de fase en cuadratura; los parámetros del bloque deben ser los siguientes:

Figura 147. **Parámetros *repack bits* modulador QPSK**

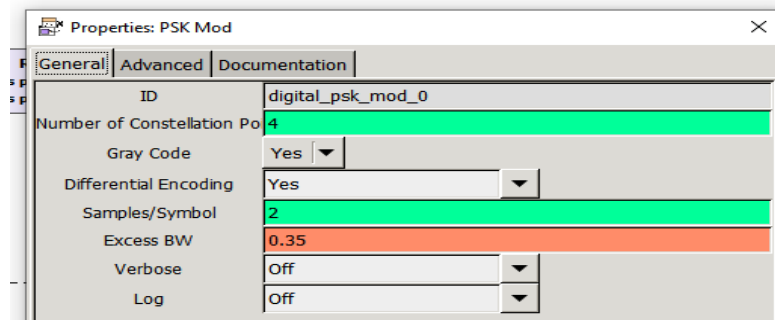


Fuente: elaboración propia, empleando GNU Radio.

4.5.1.3. **PSK *mod* modulador QPSK**

Bloque utilizado para realizar la modulación por desplazamiento de fase, permite efectuar la modulación en BPSK, QPSK y 16PSK, entre otros. Los parámetros para la realización de modulación por desplazamiento de fase en cuadratura deben ser los siguientes:

Figura 148. **Parámetros *PSK mod* modulador QPSK**

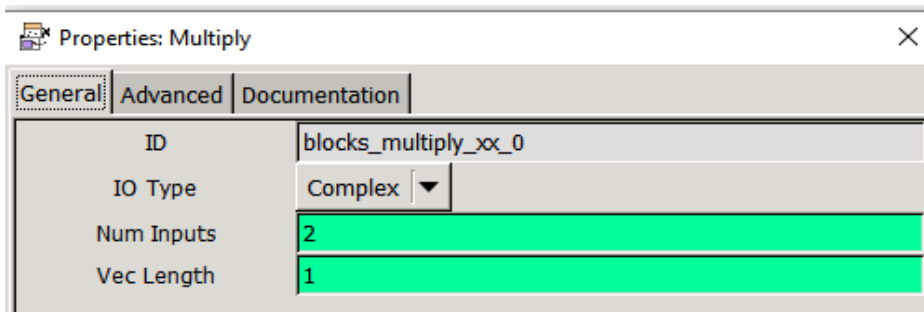


Fuente: elaboración propia, empleando GNU Radio.

4.5.1.4. *Multiply* modulador QPSK

Bloque utilizado para realizar una multiplicación entre señales; los parámetros deben ser los siguientes:

Figura 149. **Parámetros *multiply* modulador QPSK**

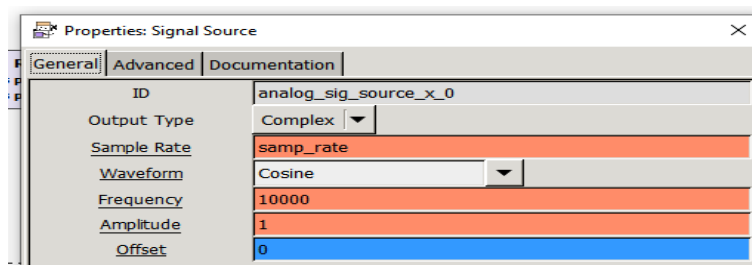


Fuente: elaboración propia, empleando GNU Radio.

4.5.1.5. *Signal source* modulador QPSK

Bloque empleado para la generación de una onda senoidal utilizada como onda portadora; los parámetros deben ser los siguientes:

Figura 150. **Parámetros *signal source* modulador QPSK**

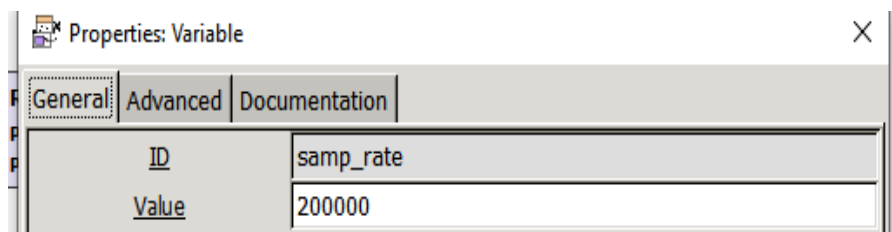


Fuente: elaboración propia, empleando GNU Radio.

4.5.1.6. Variable modulador QPSK

Bloque utilizado para establecer la frecuencia de muestreo del sistema; los parámetros deben ser los siguientes:

Figura 151. Parámetros variable modulador QPSK

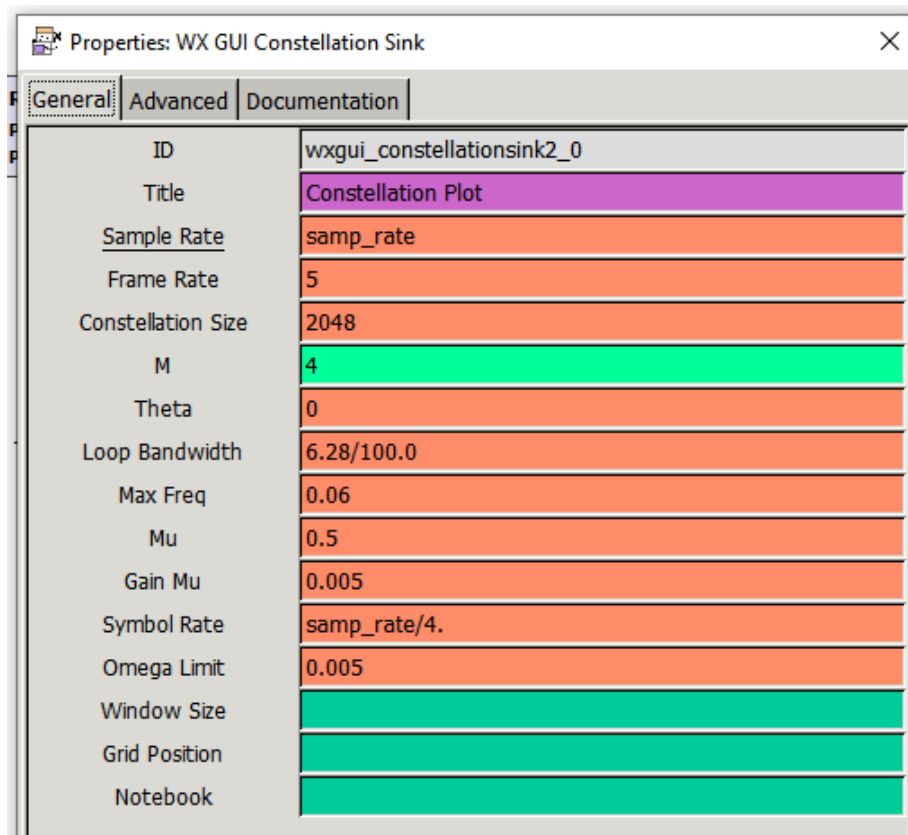


Fuente: elaboración propia, empleando GNU Radio.

