

ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

IMPLEMENTACIÓN DE PRÁCTICAS DE LABORATORIO PARA LA MATERIA DE COMUNICACIÓN DIGITAL UTILIZANDO RADIO DEFINIDA POR SOFTWARE Y GNU Radio

**TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE
INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES**

MARIO DANIEL CARCHI SAÑAY

DIRECTOR: MSc. PABLO WILIAN HIDALGO LASCANO

Quito, diciembre 2020

AVAL

Certifico que el presente trabajo fue desarrollado por Mario Daniel Carchi Sañay, bajo mi supervisión.

MSc. Pablo Wilian Hidalgo Lascano
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo, Mario Daniel Carchi Sañay, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

Mario Daniel Carchi Sañay

AGRADECIMIENTO

“Es bueno darse cuenta de que cada persona que te encuentras tiene un rol en tu vida, algunos te ponen a prueba, otros te aman, unos te utilizan y otros te enseñan. Pero los realmente importantes son los que sacan lo mejor de ti mismo. Son esas personas poco comunes y extraordinarias que dejan huellas en tu vida y te recuerdan que este mundo vale la pena”

Todavía recuerdo la ilusión y nerviosismo que tuve al ingresar en esta noble institución, aquella jerarquía que representa ser Politécnico era un motivo más para sentirse presionado, pero es justamente esa presión la que me motivo a rendir al máximo.

Es cierto que se hacen sacrificios extras, pero los resultados son los esperados; y para eso siempre estuve apoyado, desde mis padres Mario y Carmen, quienes velaron por mi toda su vida, gracias a ellos estoy aquí, culminando una etapa más de mi vida. Gracias a mis hermanos Luis, Diana y Susana, quienes me apoyaron en todo momento, brindándome consejos y ánimos para seguir adelante.

Los profesores, quienes más allá de brindarme conocimientos me inculcaron valores, ayudando a formarme como profesional y como persona. Mi sincero agradecimiento al MSc. Pablo Wilian Hidalgo Lascano, quien más allá de ser mi director de tesis fue un guía en esta etapa de mi vida, gracias por sus consejos y su apoyo para el desarrollo de este Trabajo de Titulación.

Especial reconocimiento merece el Ing. Homero Ortega Boada, profesor de la Escuela de Ingeniería Eléctrica, Electrónica y de Telecomunicaciones de la Universidad Industrial de Santander - UIS, quien me brindó toda su conocimiento para el desarrollo de este Trabajo de Titulación, ya que sin el no habría sido posible el desarrollo de este trabajo.

A todos los mencionados, mis más sinceros agradecimientos.

ÍNDICE DE CONTENIDO

AVAL	I
DECLARACIÓN DE AUTORÍA.....	II
AGRADECIMIENTO.....	III
ÍNDICE DE CONTENIDO.....	IV
GLOSARIO DE TÉRMINOS.....	VI
RESUMEN	VII
ABSTRACT	VIII
1. INTRODUCCIÓN.....	1
1.1 OBJETIVOS	1
1.2 ALCANCE	2
1.3 MARCO TEÓRICO.....	4
1.3.1 RADIO DEFINIDA POR SOFTWARE	4
1.3.2 DISPOSITIVOS QUE OPERAN CON SDR.....	5
1.3.3 FUNCIONALIDADES Y CARACTERÍSTICAS DEL USRP NI 2920	6
1.3.4 SOFTWARE DE SDR.....	10
1.3.5 GNU Radio	10
1.3.6 CONEXIÓN USRP A GNU Radio	14
1.3.7 FUNDAMENTOS DE COMUNICACIÓN DIGITAL.....	16
2. METODOLOGÍA.....	20
2.1 HOJAS GUÍA DE LAS PRÁCTICAS DE LABORATORIO.....	20
2.1.1 PRÁCTICA 1: INTRODUCCIÓN A GNU Radio Y USRP NI 2920.....	20
2.1.2 PRÁCTICA 2: ANÁLISIS DE SEÑALES.....	24
2.1.3 PRÁCTICA 3: ANÁLISIS DEL CANAL DE COMUNICACIONES.....	28
2.1.4 PRÁCTICA 4: DIGITALIZACIÓN DE SEÑALES	30
2.1.5 PRÁCTICA 5: CUANTIFICACIÓN NO UNIFORME	33
2.1.6 PRÁCTICA 6: MODULACIÓN M-PSK.....	37
2.1.7 PRÁCTICA 7: DEMODULACIÓN M-PSK	45
2.1.8 PRÁCTICA 8: ANÁLISIS DE CANALES CON ISI (<i>INTERSYMBOL INTERFERENCE</i>)	51
2.1.9 PRÁCTICA 9: DSSS (<i>DIRECT SEQUENCE SPREAD SPECTRUM</i>)	54
3. RESULTADOS Y DISCUSIÓN	57

3.1	RESULTADOS DE LAS PRÁCTICAS	57
3.1.1	PRÁCTICA 1: INTRODUCCIÓN A GNU RADIO Y USRP NI 2920	57
3.1.2	PRÁCTICA 2: ANÁLISIS DE SEÑALES.....	59
3.1.3	PRÁCTICA 3: ANÁLISIS DEL CANAL DE COMUNICACIONES.....	64
3.1.4	PRÁCTICA 4: DIGITALIZACIÓN DE SEÑALES	68
3.1.5	PRÁCTICA 5: CUANTIFICACIÓN NO UNIFORME	72
3.1.6	PRÁCTICA 6: MODULACIÓN M-PSK.....	75
3.1.7	PRÁCTICA 7: DEMODULACIÓN M-PSK	78
3.1.8	PRÁCTICA 8: ANÁLISIS DE CANALES CON ISI (<i>INTERSYMBOL INTERFERENCE</i>)	82
3.1.9	PRÁCTICA 9: DSSS (<i>DIRECT SEQUENCE SPREAD SPECTRUM</i>)	86
4.	CONCLUSIONES Y RECOMENDACIONES	92
4.1	CONCLUSIONES.....	92
4.1	RECOMENDACIONES	93
5.	REFERENCIAS BIBLIOGRÁFICAS.....	95
	ANEXOS	95

GLOSARIO DE TÉRMINOS

SDR: Software Defined Radio

GRC: GNU Radio Companion

USRP: Universal Software Radio Peripheral

NI: National Instruments

DSSS: Direct Sequence Spread Spectrum

FHSS: Frequency Hopping Spread Spectrum

FIR Filter: Finite Impulse Response Filter

WBFM: Wide Band Frequency Modulation

ISI: Intersymbol Interference

PSK: Phase Shift Key

ASK: Amplitude Shift Key

FSK: Frequency Shift Key

RESUMEN

El presente trabajo de titulación tiene como objetivo principal el desarrollo de prácticas de laboratorio para la materia de Comunicación Digital. Para ello, se usan varias herramientas que se encuentran a disposición de estudiantes y profesores, tales como GNU Radio y los equipos USRP (*Universal Software Radio Peripheral*) NI 2920, disponibles en la Facultad de Ingeniería Eléctrica y Electrónica de la Escuela Politécnica Nacional.

Este Proyecto Técnico consta de 4 capítulos en los cuales se detalla el modo de uso tanto del software como del hardware empleado en el desarrollo de las prácticas, así como el diseño de éstas, donde cada una tendrá su propio manual que será puesto a disposición del instructor de laboratorio.

En el primer capítulo, se describen las características más relevantes de la tecnología SDR (*Software Defined Radio*), así como de los diferentes dispositivos que operan bajo esta tecnología. Además, se expone un breve resumen del modo de uso y configuración básica del software GNU Radio.

En el segundo capítulo se presentan las hojas guía para estudiantes del laboratorio de Comunicación Digital, es decir, los requerimientos necesarios para cada práctica y los pasos a seguir para su desarrollo.

En el tercer capítulo se exponen los resultados obtenidos de cada práctica, al variar distintos parámetros de los flujogramas.

En el cuarto capítulo se presentan las conclusiones a las cuales se llegó después del desarrollo del presente trabajo de titulación. Además, se mencionan una serie de recomendaciones que serán útiles para futuros trabajos de titulación.

Además, en forma de anexos, se presentan la resolución de las prácticas de laboratorio, tanto en la parte teórica como en el procedimiento práctico.

PALABRAS CLAVE: GNU Radio, USRP, Comunicación Digital, Radio Definida por Software.

ABSTRACT

The main objective of this final career project is the development of laboratory practices for the subject of Digital Communication. To do this, some tools that are available to students and teachers will be used, such as GNU Radio and the USRP (*Universal Software Radio Peripheral*) NI 2920 equipment, available at the Faculty of Electrical and Electronic Engineering of the Escuela Politécnica Nacional.

This Technical Project consists of 4 chapters in which the way of use of both the software and the hardware used in the development of the practices is detailed, as well as the design of these, which will have their own manual that will be made available to the lab instructor.

In the first chapter, the most relevant characteristics of SDR (Software Defined Radio) technology are described, as well as the different devices that operate under this technology. In addition, there is a summary of how to use and configure the GNU Radio software.

The second chapter presents the guide sheets for students for the Digital Communication laboratory, that means, the necessary requirements for each practice and the steps to follow for its development.

The third chapter presents the results obtained from each practice, by varying different parameters of the flowcharts.

The fourth chapter presents the conclusions reached after the development of this degree work. In addition, a series of recommendations are mentioned that will be useful for future degree work.

In addition, in the form of annexes, the solution of the laboratory practices is presented, both in the theoretical part and in the practical procedure.

KEYWORDS: GNU Radio, USRP, Digital Communication, Software Defined Radio.

1. INTRODUCCIÓN

En los últimos años, los sistemas de comunicación por radio han ido evolucionando de manera acelerada, a tal punto que, las comunicaciones se encuentran en un ambiente de “todo definido por software”. SDR (*Software Defined Radio*) considera aquellos radios que incluyen un transmisor cuyos parámetros de operación, tales como, banda de frecuencias, tipo de modulación o potencia máxima de salida pueden ser alterados mediante cambios en software, sin realizar cambios a los componentes de hardware que se relacionan con la emisión de radiofrecuencias [1].

GNU Radio es un conjunto de herramientas de desarrollo de software gratuito y de código abierto, el cual suministra bloques de procesamiento de señales para implementar aplicaciones mediante software [2].

Durante los últimos años, las prácticas de laboratorio de la materia de Comunicación Digital en la Escuela Politécnica Nacional no han tenido un cambio significativo en cuanto a las herramientas utilizadas tanto en hardware como en software, provocando de esta manera que se omitan varios temas correspondientes al pensum de la asignatura, tales como: análisis de diagramas de ojo, digitalización de señales y cuantización no uniforme, entre otros.

Por esta razón, se propone el uso de las herramientas antes mencionadas, ya que éstas, proporcionarían mayor versatilidad en el desarrollo de las prácticas y complementarían el conocimiento de los estudiantes en cuanto a la materia de Comunicación Digital.