4.5.1.7. WX GUI *constellation sink* modulador QPSK

Bloque utilizado para graficar el diagrama de constelaciones de la señal modulada; los parámetros deben ser los siguientes:

Figura 152. **Parámetros WX GUI constellation sink modulador QPSK**



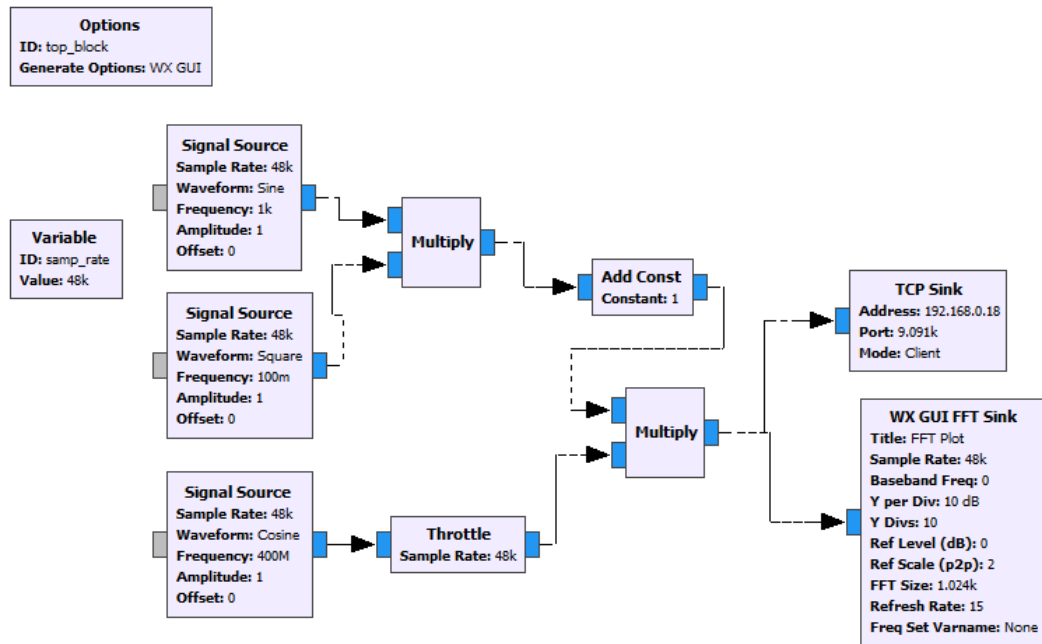
Parameter	Value
ID	wxgui_constellationsink2_0
Title	Constellation Plot
Sample Rate	samp_rate
Frame Rate	5
Constellation Size	2048
M	4
Theta	0
Loop Bandwidth	6.28/100.0
Max Freq	0.06
Mu	0.5
Gain Mu	0.005
Symbol Rate	samp_rate/4.
Omega Limit	0.005
Window Size	
Grid Position	
Notebook	

Fuente: elaboración propia, empleando GNU Radio.

4.6. **Práctica 6: modulación ASK**

La sexta práctica aplicable al laboratorio de Comunicaciones 1 consiste en trabajar la modulación por desplazamiento de amplitud, enfocándose en el modulador y transmisor, y utilizando como receptor uno de AM convencional.

Figura 153. Diagrama de bloques transmisor ASK



Fuente: elaboración propia, empleando GNU Radio.

4.6.1. Modulador ASK

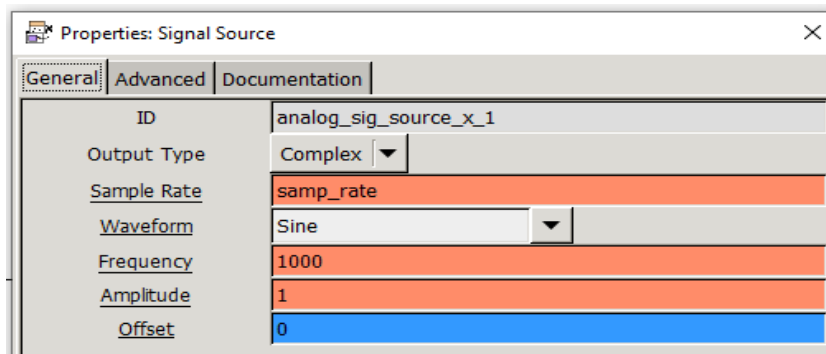
Los bloques utilizados para trabajar la modulación en frecuencia de banda angosta y su transmisión son los siguientes:

- *Signal source*
- *Trottle*
- *Multiply*
- *Add const*
- *TCP sink*
- *WX GUI FFT sink*
- *Variable*

4.6.1.1. *Signal source* transmisor ASK

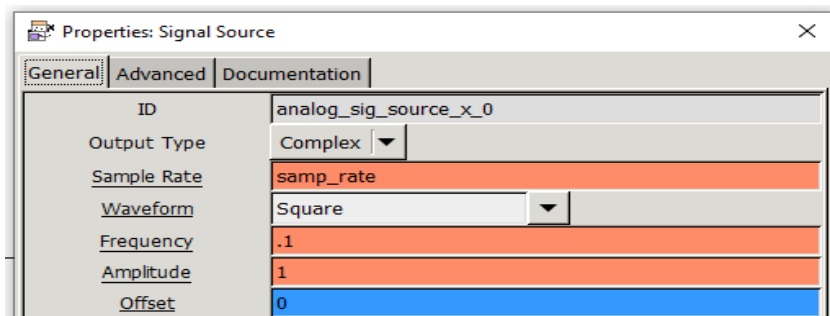
Los bloques *signal source*, serán utilizados para la creación de tres ondas; de los cuales, los bloques 1 y 2 servirán para crear la onda moduladora y el tercer bloque *signal source*, será una onda cosenoidal que funcionará como onda portadora con una frecuencia de 400 MHz: los parámetros de los bloques *signal source* deben ser los siguientes:

Figura 154. **Parámetros *signal source* 1 transmisor ASK**



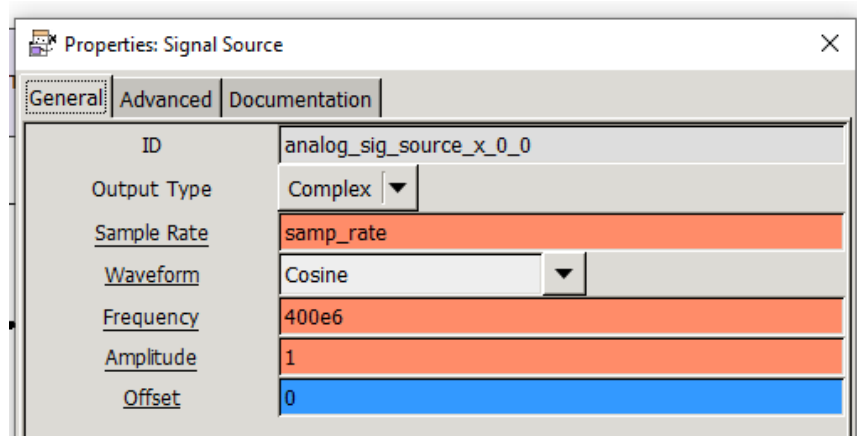
Fuente: elaboración propia, empleando GNU Radio.

Figura 155. **Parámetros *signal source* 2 transmisor ASK**



Fuente: elaboración propia, empleando GNU Radio.

Figura 156. **Parámetros *signal source* 3 transmisor ASK**

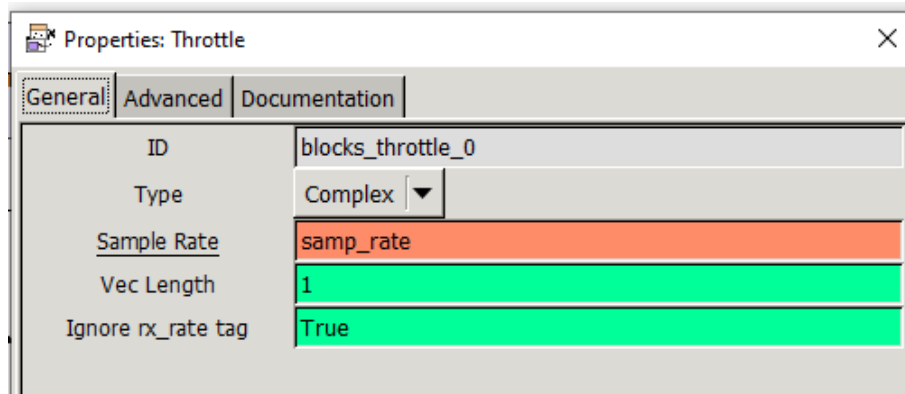


Fuente: elaboración propia, empleando GNU Radio.

4.6.1.2. ***Trottle* transmisor ASK**

El bloque *trottle* se encarga de regular el flujo de datos, manteniéndolo constante a su salida; los parámetros deben ser los siguientes:

Figura 157. **Parámetros *trottle* transmisor ASK**

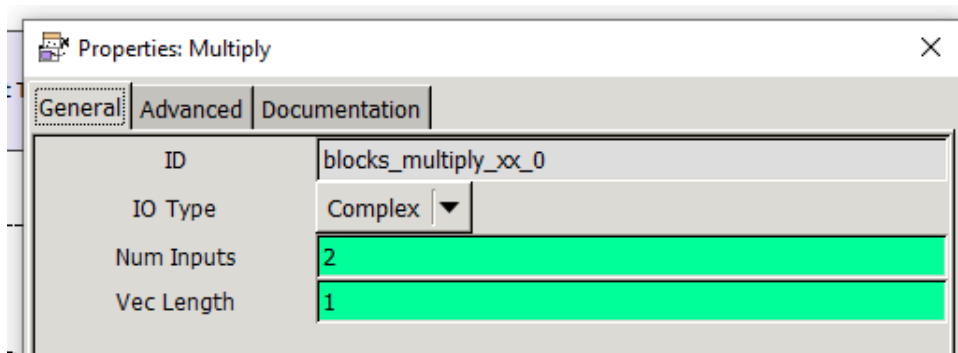


Fuente: elaboración propia, empleando GNU Radio.

4.6.1.3. *Multiply* transmisor ASK

Los multiplicadores serán utilizados para efectuar el producto entre las señales, tanto para crear la señal moduladora como para obtener la señal modulada; los parámetros deben ser los siguientes para ambos bloques:

Figura 158. **Parámetros *multiply* transmisor ASK**

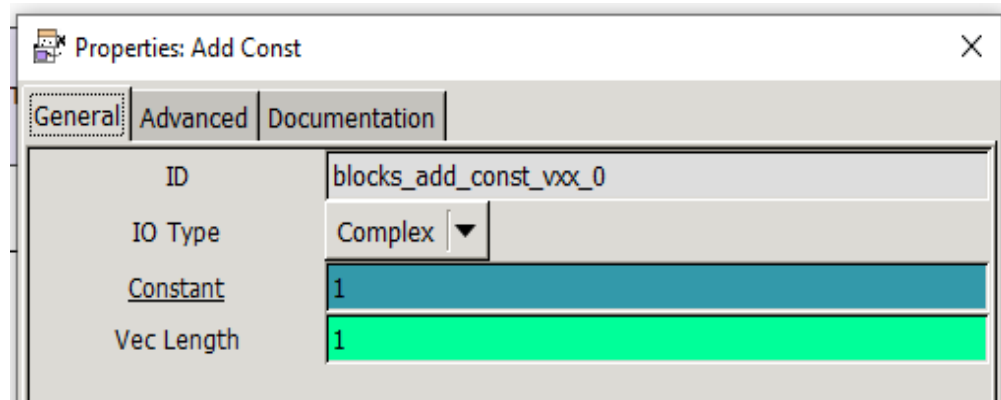


Fuente: elaboración propia, empleando GNU Radio.

4.6.1.4. *Add constant* transmisor ASK

El bloque *Add constant* es utilizado para agregar una componente DC a la señal moduladora (pista de audio) necesaria para trabajar con la modulación en amplitud; los parámetros de este bloque son los siguientes:

Figura 159. **Parámetros *Add constant* transmisor ASK**

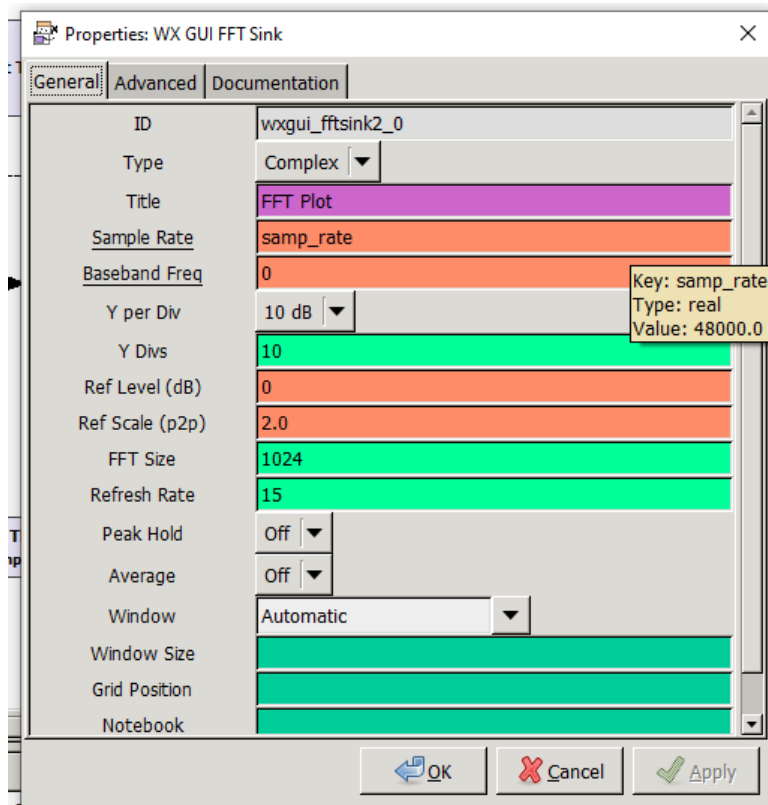


Fuente: elaboración propia, empleando GNU radio.

4.6.1.5. **WX GUI FFT *sink* transmisor ASK**

El bloque WX GUI FFT *sink* es utilizado para desplegar la gráfica de la transformada rápida de Fourier de la onda modulada; los parámetros deben ser los siguientes:

Figura 160. **Parámetros WX GUI FFT *sink* transmisor ASK**

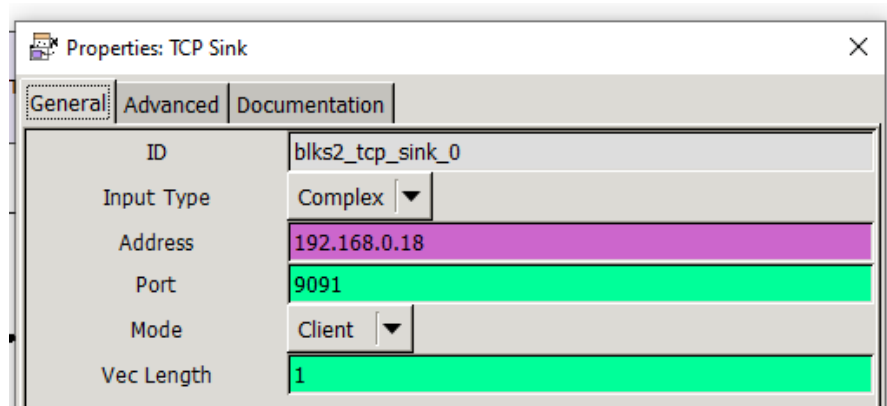


Fuente: elaboración propia, empleando GNU Radio.

4.6.1.6. **TCP *sink* transmisor ASK**

El bloque TCP *sink*, será utilizado para la comunicación Raspberry; es necesario colocar la dirección ip de la Raspberry y el puerto que se utilizará con la librería RPITX para la transmisión; los parámetros deben estar de la siguiente manera:

Figura 161. **Parámetros TCP sink transmisor ASK**

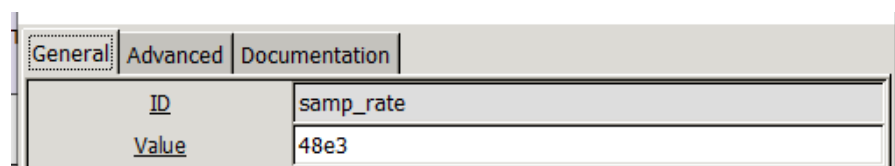


Fuente: elaboración propia, empleando GNU Radio.