1.1 OBJETIVOS

El objetivo general de este Proyecto Técnico es implementar las prácticas de laboratorio para la materia de Comunicación Digital utilizando Radio Definida por Software y GNU Radio.

Los objetivos específicos del Proyecto Técnico son:

- Analizar las características y funcionalidades de los equipos SDR (*Software Defined Radio*) y GNU Radio.
- Analizar los fundamentos teóricos de la materia de Comunicación Digital.
- Diseñar las prácticas de Comunicación Digital en base al PEA de la materia.
- Analizar los resultados obtenidos durante el desarrollo de cada una de las prácticas.

1.2 ALCANCE

El presente Proyecto de Titulación se enfoca en el diseño e implementación de prácticas de laboratorio de Comunicación Digital, mediante el uso de dispositivos que operan bajo la tecnología SDR y GNU Radio.

El estudiante mediante un computador podrá visualizar el diagrama de flujo correspondiente a cada práctica. Podrá controlar, en tiempo real, los diferentes parámetros establecidos en cada práctica. De igual manera, podrá obtener gráficas de los resultados de las prácticas, y su variación de acuerdo con la configuración de parámetros en tiempo real.

La mayoría de las prácticas a implementarse están compuestas de dos partes, una para la parte de transmisión (Figura 1.1) y otra para la parte de recepción (Figura 1.2), las cuales, en un ambiente de simulación, necesitarían de una sola computadora. Sin embargo, se utilizarán dos computadoras y dos dispositivos USRP (*Universal Software Radio Peripheral*) NI 2920, ya que se conectarán de manera inalámbrica en un entorno de laboratorio (Figura 1.3).

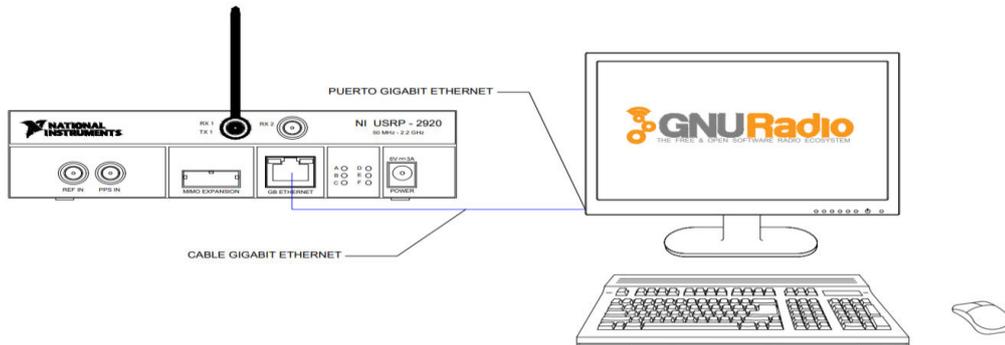


Figura 1.1. Diagrama TX USRP NI 2920

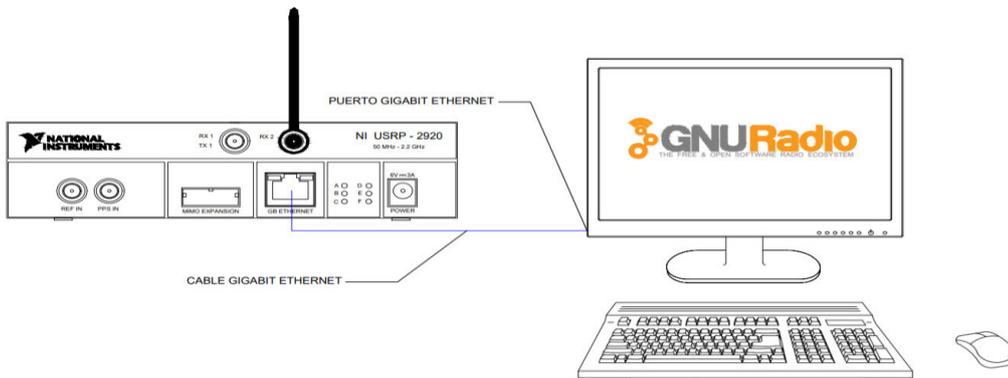


Figura 1.2. Diagrama RX USRP NI 2920

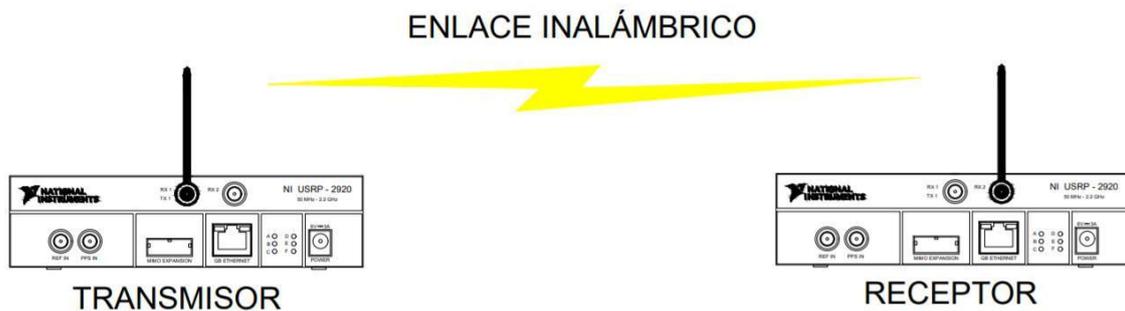


Figura 1.3. Enlace inalámbrico entre USRP NI 2920

La implementación de las partes de transmisión y de recepción de cada práctica se desarrollará en GNU Radio. Sin embargo, existen dos maneras de implementarlas; la primera usando bloques de moduladores, filtros, transmisores y receptores propios de GNU Radio; la segunda opción es a partir de bloques jerárquicos o programación en Python (opcional), crear cada bloque necesario para cada práctica. Para el presente Proyecto, se optó por la segunda opción, esto es, desarrollar las prácticas mediante bloques jerárquicos, ya que, con ello se permitirá a los estudiantes comprender de mejor manera el funcionamiento de la plataforma de GNU Radio y mostrar a mayor detalle el proceso de cada una de las etapas de un sistema digital de comunicación.

El presente proyecto implementa 9 prácticas, entre las cuales se tienen:

Práctica 1: Introducción a GNU Radio y USRP NI-2920

Objetivo: Brindar a los estudiantes los conocimientos básicos de funcionamiento de la plataforma GNU Radio y de los equipos NI-USRP-2920.

Práctica 2: Análisis de señales

Objetivo: Analizar los distintos parámetros de señales conocidas, parámetros como período, frecuencia y amplitud, así como operaciones entre señales.

Práctica 3: Análisis del canal de comunicaciones

Objetivo: Analizar las alteraciones que sufre una señal al atravesar el canal de comunicaciones.

Práctica 4: Digitalización de señales

Objetivo: Analizar la digitalización de señales y sus conceptos asociados como: muestreo, cuantización y codificación.

Práctica 5: Cuantificación no uniforme

Objetivo: Analizar la cuantificación no uniforme dentro del proceso de digitalización de señales.

Práctica 6: Modulación M-PSK

Objetivo: Comprender el proceso de Modulación Digital de múltiples estados. Caso de estudio: Modulación PSK (*Phase Shift Key*) [3].

Práctica 7: Demodulación M-PSK

Objetivo: Comprender el proceso de Demodulación Digital de múltiples estados. Caso de estudio: Demodulación PSK (*Phase Shift Key*) [3].

Práctica 8: Análisis de canales con ISI (*Intersymbol Interference*)

Objetivo: Analizar diferentes conceptos relacionados a la transmisión digital de señales, tales como factor de *roll-off*, *jitter*, ISI y diagrama del ojo.

Práctica 9: DSSS (*Direct Sequence Spread Spectrum*)

Objetivo: Comprender el sistema DSSS mediante la utilización de un sistema de modulación BPSK.

Además, se elaborará un manual de cada práctica para proporcionarle como guía al instructor. Igualmente, se realizarán todas las simulaciones y ejercicios de cada práctica, las cuales se incluirán en el manual del instructor para que pueda solventar cualquier duda de los estudiantes.

1.3 MARCO TEÓRICO

En este capítulo se explorarán conceptos que serán necesarios para el desarrollo y comprensión del Trabajo de Titulación. Se estudiarán conceptos como *SDR (Software Defined Radio)*, así como los diferentes dispositivos que operan bajo esta tecnología. Además, se dará una breve introducción del *software* utilizado en este trabajo, es decir, GNU Radio. Por otra parte, se mencionarán algunas definiciones de Comunicación Digital, los cuales fungirán como introducción para el capítulo II.

1.3.1 RADIO DEFINIDA POR SOFTWARE

El término “Radio Definida por Software” fue propuesto por Joe Mitola en 1991 para hacer referencia a radios reconfigurables [4]; es decir, un mismo dispositivo capaz de realizar distintas funciones en diferentes instantes de tiempo, todo esto sin modificar sus componentes físicos.

SDR (*Software Defined Radio*) también conocida como SBR (*Software Based Radio*) o SR (*Software Radio*) [5], posee algunas definiciones, las cuales en conclusión exponen una alternativa en la cual se permite al usuario controlar la mayoría de las funcionalidades de un dispositivo de comunicaciones mediante software.

Además, debido a la versatilidad que presentan estos equipos, resultan útiles para aplicaciones en diferentes campos de comunicaciones, tales como: Televisión Digital, Comunicaciones Analógicas, Comunicaciones Digitales y Comunicaciones Inalámbricas.

1.3.2 DISPOSITIVOS QUE OPERAN CON SDR

En la actualidad existen diferentes dispositivos que operan bajo este tipo de tecnología, los cuales son utilizados en distintos campos, tales como: profesional, académico, táctico, etc.

Entre los equipos SDR disponibles al año 2020 se tienen:

1.3.2.1 HackRF One

Es un periférico SDR capaz de transmitir o recibir señales de radio en un rango de frecuencias de 1 - 6 MHz, como indica la tabla 1.1; es una plataforma de hardware de código abierto que puede ser programado para operación independiente o mediante un cable USB. [6]

Tabla 1.1. Características HackRF One [6]

Frecuencia operativa	1 – 6 MHz
Software controlador	GNU Radio, SDR
Cable controlador	USB 2.0
Conector	SMA hembra

1.3.2.2 YARD Stick One

Dispositivo que funciona bajo la tecnología SDR; puede transmitir o recibir señales de radio en un rango de frecuencias menores a 1 GHz, siendo programado mediante un cable USB. Otras funciones se muestran en la tabla 1.2 [6].