4.6.1.7. Variable transmisor ASK

Se utiliza una variable para especificar la frecuencia de muestreo; los parámetros serán los siguientes:

Figura 162. **Parámetros variable transmisor ASK**

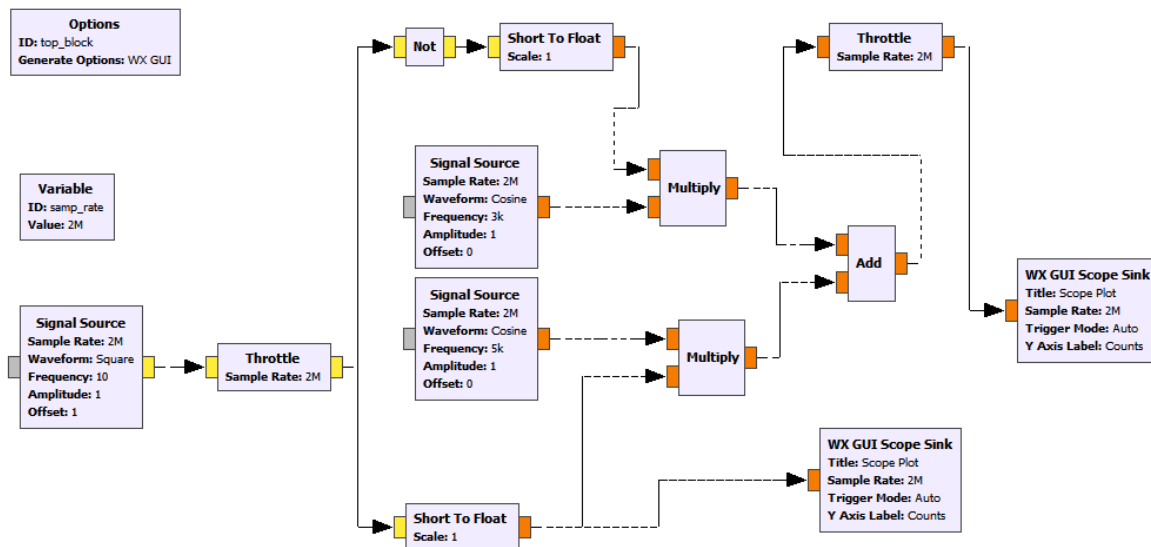


Fuente: elaboración propia, empleando GNU Radio.

4.7. Práctica 7: modulación FSK

La séptima práctica aplicable al laboratorio de Comunicaciones 1 consiste en la realización de un modulador, trabajando la modulación por desplazamiento de frecuencia.

Figura 163. Diagrama de bloques modulador FSK



Fuente: elaboración propia, empleando GNU Radio.

4.7.1. Modulador FSK

Los bloques utilizados para trabajar la modulación por desplazamiento de frecuencia son los siguientes:

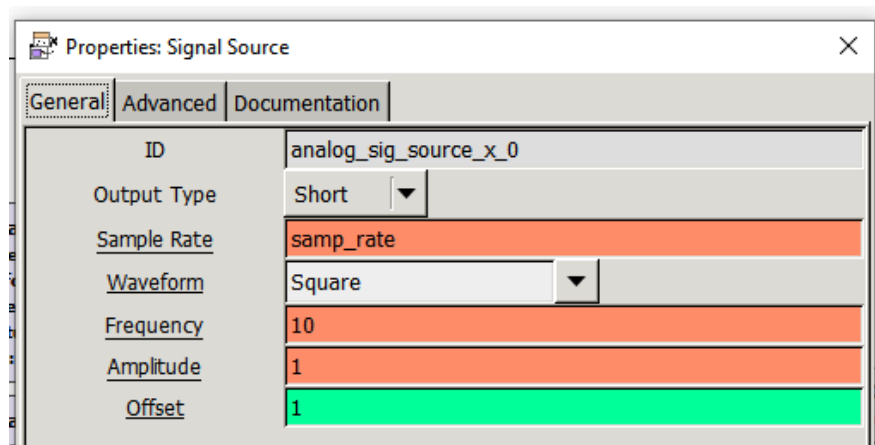
- *Signal source*
- *Trottle*
- *Not*

- *Short to float*
- *Multiply*
- *Add*
- *Scope sink*
- *Variable*

4.7.1.1. **Signal source modulador FSK**

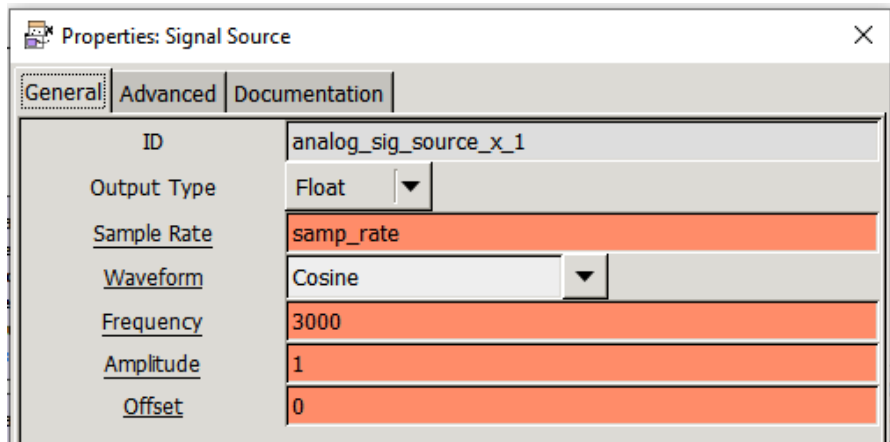
Los bloques *Signal source*, serán utilizados para la creación de tres ondas, de las cuales, el bloque 1 genera una onda cuadrada empleada como señal moduladora; los bloques 2 y 3 serán utilizados para generar 2 ondas con frecuencias distintas, para ser multiplicadas por la señal moduladora y la señal negada para la obtención de la señal modulada.

Figura 164. **Parámetros *signal source* 1 modulador FSK**



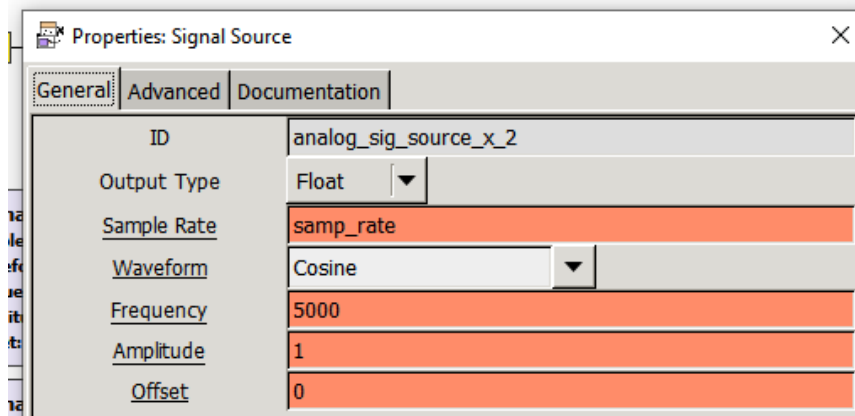
Fuente: elaboración propia, empleando GNU Radio.

Figura 165. **Parámetros *signal source 2* modulador FSK**



Fuente: elaboración propia, empleando GNU Radio.

Figura 166. **Parámetros *signal source 3* modulador FSK**

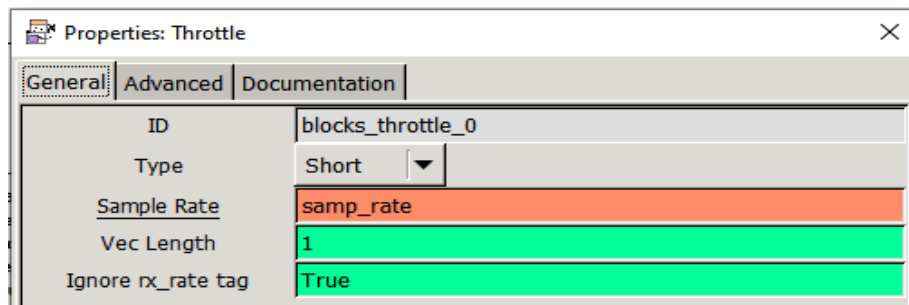


Fuente: elaboración propia, empleando GNU Radio.

4.7.1.2. *Throttle* modulador FSK

El bloque *Throttle* se encarga de regular el flujo de datos, manteniéndolo constante a su salida; los parámetros deben ser los siguientes:

Figura 167. **Parámetros *throttle* modulador FSK**

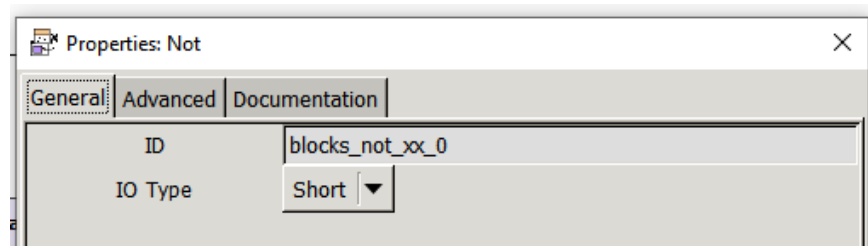


Fuente: elaboración propia, empleando GNU Radio.

4.7.1.3. *Not* modulador FSK

Bloque utilizado para negar la señal moduladora que va a ser utilizada como factor en la multiplicación de señales, para la creación de la onda modulada.

Figura 168. **Parámetros *not* modulador FSK**

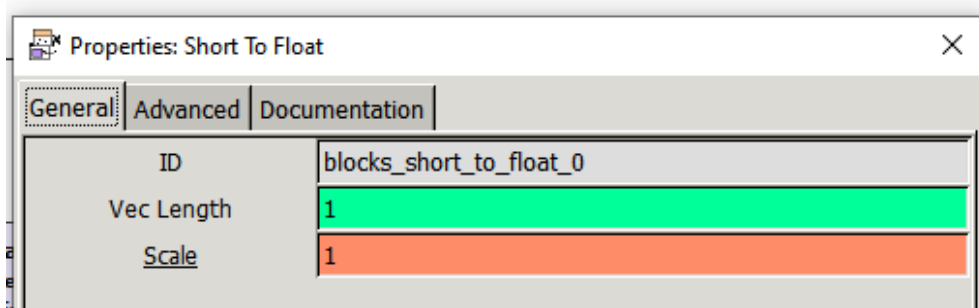


Fuente: elaboración propia, empleando GNU Radio.

4.7.1.4. *Short to float* modulador FSK

Bloque utilizado para hacer la conversión entre tipos de variables.

Figura 169. **Parámetros *short to float* modulador FSK**

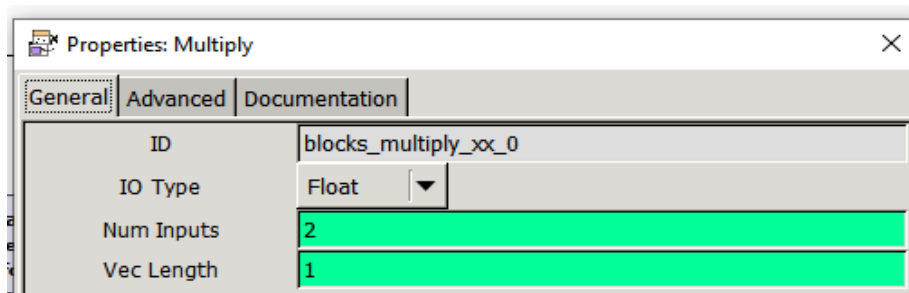


Fuente: elaboración propia, empleando GNU Radio.

4.7.1.5. *Multiply* modulador FSK

Los multiplicadores serán utilizados para efectuar el producto entre las señales; los parámetros deben ser los siguientes para ambos bloques:

Figura 170. **Parámetros *multiply* modulador FSK**

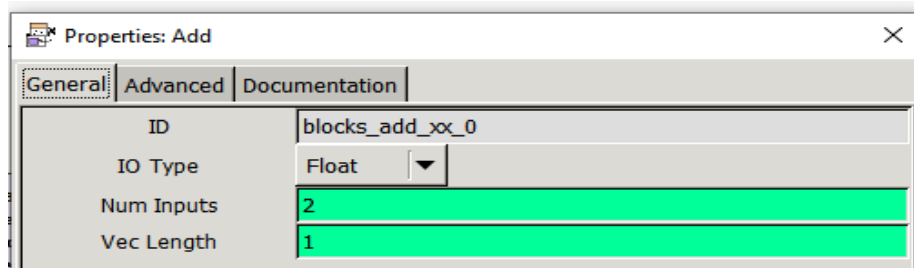


Fuente: elaboración propia, empleando GNU Radio.

4.7.1.6. Add modulador FSK

El bloque *Add* es utilizado para sumar las señales que conforman las 2 frecuencias empleadas y conseguir la onda modulada en desplazamiento de frecuencia.

Figura 171. Parámetros *Add* modulador FSK

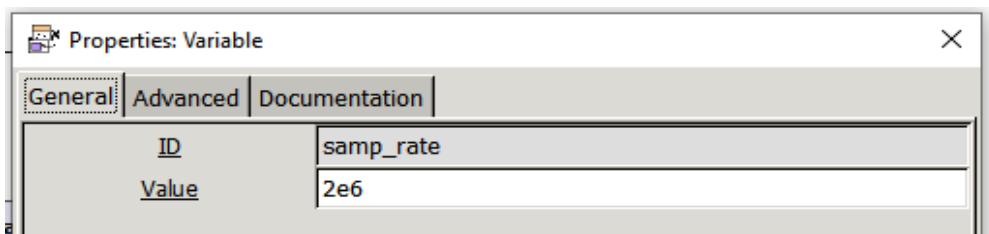


Fuente: elaboración propia, empleando GNU Radio.

4.7.1.7. Variable modulador FSK

Se utiliza una variable para especificar la frecuencia de muestreo; los parámetros serán los siguientes:

Figura 172. Parámetros variable modulador FSK

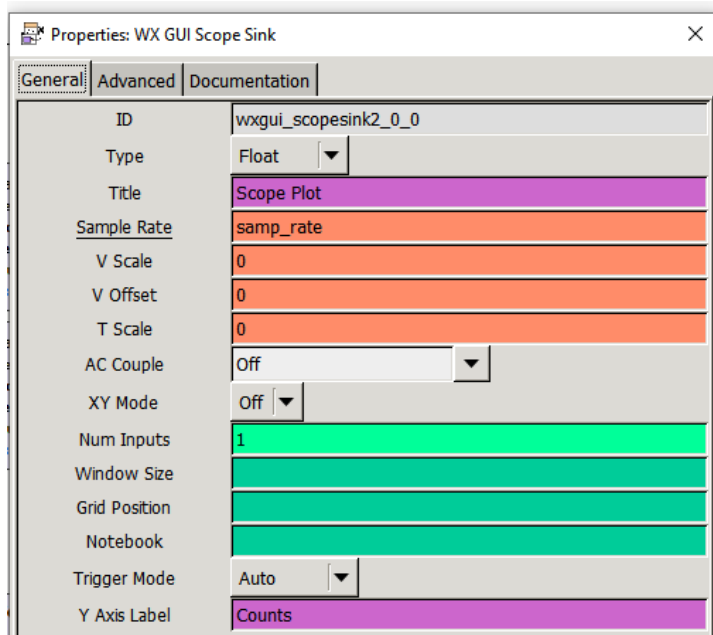


Fuente: elaboración propia, empleando GNU Radio.

4.7.1.8. WX GUI scope sink modulador FSK

Bloque utilizado para la visualización de la forma de onda de la señal modulada.