Tabla 1.2. Características YARD Stick One [6]

Frecuencia operativa	300 - 348 MHz, 391 - 464 MHz y 782 - 928 MHz
Software controlador	Shell Python / Software propio

Cable controlador	USB 2.0
Conector	SMA hembra

1.3.2.3 RTL SDR R820T2

Dispositivo que funciona bajo tecnología SDR; puede transmitir o recibir señales de radio en un rango de frecuencias menores a 1,8 GHz, como indica la tabla 1.3, siendo programado mediante un cable USB. [6]

Tabla 1.3. Características R820T2 [6]

Frecuencia operativa	500 KHz - 1,766 GHz
Software controlador	Software propio R820T
Cable controlador	USB 2.0
Conector	SMA hembra

1.3.2.4 USRP NI 2920

Dispositivo que funciona bajo tecnología SDR; puede transmitir o recibir señales de radio dentro de un rango de frecuencias de 20 MHz – 2.2 GHz, como indica la figura 1.4. [6]

Tabla 1.4. Características USRP NI-2920 [6]

Frecuencia operativa	20 MHz – 2.2 GHz
Software controlador	GNU Radio / LabVIEW
Cable controlador	Gigabit Ethernet
Conector	SMA hembra

1.3.3 FUNCIONALIDADES Y CARACTERÍSTICAS DEL USRP NI 2920

El modelo USRP 2920 (Figura 1.4) es un dispositivo SDR que permite tanto transmisión como recepción de datos; éste posee diferentes interfaces de control (Tabla 1.5).

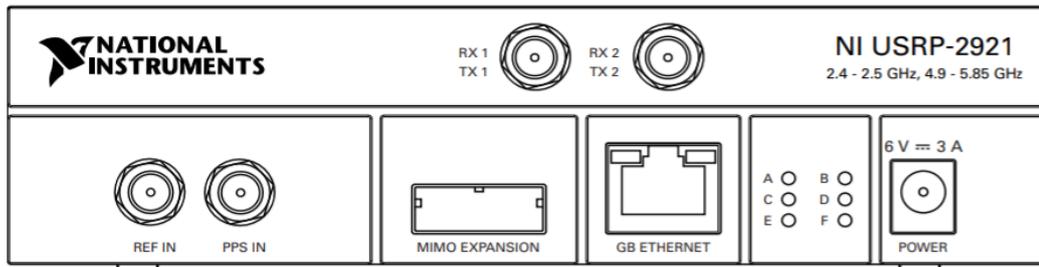


Figura 1.4. Panel frontal USRP NI – 2920 [7]

Tabla 1.5. Interfaces USRP NI – 2920 [7]

INTERFAZ	FUNCIONALIDAD
RX1 TX1	Utilizada para transmisión y recepción, la cual es configurable mediante el software a utilizar. Es una interfaz con una impedancia de 50 Ω .
RX2	Utilizada únicamente para la recepción de señales de radiofrecuencia con una impedancia de 50 Ω .
REF IN	Terminal de entrada para una señal de referencia externa para el Oscilador Local (LO) en el dispositivo. REF IN presenta una impedancia de 50 Ω y una entrada de referencia de un solo extremo. REF IN acepta una señal de 10 MHz con una potencia de entrada mínima de 0 dBm (.632 Vpk-pk) y una potencia de entrada máxima de 15 dBm (3.56 Vpk-pk) para una onda cuadrada o sinusoidal.
PPS IN	Terminal de entrada para la referencia de temporización de pulso por segundo (PPS). PPS IN presenta una impedancia de 50 Ω y una entrada de un solo extremo. PPS IN acepta 0 V a 3.3 V TTL y 0 V a 5 V Señales TTL.
EXPANSIÓN MIMO	Interfaz utilizada para conectar dos dispositivos USRP mediante un cable MIMO.
GB ETHERNET	Puerto Gigabit Ethernet utilizado para conectar el ordenador por el cual se va a configurar el dispositivo.
POWER	Entrada para el cable de poder de 6 V.

En la tabla 1.6 se muestran las características de radio del dispositivo USRP NI 2920.

Tabla 1.6. Características de radio USRP NI – 2920 [7]

	Transmisión	Recepción
Rango de frecuencia	50 MHz – 2.2 GHz	50 MHz – 2.2 GHz
Rango de ganancia	1.0 dB – 31dB	5.0 dB – 31.5 dB
Pasos de ganancia	1.0 dB	0.5 dB
Convertidor análogo digital	2 canales, 400 MS/s	2 canales, 100 MS/s
Ancho de banda máximo	16 bits a 20 MHz 8 bits a 40 MHz	16 bits a 20 MHz 8 bits a 40 MHz

Además, para verificar el correcto funcionamiento del dispositivo es primordial conocer el significado de cada uno de los leds del panel frontal del dispositivo (Tabla 1.7).

Tabla 1.7. Indicadores LED [7]

Led	Indicación
A	Indica el estado de la transmisión del dispositivo: <ul style="list-style-type: none"> • OFF: el módulo no está transmitiendo datos. • GREEN: el módulo está transmitiendo datos.
B	Indica el estado del enlace MIMO cableado: <ul style="list-style-type: none"> • OFF: el dispositivo no está conectado mediante cable MIMO. • GREEN: el dispositivo está conectado mediante cable MIMO.
C	Indica el estado de la recepción del dispositivo: <ul style="list-style-type: none"> • OFF: el módulo no está recibiendo datos. • GREEN: el módulo está recibiendo datos.
D	Indica el estado del <i>firmware</i> del dispositivo: <ul style="list-style-type: none"> • OFF: el <i>firmware</i> del dispositivo no está cargado. • GREEN: el <i>firmware</i> del dispositivo está cargado.
E	Indica el estado del bloqueo de referencia del oscilador local del dispositivo: <ul style="list-style-type: none"> • OFF: el dispositivo no tiene señal de referencia. • GREEN: el dispositivo tiene señal de referencia.
F	Indica el estado del equipo: <ul style="list-style-type: none"> • OFF: el equipo está apagado. • GREEN: el equipo está encendido.

El USRP NI 2920 posee varios bloques de procesamiento interno, los cuales van modificando a la señal transmitida o recibida en datos que puedan ser entendidos por el ordenador al cual está conectado el dispositivo.

En la figura 1.5, se muestra la estructura interna de la USRP NI 2920, en la cual las características que se deben tener en cuenta se detallan a continuación.

1.3.3.1 Control de transmisión

En la parte de transmisión del dispositivo USRP NI 2920 se tienen los siguientes bloques [8]:

- *DUC (Digital Up Converter)*: este bloque es encargado de mezclar, filtrar e interpolar la señal a transmitir a una frecuencia de muestreo de 400 MS/s.
- *DAC (Digital to Analog Converter)*: este bloque realiza la conversión digital - análoga de la señal a transmitir.
- *Filtro pasa bajos*: este bloque reduce el ruido y las componentes de alta frecuencia de la señal a transmitir [8].
- *Mixer*: realiza una conversión de las señales a una frecuencia especificada por el usuario [8].
- *PLL: Phase Loop Lock*, es el encargado de controlar el oscilador de voltaje VCO de tal manera que el reloj y el oscilador local del dispositivo se sincronicen en frecuencia a una señal de referencia [8].
- *Amplificador*: controla la ganancia de la señal a transmitir.

1.3.3.2 Control de Recepción

En la parte de recepción del dispositivo USRP NI 2920 se tienen los siguientes bloques:

- *Amplificador de bajo ruido y amplificador controlado*: encargados de amplificar la señal entrante [8].
- *PLL: Phase Loop Lock*, es el encargado de controlar el oscilador de voltaje VCO de tal manera que el reloj y el oscilador local del dispositivo se sincronicen en frecuencia a una señal de referencia [8].
- *Mixer*: convierte las señales a banda base en sus componentes de fase I y Q correspondientes [8].
- *El filtro pasabajos*: encargado de reducir los efectos del ruido y las componentes de alta frecuencia de la señal.
- *ADC*: convertidor análogo digital.
- *DDC: Digital Down Converter*, encargado de mezclar, filtrar y decimar la señal a transmitir a una frecuencia de muestreo de 100 MS/s [8].

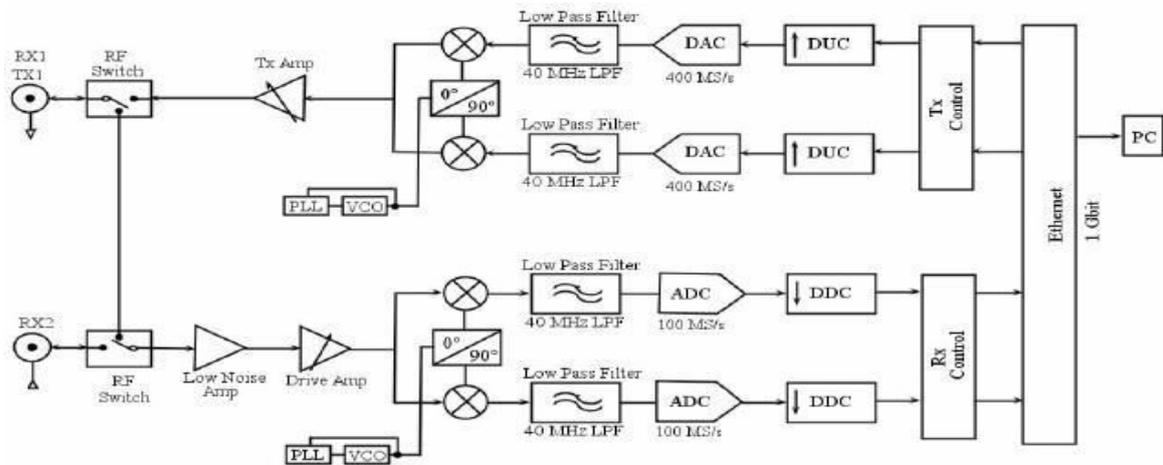


Figura 1.5 Diagrama de bloques USRP NI 2920 [8]

Además, otro aspecto para tomar en cuenta en el funcionamiento del dispositivo es el tipo de señal que admite. El USRP NI 2920, reconoce señales continuas en el tiempo, por lo que, se necesita adaptar la señal antes de ser enviada al USRP.

En conclusión, luego de que se realiza la adaptación de la señal, esta, atraviesa por distintos procesos antes de ser transmitida, es decir, la señal es interpolada, convertida a analógica, filtrada, convertida a la frecuencia a la cual se transmitirá y finalmente amplificada.

Por otra parte, las señales que recibe el dispositivo realizan el proceso inverso, pero con algunas modificaciones, es decir, la señal es decimada y se realiza una conversión analógica – digital.

1.3.4 SOFTWARE DE SDR

Como se mencionó anteriormente, dependiendo del dispositivo, existen diferentes tipos de software que pueden trabajar con SDR. En el caso de la USRP NI 2920, es posible trabajar con diferentes programas, tales como: GNU Radio y LabVIEW.

Para el presente Trabajo de Titulación, se optó por GNU Radio, debido a las ventajas que ofrece, las cuales serán descritas en este capítulo.