Figura 173. Parámetros WX GUI scope sink modulador FSK

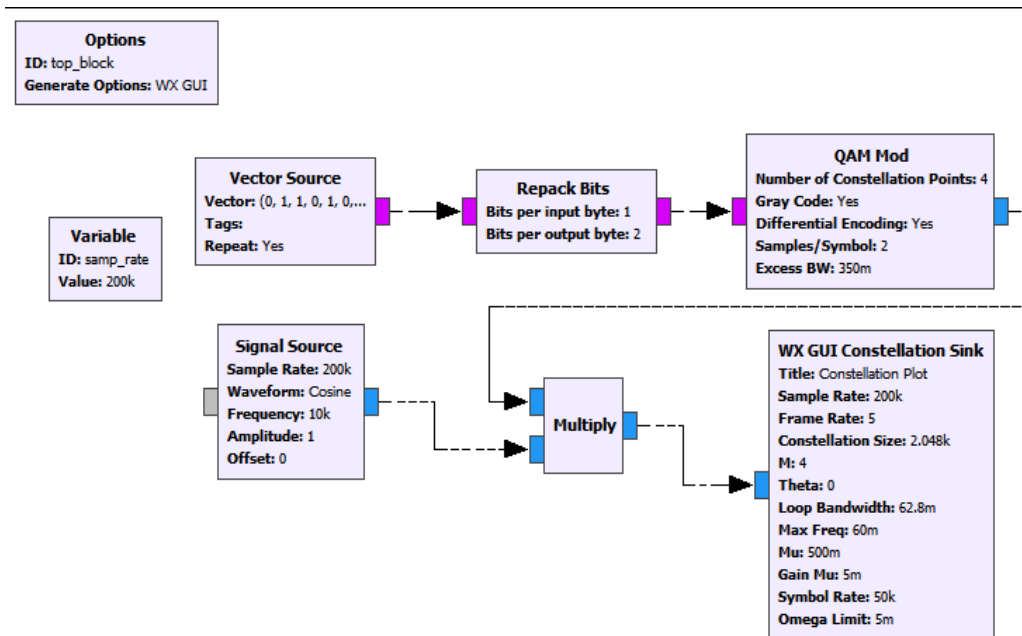


Fuente: elaboración propia, empleando GNU Radio.

4.8. Práctica 8: modulación QAM

La octava práctica aplicable al laboratorio de Comunicaciones 1 consiste en trabajar la modulación de amplitud en cuadratura, utilizando como señal moduladora un vector de bits.

Figura 174. Diagrama de bloques modulador QAM



Fuente: elaboración propia, empleando GNU Radio.

4.8.1. Modulador QAM

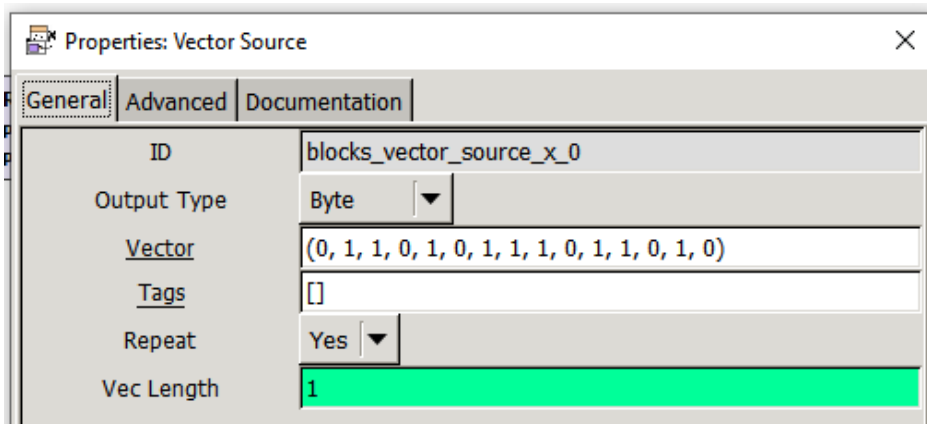
Los bloques utilizados para trabajar la modulación de amplitud en cuadratura son los siguientes:

- *Vector source*
- *Repack Bits*
- *QAM mod*
- *Signal source*
- *Multiply*
- *WX GUI constellation sink*
- *Variable*

4.8.1.1. *Vector source* modulador QAM

Bloque utilizado como fuente de información para obtener la señal moduladora; los parámetros del bloque deben ser los siguientes:

Figura 175. **Parámetros *vector source* modulador QAM**

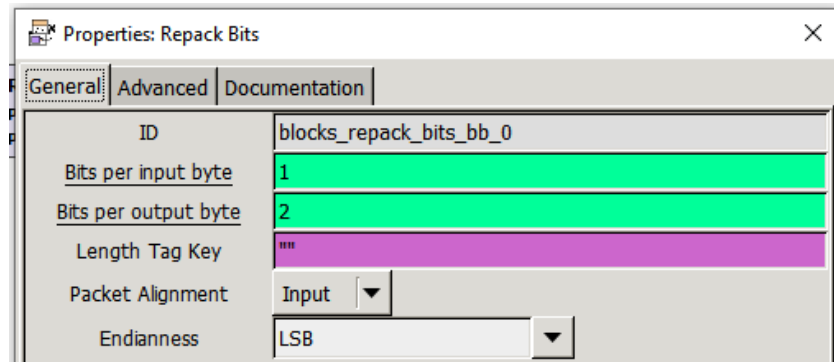


Fuente: elaboración propia, empleando GNU Radio.

4.8.1.2. *Repack bits* modulador QAM

Bloque utilizado para empaquetar los bits en parejas; este es necesario para la modulación de amplitud en cuadratura; los parámetros del bloque deben ser los siguientes:

Figura 176. **Parámetros *Repack bits* modulador QAM**

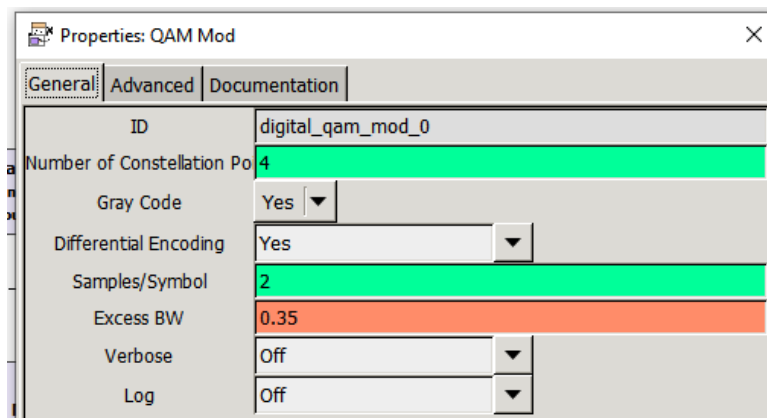


Fuente: elaboración propia, empleando GNU Radio.

4.8.1.3. **QAM *mod* modulador QAM**

Bloque utilizado para realizar la modulación de amplitud en cuadratura, para trabajar la modulación 4 QAM; los parámetros del bloque deben ser los siguientes:

Figura 177. **Parámetros QAM *mod* modulador QAM**

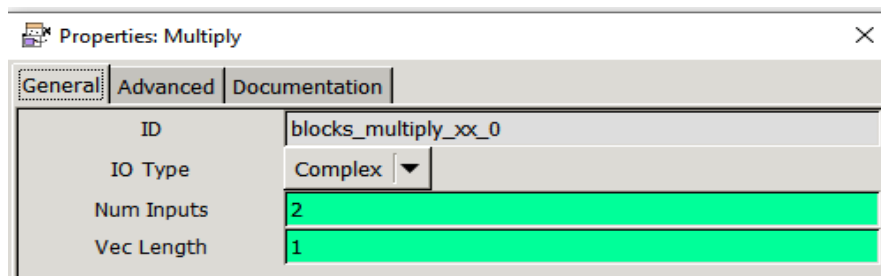


Fuente: elaboración propia, empleando GNU Radio.