1.3.5 GNU Radio

GNU Radio es un conjunto de herramientas de desarrollo de software gratuito y de código abierto, el cual suministra bloques de procesamiento de señales para implementar aplicaciones mediante software [2]. Entre sus ventajas se encuentra el poder usarlo con dispositivos de RF externo para crear radios definidas por software, o sin hardware simplemente para simular escenarios prácticos.

GNU Radio al ser un software libre, es muy útil en el campo de investigación y en el campo académico, ya que, mediante éste, tanto investigadores como estudiantes pueden comprender de mejor manera temas relacionados a las Comunicaciones Inalámbricas, Comunicaciones Digitales, Televisión Digital, etc.

1.3.5.1 Características de GNU Radio

GNU Radio es una plataforma que se caracteriza por su compatibilidad con diferentes sistemas operativos, programación entre bloques y programación mediante Python o C++, entre otros.

- **Compatibilidad con varios sistemas operativos**

Debido a que la plataforma no demanda excesivos recursos, ésta puede trabajar sobre sistemas operativos como Windows, Linux y MacOS.

- **Programación por bloques**

Una de las ventajas de GNU Radio sobre otros entornos de simulación es que el usuario puede desarrollar cualquier aplicación utilizando bloques jerárquicos, sin necesidad de conocer algún lenguaje de programación. Además, la configuración de cada bloque se lo realiza a través de la interfaz gráfica que proporciona el software.

- **Programación mediante Python o C++**

GNU Radio al ser una herramienta utilizada también por investigadores, permite la programación mediante código, útil para crear ciertos bloques que la plataforma no proporciona al usuario, los mismos que pueden ser creados mediante programación en Python o C++.

1.3.5.2 Instalación de GNU Radio

Como se había mencionado anteriormente, GNU Radio presenta varias ventajas, entre las que se tiene que el programa puede funcionar sobre cualquier sistema operativo, ya sea Windows, Linux, o IOS. Para el presente proyecto de titulación, se optó por GNU Radio para Windows, esto debido a que, es uno de los sistemas operativos más utilizado y conocido por docentes y estudiantes. A continuación, se muestra el procedimiento para la instalación de este software.

- Primero, se ingresa a la página propia de GNU Radio, para descargar el archivo ejecutable.
- Una vez descargado el archivo, se procede con la instalación, tal como se indica en las figuras 1.6 y 1.7.

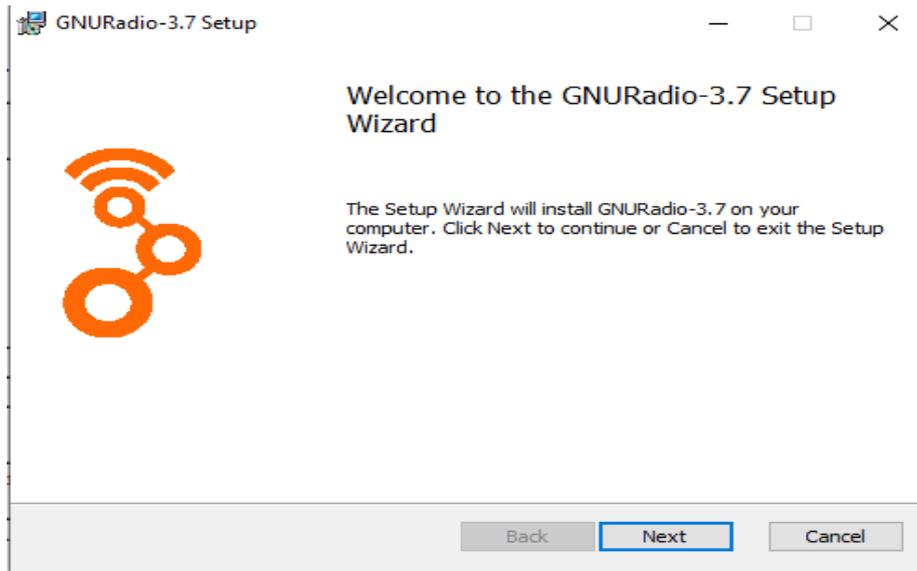


Figura 1.6. Cuadro de diálogo GNU Radio [2]

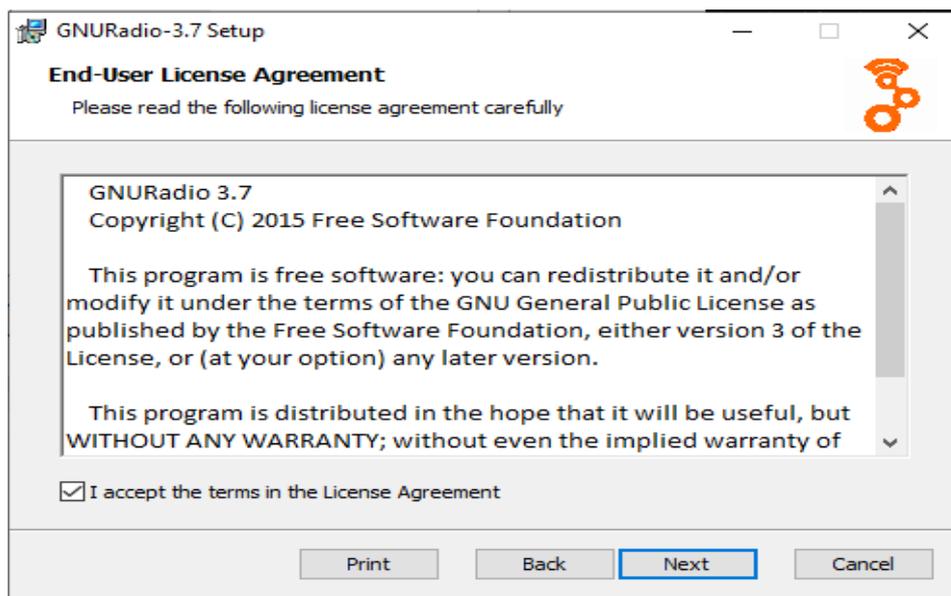


Figura 1.7. Cuadro de diálogo GNU Radio (licencia) [2]

Una vez realizado el proceso de instalación, el programa muestra la interfaz gráfica de GNU Radio, tal como se presenta en la figura 1.8.

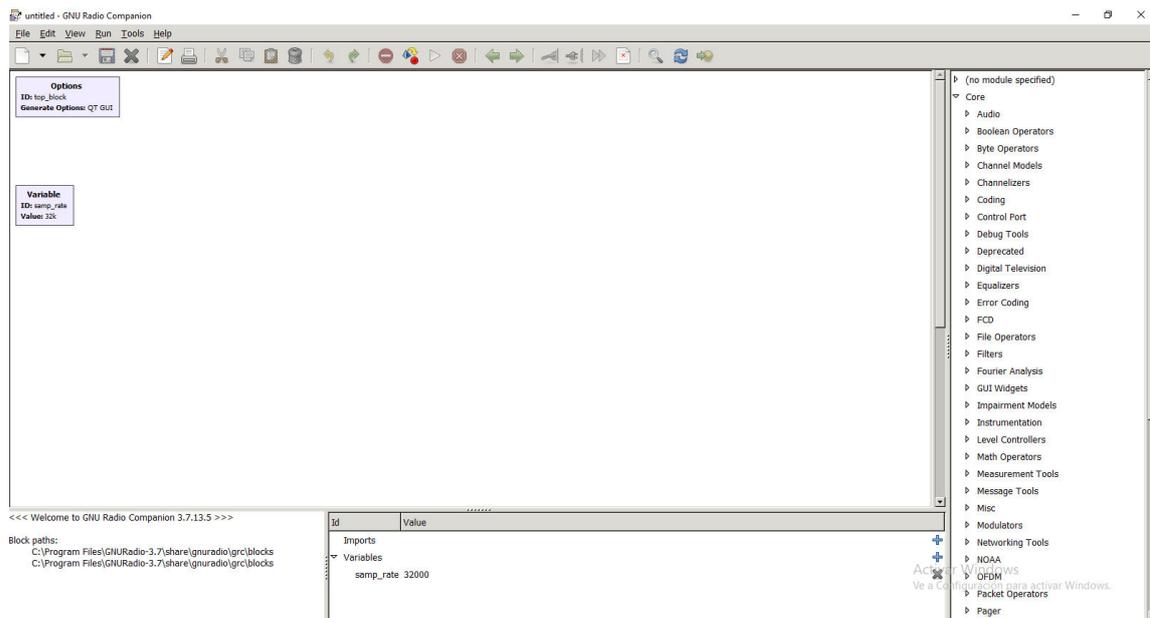


Figura 1.8. Interfaz gráfica de GNU Radio [2]

En la plataforma, existen diferentes versiones de GNU Radio, siendo la más reciente la versión 3.8.0.0. Sin embargo, para el presente proyecto se utilizará la versión 3.7.13.5, debido a que, las últimas versiones de software aún están en período de prueba y en el caso de GNU Radio, pueden carecer de bloques necesarios para el desarrollo del proyecto.

1.3.5.3 Programación en GNU Radio

Otra ventaja que posee GNU Radio es el tipo de programación que puede manejar, la cual puede ser de dos maneras; la primera mediante bloques, los cuales, son configurados mediante la propia interfaz gráfica; la segunda manera, creando los bloques a partir de código, configurando sus parámetros en el propio *script*.

En GNU Radio, se tienen diferentes tipos de bloques, propios del software; entre los principales se tienen:

- *Fuentes*: son bloques que poseen generadores de señales, generadores de ruido, generadores de señales de audio, entre los principales.
- *Procesamiento de señales*: apartado en el cual se encuentran los bloques que realizan una modificación de la señal que atraviesa por éstos; entre los más conocidos se tienen: moduladores, filtros, codificadores, operadores lógicos y aritméticos.
- *Sumideros*: también conocidos como *Sinks*, son instrumentos mediante los cuales se pueden visualizar los resultados generados de determinado proceso. Entre los principales sumideros, se tienen: osciloscopios, analizadores de espectros,

visualizador de constelaciones. Además, contienen los bloques utilizados para la conexión con los dispositivos USRP.

- *Controladores de gráficas:* son bloques utilizados para controlar diferentes parámetros de las gráficas en tiempo real, que permiten controlar parámetros tales como amplitud de señales y frecuencia de operación. Todos estos bloques serán explicados posteriormente en el capítulo 2.

Además, para trabajar con bloques y realizar las conexiones entre éstos, se debe tener en cuenta el tipo de datos que admite cada bloque; este aspecto será de vital importancia para el desarrollo de las prácticas de laboratorio. En la figura 1.9, se muestran los diferentes tipos de datos que maneja la plataforma.

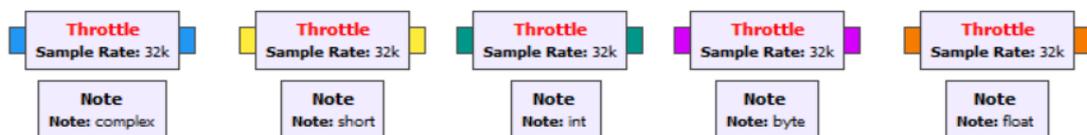


Figura 1.9. Tipos de datos de GNU Radio [2]

Sin embargo, existen bloques que no manejan el mismo tipo de datos, pero se necesita realizar una conexión entre ellos; con este propósito se recurre a bloques que realizan la conversión de datos. Entre estos bloques se tienen los mostrados en la figura 1.10.

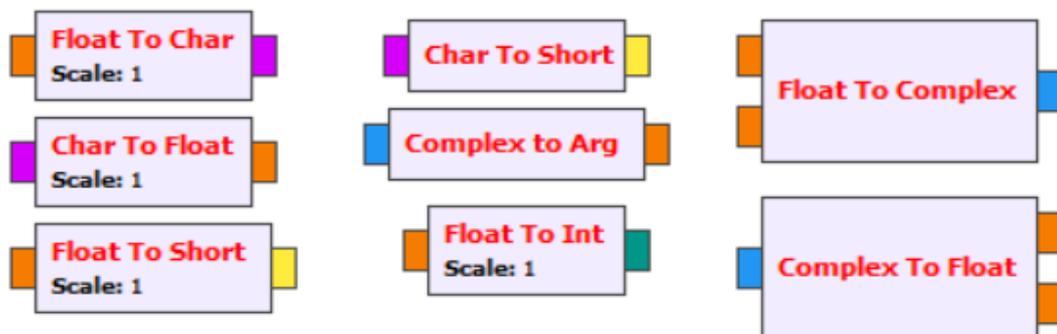


Figura 1.10. Tipos de conversores GNU Radio [2]

1.3.6 CONEXIÓN USRP A GNU Radio

Para realizar la conexión lógica y física entre el computador y el dispositivo USRP se deben tener en cuenta dos aspectos primordiales. En cuanto a la parte física se debe considerar lo siguiente:

- El ordenador al cual se conectará el dispositivo USRP debe tener un puerto Gigabit Ethernet, esto debido a las velocidades que maneja el dispositivo.

- Se debe realizar la conexión de la antena al USRP, ya que, en caso de no realizarse, el dispositivo puede sufrir algún daño.

Para la conexión de la parte lógica, se tomará en cuenta lo siguiente:

- El dispositivo USRP cuenta con una dirección IP por defecto, la cual es 192.168.10.2; esta a su vez, puede ser cambiada mediante una utilidad de LabVIEW (Figura 1.11). Sin embargo, para el presente proyecto, se mantendrá la dirección IP propia del dispositivo.

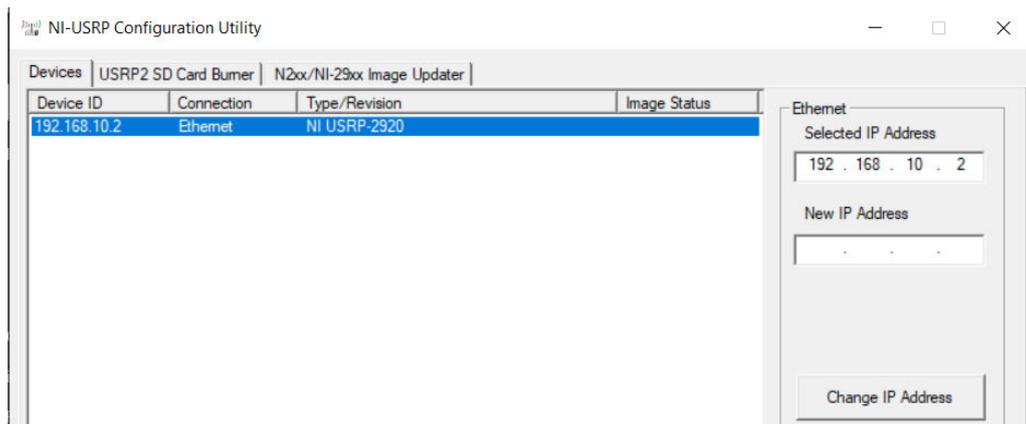


Figura 1.11. Cambio de IP de USRP NI 2920 [9]

- Ahora, para que exista una comunicación entre el dispositivo USRP y el ordenador, este último, debe tener una dirección IP perteneciente al mismo segmento de red del USRP. Por esta razón, se debe cambiar la IP del ordenador (Figura 1.12).

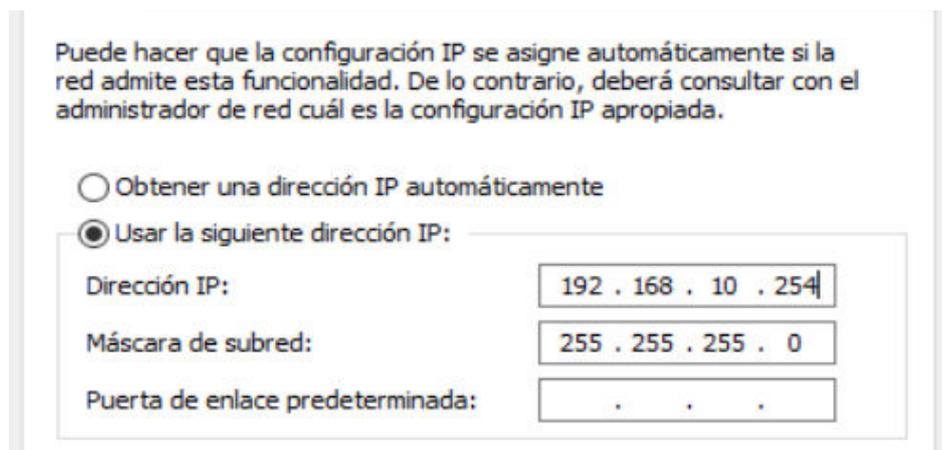


Figura 1.12. Configuración del adaptador de red de la PC

- Finalmente, para verificar la conexión entre el ordenador y el dispositivo USRP se realiza una prueba de conectividad entre éstos (Figura 1.13).

```

C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Versión 10.0.18362.295]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\lenovo>ping 192.168.10.2

Haciendo ping a 192.168.10.2 con 32 bytes de datos:
Respuesta desde 192.168.10.2: bytes=32 tiempo=2ms TTL=32
Respuesta desde 192.168.10.2: bytes=32 tiempo=1ms TTL=32
Respuesta desde 192.168.10.2: bytes=32 tiempo=1ms TTL=32
Respuesta desde 192.168.10.2: bytes=32 tiempo=1ms TTL=32

Estadísticas de ping para 192.168.10.2:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 1ms, Máximo = 2ms, Media = 1ms

C:\Users\lenovo>

```

Figura 1.13. Prueba de conectividad con USRP

Otra manera de verificar que exista conectividad entre los dispositivos es mediante la pantalla de comandos de GNU Radio, para lo cual se digita el comando `uhd_usrp_probe`, cuyo resultado se muestra en la figura 1.14.

```

GNURadio Command Prompt
setting gnuradio environment
Microsoft Windows [Versión 10.0.18362.295]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Program Files\GNURadio-3.7\bin>uhd_usrp_probe
[INFO] [UHD] Win32; Microsoft Visual C++ version 14.0; Boost_106000; UHD_3.14.1.HEAD-0-g5491b80e
[INFO] [USRP2] Opening a USRP2/N-Series device...
[INFO] [USRP2] Current recv frame size: 1472 bytes
[INFO] [USRP2] Current send frame size: 1472 bytes
[WARNING] [UDP] The MTU (1472) is larger than the FastSendDatagramThreshold (1024)!
This will negatively affect the transmit performance.
See the transport application notes for more detail.

[INFO] [USRP2] Detecting internal GPSDO...
[INFO] [GPS] No GPSDO found

-----
Device: USRP2 / N-Series Device
-----
Mboard: N210r4
hardware: 2577
product: 30194
mac-addr: 00:80:2f:21:fb:58
ip-addr: 192.168.10.2
subnet: 255.255.255.255
gateway: 255.255.255.255

```

Figura 1.14. Command Prompt GNU Radio [2]

1.3.7 FUNDAMENTOS DE COMUNICACIÓN DIGITAL

Para entender el concepto de Comunicación Digital, se deben previamente mencionar los siguientes términos relacionados:

- *Comunicación*: proceso por el cual se transfiere información desde un punto a otro [10].

- *Sistema de comunicaciones*: infraestructura que proporcionan el enlace para el envío de información entre la fuente y el destino [10] (Figura 1.15).

Por ende, un sistema de Comunicación Digital transfiere información digital desde una fuente digital hacia el destino [10].

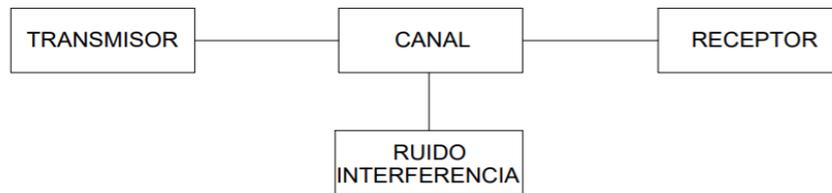


Figura 1.15. Sistema de comunicaciones

1.3.7.1 Canal de comunicaciones

Es el medio por el cual, la información viaja desde el punto origen al punto destino; sin embargo, existen diferentes anomalías que pueden afectar a la señal al cruzar por este medio [10]. Entre estas anomalías se tienen:

- Distorsión
- Atenuación
- Ruido
- Interferencia

Todas estas anomalías serán estudiadas con mayor detalle en el siguiente capítulo.

1.3.7.2 Digitalización de señales

Se refiere al proceso por el cual una señal analógica es transformada a formato digital, con el objetivo de facilitar su procesamiento, además de permitir que las señales sean más resistentes a las anomalías que introduce el canal [10].

1.3.7.3 Modos de transmisión

En comunicaciones de datos se tienen tres modos de gestionar el canal para transmitir la señal de información:

- *Simplex*: la comunicación se efectúa en un solo sentido [10].
- *Half Dúplex*: la comunicación se efectúa en los dos sentidos, pero no simultáneamente [10].
- *Full Dúplex*: la comunicación se efectúa en los dos sentidos, pero de manera simultánea [10].

1.3.7.4 Transmisión en banda base

Transmisión en banda base hace referencia a la transmisión de la información en forma digital, es decir sin realizar ningún tipo de modulación previa [10].

Esta técnica de transmisión se utiliza sobre medios de transmisión guiados. Además, requiere un tipo de codificación de línea (conversión de la señal digital de la fuente a una forma digital robusta) previa a la transmisión de la señal, de tal manera que, la señal se adapte al medio por el cual va a atravesar.

Entre algunos de los códigos de línea que se utilizan para este tipo de transmisión se tienen: NRZ, bipolar o AMI, Manchester, codificación diferencial.