4.8.1.4. *Multiply* modulador QAM

Bloque utilizado para realizar una multiplicación entre señales; los parámetros deben ser los siguientes:

Figura 178. **Parámetros *multiply* modulador QAM**

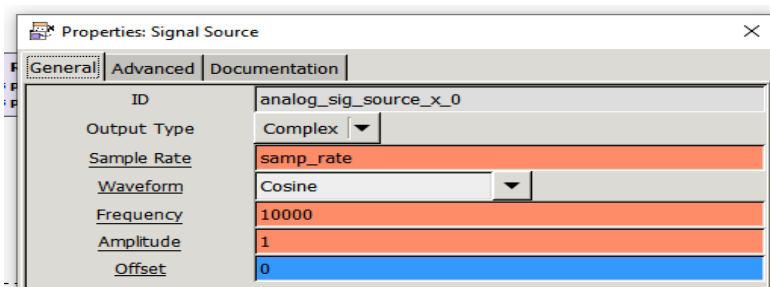


Fuente: elaboración propia, empleando GNU Radio.

4.8.1.5. *Signal source* modulador QAM

Bloque utilizado para la generación de una onda senoidal empleada como onda portadora; los parámetros deben ser los siguientes:

Figura 179. **Parámetros *signal source* modulador QAM**

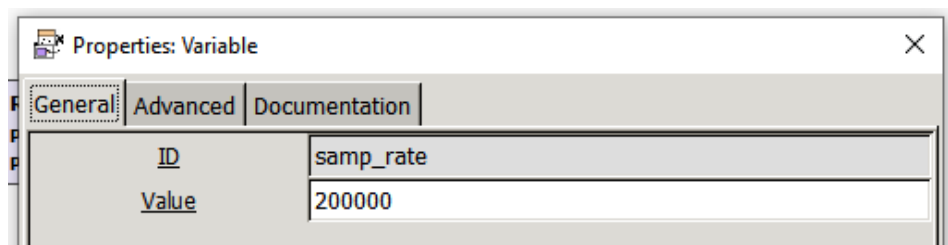


Fuente: elaboración propia, empleando GNU Radio.

4.8.1.6. Variable modulador QAM

Bloque utilizado para establecer la frecuencia de muestreo del sistema; los parámetros deben ser los siguientes:

Figura 180. Parámetros variable modulador QAM

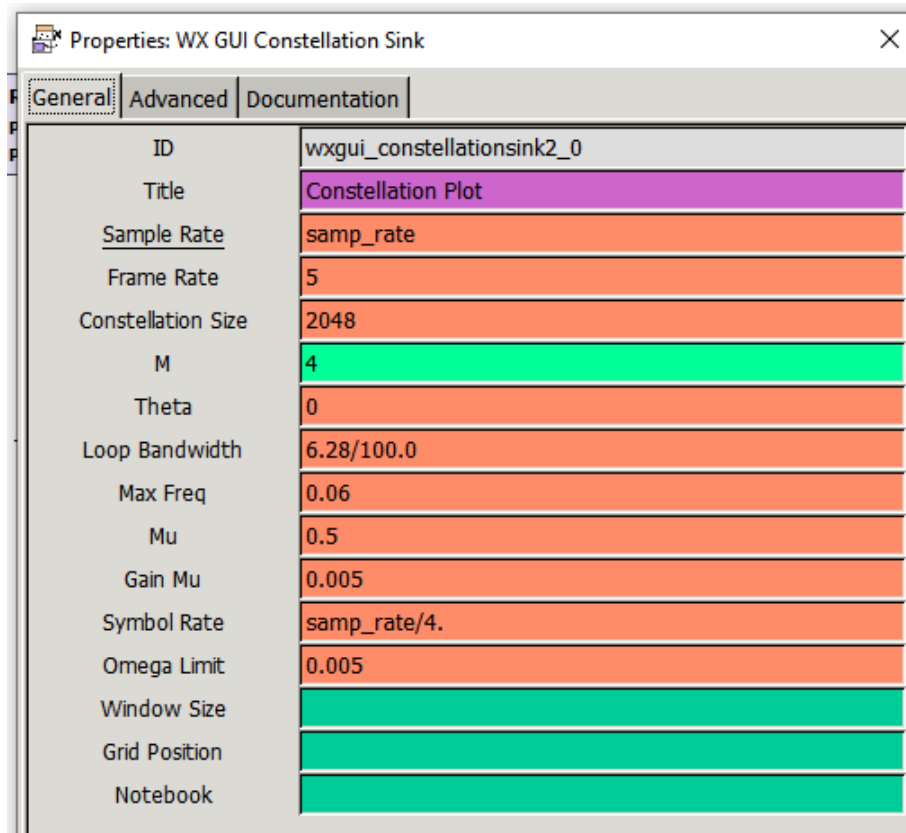


Fuente: elaboración propia, empleando GNU Radio.

4.8.1.7. WX GUI *constellation sink* modulador QAM

Bloque utilizado para graficar el diagrama de constelaciones de la señal modulada; los parámetros deben ser los siguientes:

Figura 181. **Parámetros WX GUI *constellation sink* modulador QAM**



Parameter	Value
ID	wxgui_constellationsink2_0
Title	Constellation Plot
Sample Rate	samp_rate
Frame Rate	5
Constellation Size	2048
M	4
Theta	0
Loop Bandwidth	6.28/100.0
Max Freq	0.06
Mu	0.5
Gain Mu	0.005
Symbol Rate	samp_rate/4.
Omega Limit	0.005
Window Size	
Grid Position	
Notebook	

Fuente: elaboración propia, empleando GNU Radio.

CONCLUSIONES

1. La tecnología de radio definida por software permite la implementación de radio enlaces a nivel de laboratorio, utilizando software libre y hardware de fácil acceso.
2. Los sistemas de telecomunicaciones pueden implementarse utilizando distintos esquemas de modulación, los cuales, al trabajar en amplitud, fase y frecuencia, permiten trabajar con datos analógicos o digitales.
3. La radio definida por software busca reemplazar una buena parte del hardware utilizado en los sistemas de telecomunicaciones, por el software dedicado al procesamiento digital de señales.
4. En los sistemas de telecomunicaciones la aplicación final dependerá de la complejidad del sistema físico, así como los diagramas trabajados en GNU Radio para el procesamiento de las señales, tanto para la etapa de transmisión como de recepción.
5. La tecnología de radio definida por software, utilizada como herramienta de laboratorio, permite poner en práctica y estudiar los distintos esquemas de modulación estudiados en el laboratorio de Comunicaciones 1 de la carrera de Ingeniería Electrónica, en la Universidad de San Carlos de Guatemala.

RECOMENDACIONES

1. Al trabajar con sistemas de telecomunicaciones, es de suma importancia estudiar la teoría detrás de los distintos esquemas de modulación.
2. Estudiar el análisis matemático, así como la representación gráfica de los esquemas de modulación ayuda a una mejor comprensión del comportamiento de los sistemas.
3. Es importante conocer el funcionamiento de los distintos bloques utilizados en GNU Radio, ya que esto permite la modificación de los parámetros de cada uno de los bloques para la optimización en la recepción o transmisión.
4. Para la implementación de un radio enlace a nivel de laboratorio, es preferible utilizar una Raspberry pi versión 3B+ en adelante, por los requerimientos de los sistemas.
5. Las prácticas presentadas fueron realizadas en el sistema operativo Ubuntu, en su versión 16.04; tanto la instalación de los *drivers* como del software utilizado están planteados para la versión de sistema operativo descrita anteriormente y si se desea utilizar otra, es necesario validar la compatibilidad.