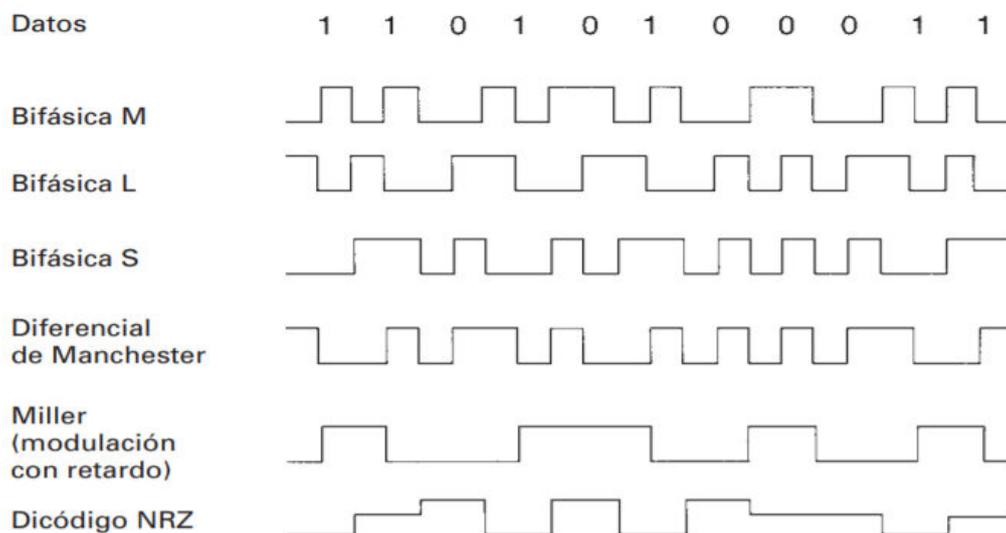


Figura 1.16. Códigos de línea [10]

1.3.7.5 Modulación Digital

También llamada transmisión en banda ancha es una técnica mediante la cual los datos son transmitidos mediante una onda portadora, constituida por una señal analógica; de modo que la información digital sea transmitida mediante la variación de uno o más parámetros de la señal analógica, es decir, de forma modulada [10].

Entre las diferentes técnicas de modulación digital se tienen las siguientes:

- **ASK (*Amplitude Shift Keying*)**

Es el tipo de modulación más sencilla, la cual consiste en representar los estados de la señal digital mediante una variación de amplitud de la señal portadora [10].

- **FSK (*Frequency Shift Keying*)**

Es otro tipo de modulación bastante sencilla, en la que cada estado de la señal digital le corresponde una determinada frecuencia de la señal portadora [10].

- **PSK (*Phase Shift Keying*)**

Es un tipo de modulación en la cual la fase de la señal portadora varía de acuerdo con los estados de la señal modulante de datos [10].

Como se pudo apreciar en este capítulo, los temas relacionados con Comunicación Digital se abordan muy brevemente. Sin embargo, cada uno de los tópicos relacionados con la asignatura se estudiarán mejor manera en el capítulo II, esto para una mejor comprensión de cada una de las prácticas de laboratorio.

2. METODOLOGÍA

En este capítulo se presentará las hojas guía correspondientes al desarrollo de cada una de las 9 prácticas de la asignatura de Comunicación Digital a ser utilizadas por el estudiante; es decir, se mostrará paso a paso la elaboración de los flujogramas correspondiente a cada tópico de las prácticas de laboratorio. Para ello, se utilizarán las herramientas antes mencionadas, es decir, el *software* GNU Radio y los dispositivos USRP NI 2920.

2.1 HOJAS GUÍA DE LAS PRÁCTICAS DE LABORATORIO

En este apartado se presentan las hojas guía correspondientes a cada práctica de laboratorio que realizará el estudiante. Las mismas contendrán una breve introducción teórica de cada tema, los requisitos para cada práctica, el procedimiento a seguir en el laboratorio y finalmente un reporte de lo realizado en la práctica.

2.1.1 PRÁCTICA 1: INTRODUCCIÓN A GNU Radio Y USRP NI 2920

2.1.1.1 Objetivo

Brindar a los estudiantes los conocimientos básicos de funcionamiento de la plataforma GNU Radio y de los equipos NI-USRP-2920.

2.1.1.2 Marco teórico

GNU Radio es un conjunto de herramientas de desarrollo de software gratuito y de código abierto, que suministra bloques de procesamiento de señales para implementar aplicaciones mediante software [2]. Entre sus ventajas están las de poder usar con dispositivos externos de RF para crear Radios Definidos por Software, o sin hardware simplemente para simular escenarios de prácticas.

GNU Radio al ser un software libre, es muy útil en el campo de investigación y en el campo de la docencia, ya que, mediante este conjunto de herramientas, tanto investigadores como estudiantes podemos comprender de mejor manera temas relacionados a las Comunicaciones Inalámbricas, Comunicaciones Digitales, Televisión Digital, entre otros.

LIBRERÍAS DISPONIBLES

Math

Math es una librería que la plataforma de GNU Radio ofrece al usuario, útil para realizar operaciones matemáticas en base a comandos en lenguaje Python [11]. Entre los comandos básicos que se pueden utilizar gracias a esta librería se encuentran:

- *Math.log* (*a*, *b*): siendo “a” el número del cual se obtendrá el logaritmo y “b” la base del logaritmo.
- *Math.pow* (*a*, *b*): permite encontrar la potencia de un número, siendo “a” la base y “b” el exponente.
- *Math.sin* (*a*): aplica la función seno a un ángulo determinado.

Para trabajar con esta librería se utiliza el siguiente comando:

```
Import math
```

Numpy

Numpy es uno de los paquetes pertenecientes a Python, que permite operar con matrices en los algoritmos diseñados.

Entre los comandos básicos que se pueden utilizar gracias a esta librería se tienen:

- *Np.array* ([[*a*], [*b*]]): comando utilizado para representar arreglos de matrices [12].
- *Np.arange* (*a*): comando utilizado para generar un vector entre 0 y 1 de “a” elementos [12].

Para trabajar con esta librería se utiliza el siguiente comando:

```
Import numpy
```

CREACIÓN DE BLOQUES

La creación de bloques es muy útil al trabajar con GNU Radio, esto debido a que la plataforma no posee algunos bloques para el procesamiento de señales, por lo que se puede crear estos bloques a partir de los disponibles en GNU Radio. Además, la creación de bloques provee una mejor organización de los diagramas de flujos, todo esto, para una mejor visualización de los algoritmos desarrollados.

Para la creación de bloques se tienen dos opciones; la primera mediante código y la segunda a partir de bloques jerárquicos. Sin embargo, la segunda opción permite crear nuevos bloques a partir de bloques propios de la plataforma, es decir sin necesidad de tener conocimiento de programación en Python.

A continuación, a modo de ejemplo, se detalla el proceso para crear un bloque a partir de bloques jerárquicos, el cual permite obtener el doble de un valor ingresado.

- Al iniciar GNU Radio, se procede a configurar el bloque *options* (Figura 2.1), el cual permite configurar el tipo de flujograma resultante de la siguiente manera:
 - o *Title*: título del flujograma creado.
 - o *Author*: usuario que desarrolla el flujograma.
 - o *Generate options*: *Hier block* para la creación de bloques.
 - o *Category*: ubicación destino del bloque creado.

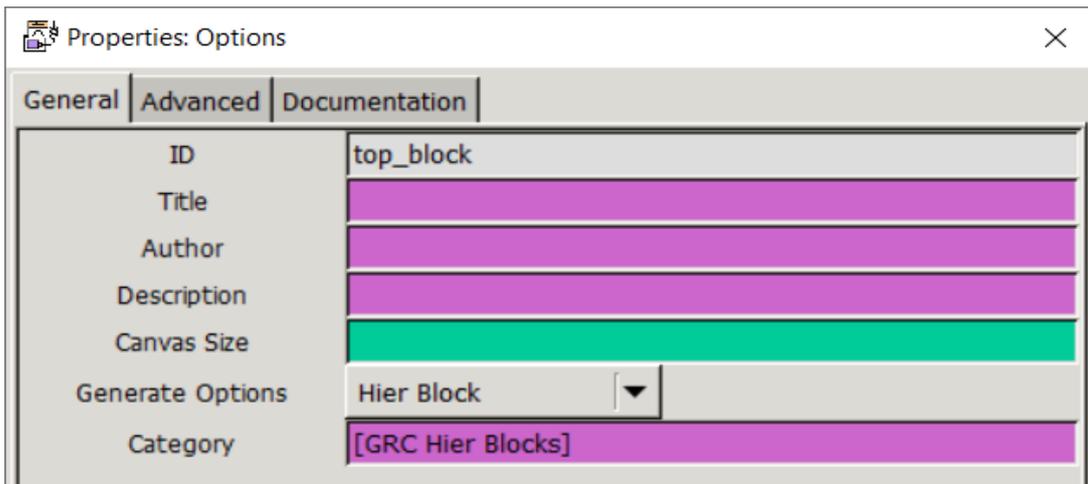


Figura 2.1. Configuración del Bloque Options GNU Radio

- Una vez configurada la plataforma, los bloques que se utilizarán son los siguientes:
 - o *Pad source*: bloque que representa la entrada del bloque a crear.
 - o *Pad sink*: bloque que representa la salida del bloque a crear.
 - o *Multiply Const*: bloque que permite obtener el doble de un valor de entrada.
- Finalmente, el bloque creado se muestra en la figura 2.2.

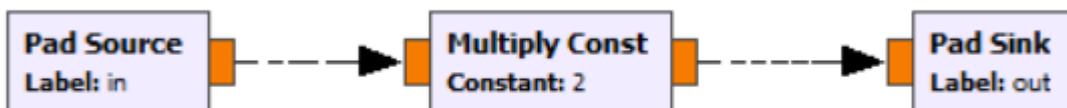


Figura 2.2. Ejemplo de creación de bloques

- Una vez creado el flujograma, se compilará y automáticamente se ubicará en la categoría descrita en el bloque *options*, tal como se muestra en la figura 2.3.



Figura 2.3. Ejemplo de bloque creado

Uso de variables

Una parte importante en el uso de GNU Radio es el uso de variables, ya que esto permite una mejor utilización de la plataforma y de los algoritmos desarrollados. Las variables permiten cambiar parte del flujograma sin necesidad de alterar todo el algoritmo, es decir, mediante su uso, se pueden variar diferentes parámetros del flujograma.

USRP NI 2920

La USRP NI 2920 (figura 2.4) es un dispositivo de radiofrecuencia que opera bajo la tecnología SDR (*Software Defined Radio*); se emplea en el campo de las comunicaciones digitales, comunicaciones analógicas, televisión digital, entre otros.

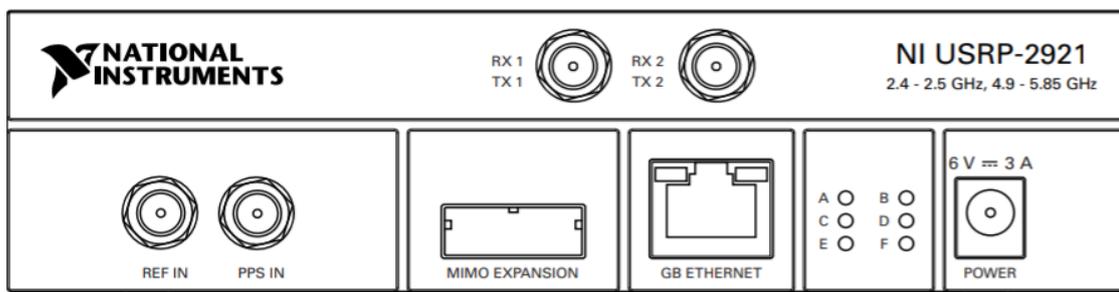


Figura 2.4. USRP NI-2920 (Vista frontal) [8]

El dispositivo USRP NI 2920 posee varias interfaces de control, las cuales serán de gran importancia para el desarrollo de diferentes aplicaciones:

- *TX1 RX1*: interfaz utilizada para transmisión o recepción de datos, la cual es configurada mediante software.
- *RX2*: interfaz utilizada únicamente para la recepción de datos.
- *MIMO EXPANSION*: interfaz utilizada para conectar dos dispositivos USRP mediante un cable MIMO.
- *GB ETHERNET*: interfaz utilizada para conectar el dispositivo USRP al puerto LAN de un computador.

2.1.1.3 Procedimiento

1. Crear un bloque que simule una calculadora básica, es decir, que realice la suma, resta, multiplicación y división entre dos valores ingresados por el usuario.
2. Crear un menú de usuario en el cual se muestren los resultados del numeral anterior en varias viñetas.

2.1.1.4 Informe

1. Elaborar un selector que permita elegir el tipo de operación a realizar y mostrarlo en una viñeta.
2. Elaborar un *Slider* que permita modificar el valor de los datos ingresados por el usuario.

2.1.2 PRÁCTICA 2: ANÁLISIS DE SEÑALES

2.1.2.1 Objetivo

Analizar los distintos parámetros de señales determinísticas, parámetros como período, frecuencia y amplitud, así como de operaciones entre señales.

2.1.2.2 Marco Teórico

El análisis de señales es de vital importancia en el campo de las comunicaciones, ya que permite comprender de mejor manera el proceso de transmisión y recepción de la información.

Una señal es definida como cualquier símbolo, gesto o ruido cuya función es transmitir información de un punto a otro [13]. En el campo de las comunicaciones digitales, la palabra señal implica una cantidad eléctrica (por ejemplo, voltaje) que posee alguna característica (por ejemplo, amplitud) que varía de manera impredecible [13].

- Señales periódicas

Llamadas así debido a que su forma de onda se repite de manera periódica, es decir, cada cierto intervalo de tiempo; en esta clase de señales se tiene, por ejemplo, a las señales sinusoidales. Un ejemplo de una señal periódica se observa en la Figura 2.5.

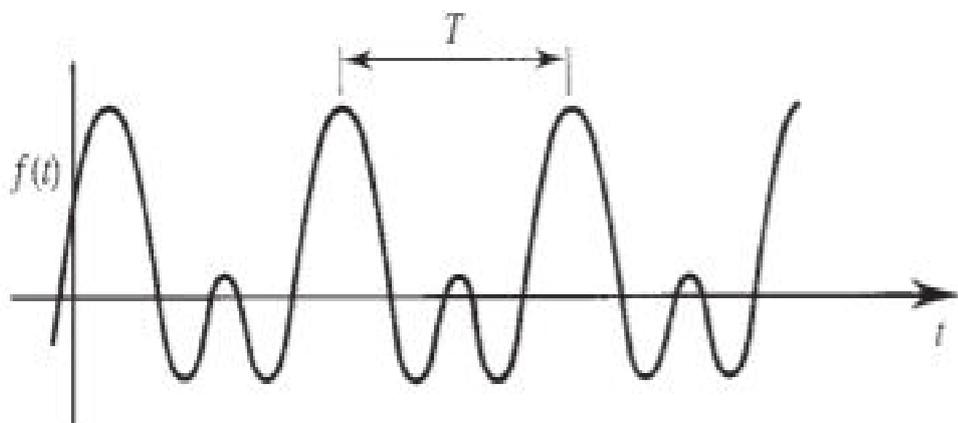


Figura 2.5. Señal periódica [13]

- **Señales no periódicas**

Son señales que, a diferencia de las periódicas, no poseen un periodo de repetición, es decir, su comportamiento es variable, no repetitivo (Figura 2.6).

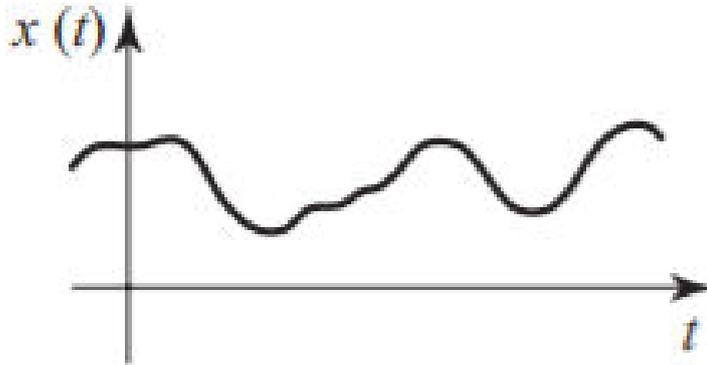


Figura 2.6. Señal no periódica [13]

Señales en el Dominio del tiempo y en el dominio de la frecuencia

- **Dominio del tiempo**

El dominio del tiempo es un término utilizado para describir el comportamiento de las señales respecto al tiempo. Un osciloscopio es uno de los instrumentos más utilizados para visualizar tanto la forma de onda como la amplitud de las señales, tal como se muestra en la figura 2.7.

- **Dominio de la frecuencia**

Si bien es cierto, al representar una señal en el dominio del tiempo, es posible visualizar parámetros tales como la amplitud o su fase, así como la forma de onda, no es posible visualizar sus componentes de frecuencia. Es por ello por lo que, el análisis de señales en el dominio de la frecuencia permite visualizar la (s) componente (s) de frecuencia que se están transmitiendo en la señal, como se muestra en la figura 2.8.

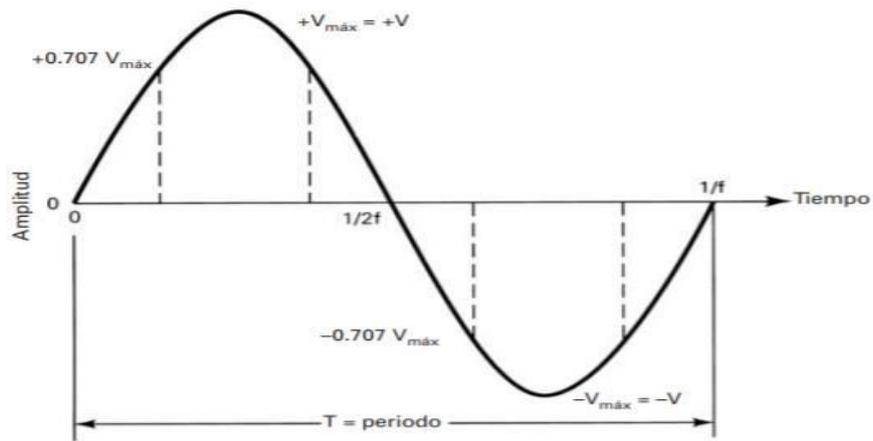


Figura 2.7. Señal sinusoidal en el dominio del tiempo [13]

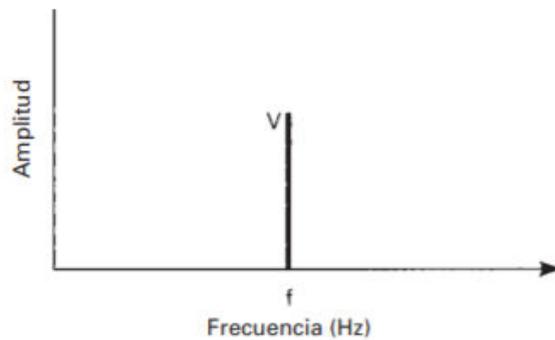


Figura 2.8. Componente de frecuencia de una senoide en el dominio de la frecuencia [13]

2.1.2.3 Trabajo Preparatorio

1. Consultar el modo de uso de los bloques UHD: USRP Sink y UHD: USRP Source.
2. Consultar las características y configuración de los bloques *WBFM Transmit* y *WBFM Receive*.
3. Consultar las herramientas básicas de GNU Radio para el análisis de señales, su modo de uso y su comparación entre ellas.
4. Consultar los bloques disponibles para operaciones entre señales, así como su configuración básica.

2.1.2.4 Procedimiento

1. Generar una onda sinusoidal de frecuencia F KHz, de amplitud A .
2. Generar una onda cuadrada de frecuencia F KHz, de amplitud A .

3. Mediante el uso del bloque *Rational Resampler*, cambiar la frecuencia de muestreo de las señales generadas en los numerales 1 y 2 a una frecuencia compatible con el equipo USRP.
4. Mediante el uso de los bloques *WBFM Transmit* y *WBFM Receive*, analizar las señales de los numerales anteriores recuperadas en el receptor.
5. Realizar el producto de las dos señales de los numerales 1 y 2. Posteriormente, realizar la transmisión y recepción de ésta, mediante los bloques *UHD: USRP Sink* y *UHD: USRP Source*.

2.1.2.5 Informe

1. Analizar los resultados obtenidos en los numerales 1 a 4 del Procedimiento. Variando el valor de la frecuencia y amplitud de las señales, esto, en la plataforma de GNU Radio, analizar el comportamiento de las señales mencionadas.
2. A partir del flujograma desarrollado en el Procedimiento, crear un menú interactivo que permita visualizar las señales de los apartados 1, 2 y 4 del procedimiento en el dominio del tiempo y la frecuencia. Anexar los resultados obtenidos.
3. Realizar un selector que permita escoger entre 4 opciones: señal 1, señal 2, suma de señales, producto de señales. Posteriormente anexarlo en el menú desarrollado en el numeral anterior. Anexar los resultados obtenidos.
4. Determinar teóricamente los coeficientes de Fourier de la señal indicada en la figura 2.9. Considerar una amplitud normalizada y un período igual a 1. Adjuntar el procedimiento del cálculo.