BIBLIOGRAFÍA

1. AMADOR FUNDORA, José Ángel; TORRES, Néstor Alonso. *RDS (radio definido por software). Consideraciones para su implementación de hardware*. La Habana, Cuba: 2013. 13 p.
2. ANGUERA, Jaume; PÉREZ, Antonio. *Teoría de antenas*. España: Universidad Ramon Llull, 2008. 336 p.
3. BALANIS, Constantine A. *Antenna Theory third edition analysis and design*. Canadá: 2005. 1073 p.
4. CARRALERO ALONSO, David. *Radio definida por software en dispositivos de bajo coste*. España: Universidad de La Laguna, 2016. 102 p.
5. FERREL G., Stremler. *Introducción a los sistemas de comunicación*. Universidad de Wisconsin, Estados Unidos: 1990. 773 p.
6. GARCÍA ALGORA, Carlos Manuel. *Radio definido por Software usando MATLAB*. Cuba: Universidad Central Marta Abreu de Las Villas, 2011. 75 p.
7. MACHADO-FERNÁNDEZ, José Raúl. *Software Defined Radio: Principios y aplicaciones básicas*. La Habana, Cuba: 2015. 18 p.

8. REY MICOLAU, Francesc; TARRÉS RUIZ, Francesc. *Comunicaciones analógicas: modulaciones AM y FM. Módulo 2.* [en línea]. <http://openaccess.uoc.edu/webapps/o2/bitstream/10609/69406/5/Sistemas%20de%20comunicaci%C3%B3n%20I_M%C3%B3dulo%202_Comunicaciones%20anal%C3%B3gicas%3B%20modulaciones%20AM%20y%20FM.pdf>. [Consulta: febrero de 2020].
9. SÁNCHEZ LAKEHAL, Alejandro. *La radio definida por software: diseño de un receptor de banda aeronáutica VHF.* España: Escuela Politécnica Superior de Ingeniería de Vilanova i la Geltrú, 10 p.
10. TAUB, Herbert; SCHILLING, Donald L. *Principles of communication systems.* 2a ed. Estados Unidos: 1986. 487 p.
11. TOMASI, Wayne. *Sistemas de comunicaciones electrónicas.* 4a ed. DeVry Institute of Technology Phoenix, Arizona: 2001. 972 p.

ANEXOS

Anexo 1. **Características Raspberry pi 3B**

- *Processor*
 - Broadcom BCM2387 chipset.
 - 1,2 GHz Quad-Core ARM Cortex-A53 (64 Bit)

- 802.11 b/g/n Wireless LAN and Bluetooth 4.1 (Bluetooth Classic and LE)
 - IEEE 802.11 b / g / n Wi-Fi. Protocol: WEP, WPA WPA2, algorithms AES-CCMP (maximum key length of 256 bits), the maximum range of 100 meters.
 - IEEE 802.15 Bluetooth, symmetric encryption algorithm Advanced Encryption Standard (AES) with 128-bit key, the maximum range of 50 meters.

- GPU
 - Dual Core Video Core IV® Multimedia Co-Processor. Provides Open GL ES 2.0, hardware-accelerated
 - Open VG, and 1080p30 H.264 high-profile decode.
 - Capable of 1Gpixel/s, 1,5 Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure

Continuación anexo 1.

- Memory
 - 1 GB LPDDR2

- *Operating System*
 - Boots from Micro SD card, running a version of the Linux operating system or Windows 10 IoT

- Dimensions
 - 85 x 56 x 17 mm

- Power: micro USB socket 5V1, 2.5A

- :
- **Connectors:**

- Ethernet: 10/100 BaseT Ethernet socket

- Video Output
 - HDMI (rev 1.3 & 1.4)
 - Composite RCA (PAL and NTSC)

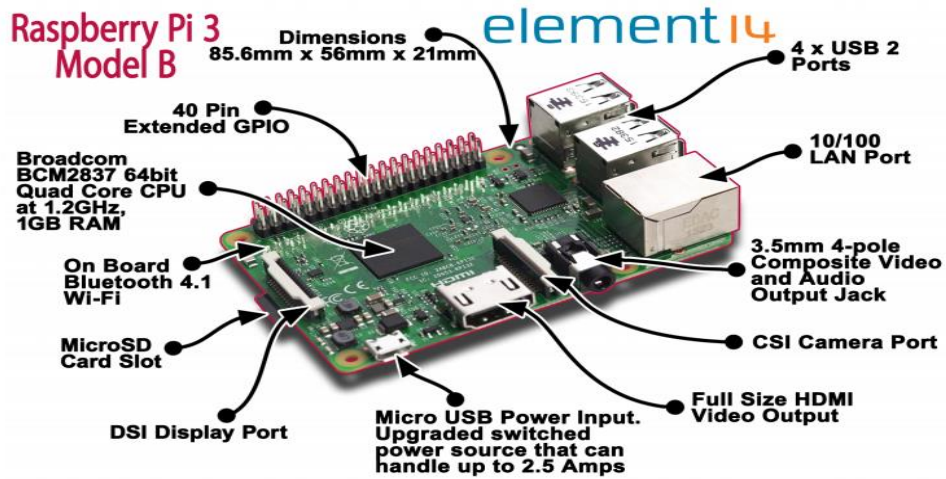
- Audio Output
 - Audio Output 3.5 mm Jack
 - HDMI
 - USB 4 x USB 2.0 Connector

Continuación anexo 1.

- GPIO Connector
 - 40-pin 2.54 mm (100 mil) expansion header: 2x20 strip
 - Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines
- Camera Connector: 15-pin MIPI Camera Serial Interface (CSI-2)
- Display Connector
 - Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane
- Memory Card Slot
 - Push/pull Micro SDIO Módulo RTL2832u

Fuente: TERRAELETRÓNICA. Características Raspberry pi 3B.
https://www.terraelectronica.ru/pdf/show?pdf_file=%252 Fds %252 Fpdf%252FT%252FTechicRP3.pdf. Consulta: mayo de 2020.

Anexo 2. Raspberry pi version 3B



Fuente: TERRAELETRÓNICA. Raspberry pi version 3B
https://www.terraelectronica.ru/pdf/show?pdf_file=%252Fds%252Fpdf%252FT%252FTechicRP3.pdf. Consulta: mayo de 2020.

Anexo 3. **Módulo USB Digital DVB-T SDR+DAB+FM HDTV
TVRTL2832U+ R820T**

CHOOSE A GENUINE RTL-SDR BLOG V3

ENTIRE PCB REDESIGNED FOR LOWER NOISE

REDESIGNED THERMAL LAYOUT (HELPS FIX VCO LOCK PROBLEMS)

BETTER LDO (LESS NOISE AND LOWER VOLTAGE OPERATION)

5V LINE FERRITE CHOKE

IMPROVED FRONT END DESIGN (RESULTING IN HIGHER L-BAND SNR)

4.5V BIAS TEE (SOFTWARE CONTROLLED)

1PPM TCXO

R820T2

SMA FEMALE CONNECTOR

ADDITIONAL ESD PROTECTION

DIRECT SAMPLING CIRCUIT ENABLES HF RECEPTION (WITH LPF)

EXPANSION PORTS

CLK SELECTOR JUMPER

GPIO EXPANSION PORTS

USB RF CHOKE (REMOVES USB NOISE)

TOUGH CONDUCTIVE METAL ENCLOSURE (REDUCES INTERFERENCE)

THERMAL PAD COOLING (REMOVES HEAT FROM PCB AND TRANSFERS IT TO THE METAL CASE RESULTING IN NO HEAT RELATED VCO LOCK PROBLEMS)

STANDARD/OTHER BRAND RTL-SDR (NOISE FLOOR FULL OF SPURS)

RTL-SDR V3 NOISE FLOOR (SIGNIFICANTLY REDUCED SPURS/BIRDIES)

FULL 2-YEAR WARRANTY AGAINST MANUFACTURING FAULTS
EMAIL & FORUM SUPPORT
SUPPORTS THE BLOG FOR NEW CONTENT, TUTORIALS AND PRODUCTS!

GENUINE GUARANTEE:
BE WARY OF INFERIOR
RTL-SDR BLOG V3 COUNTERFEITS!

RTL-SDR BLOG

Fuente: *RTL-SDR.Módulo USB Digital DVB-T SDR+DAB+FM HDTV*

TVRTL2832U+ R820T <https://www.rtl-sdr.com/buy-rtl-sdr-dvb-t-dongles/>. Consulta: mayo de 2020.