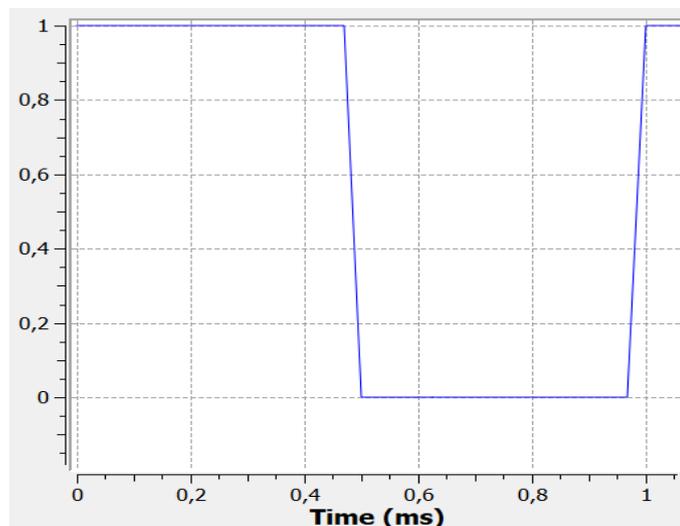


Figura 2.9. Señal Cuadrada

5. Graficar la señal del numeral anterior en GNU Radio, a partir del cálculo de los coeficientes de Fourier indicados en el numeral anterior. Utilizar los siguientes bloques:

- a. *Constant Source*
- b. *Signal Source*
- c. *Add y QT Time Sink*

Adjuntar el flujograma desarrollado.

2.1.3 PRÁCTICA 3: ANÁLISIS DEL CANAL DE COMUNICACIONES

2.1.3.1 Objetivo

Analizar las alteraciones que sufre una señal al atravesar el canal de comunicaciones.

2.1.3.2 Marco Teórico

CANAL

Medio físico por el cual se transmite la información e introduce diferentes anomalías que pueden afectar a la señal, tales como:

- Ruido
- Interferencia
- Atenuación

ATENUACIÓN

Consiste en la disminución de la potencia de la señal con la distancia, ocasionando de esta manera, que ésta no llegue a su destino de manera adecuada, cuando el receptor se encuentre a una distancia considerable del transmisor.

INTERFERENCIA

La interferencia de ondas de radio se produce siempre que se combinan dos o más ondas electromagnéticas (Figura 2.10) de tal manera que se degrada el funcionamiento del sistema [10]. Este fenómeno se produce debido a que pueden existir dos o más transmisores emitiendo en una misma frecuencia de trabajo.

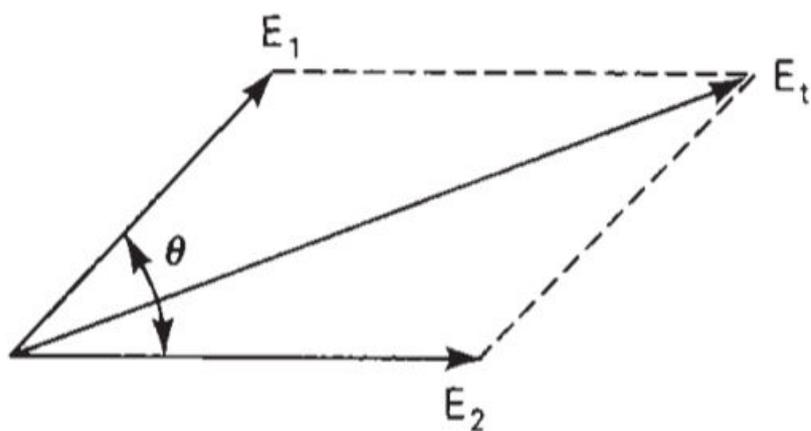


Figura 2.10. Suma vectorial de dos señales (interferencia) [10]

RUIDO

Es la anomalía más conocida en los sistemas de comunicaciones. Se define como ruido a cualquier energía eléctrica indeseable que queda entre la banda de paso de la señal [10]. Generalmente, el ruido se presenta como componentes de frecuencia no deseadas en la señal a transmitir. Un ejemplo de una señal con ruido se muestra en la figura 2.11

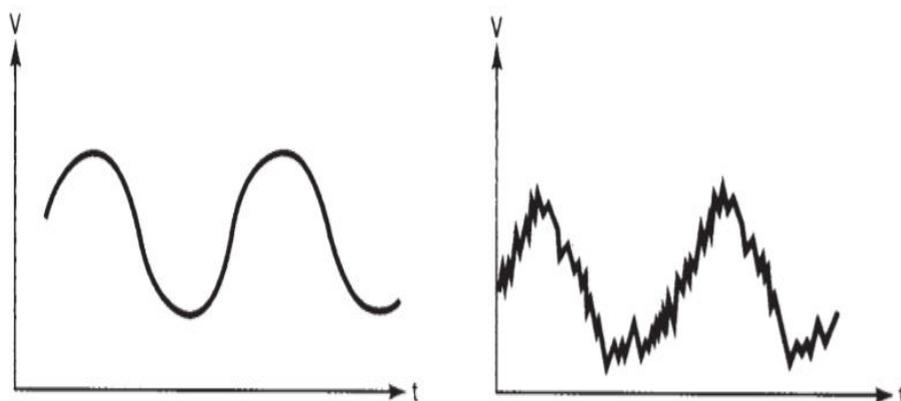


Figura 2.11. (a) señal sin ruido (b) señal con ruido [10]

2.1.3.3 Trabajo Preparatorio

1. Consultar el modo de uso del bloque *Channel Model* de GNU Radio.
2. Consultar el modo de uso del bloque *Rational Resampler*, configuración de parámetros, así como efectos negativos en el desarrollo de aplicaciones.
3. Preparar un archivo .wav (canción o archivo de voz grabado) a una frecuencia de muestreo de 11 KHz o 44.1 KHz.

2.1.3.4 Procedimiento

1. Generar en GNU Radio una señal sinusoidal de frecuencia F KHz y de amplitud A mediante el bloque *Signal Source*.
2. Mediante el uso del bloque *Rational Resampler*, cambiar la frecuencia de muestreo de la señal de audio a una frecuencia compatible con el dispositivo USRP.
3. Mediante el uso de los bloques *WBFM Transmit* y *Receiver*, transmitir y recibir la señal del numeral 1. Además, añadir ruido gaussiano a la señal transmitida para visualizar sus efectos en la señal recibida.
4. Realizar el proceso descrito en los numerales anteriores usando un archivo de audio .wav como fuente de información, así como el bloque *Rational Resampler*.

2.1.3.5 Informe

1. Analizar los resultados obtenidos durante la práctica.
2. A partir del flujograma desarrollado en el Procedimiento, elaborar un selector de señales, el cual permita escoger una señal sinusoidal o un archivo .wav como fuente de información.
3. Realizar una simulación en GNU Radio del flujograma desarrollado en el Procedimiento, utilizando el bloque *Channel Model* como canal de comunicaciones. Adjuntar y comentar los resultados obtenidos.

2.1.4 PRÁCTICA 4: DIGITALIZACIÓN DE SEÑALES

2.1.4.1 Objetivo

Analizar la digitalización de señales y sus conceptos asociados como: muestreo, cuantización y codificación.

2.1.4.2 Marco Teórico

Digitalización

Digitalización es el proceso mediante el cual una señal analógica se convierte en una señal digital, como por ejemplo la señal correspondiente a la voz se convierte en una serie de pulsos de amplitud A o 0 . Estos pulsos son denominados bits; de este modo, la información puede transmitirse como un tren de pulsos a través de un medio de transmisión.

El proceso de convertir señales analógicas en digitales se da con el fin de hacer a la señal más resistente a ruido y a la interferencia. Este proceso consta de 3 etapas principales:

- Muestreo

- Cuantificación
- Codificación

MUESTREO

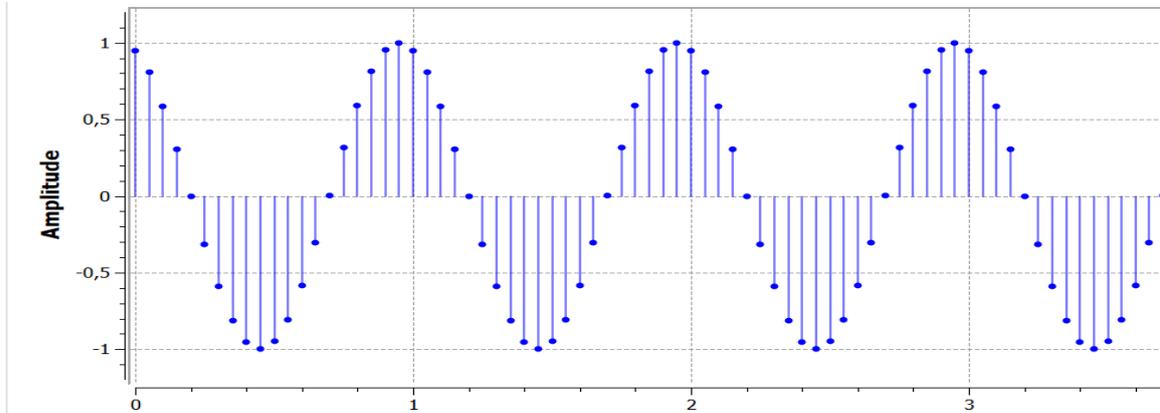


Figura 2.12. Muestreo de Señales

Consiste en obtener muestras de la señal analógica a procesar en determinados instantes de tiempo, convirtiendo a la señal en una secuencia de muestras uniformemente espaciadas en el tiempo (Figura 2.12).

Sin embargo, las muestras no pueden estar separadas cualquier intervalo de tiempo. El intervalo de tiempo que separa cada muestra debe cumplir la Ecuación 2.1.

$$T_s \leq 1/(2 * f_{max}) \quad (2.1)$$

Siendo f_{max} , la frecuencia máxima de la señal a procesar. Esta ecuación es conocida como el Teorema de Muestreo, el cual establece que las muestras de una señal se deben tomar a una frecuencia mayor o igual a dos veces la frecuencia máxima de la señal a procesar, como se puede ver en la figura 2.13.

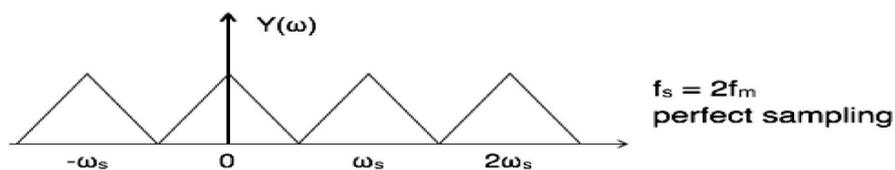


Figura 2.13. Teorema de Muestreo

CUANTIFICACIÓN

Es el proceso por el cual, una muestra de la señal, que puede tener cualquier valor, se hace corresponder a un determinado valor dentro de un conjunto limitado de valores llamados *intervalos de cuantificación*, como se muestra en la figura 2.14.

Ahora, el número de intervalos de cuantificación está definido por el número de bits que van a ser utilizados para representar a cada muestra, como indica la Ecuación 2.2.

$$\#Q.I = 2^n \quad (2.2)$$

Siendo n el número de bits necesarios para representar cada muestra.

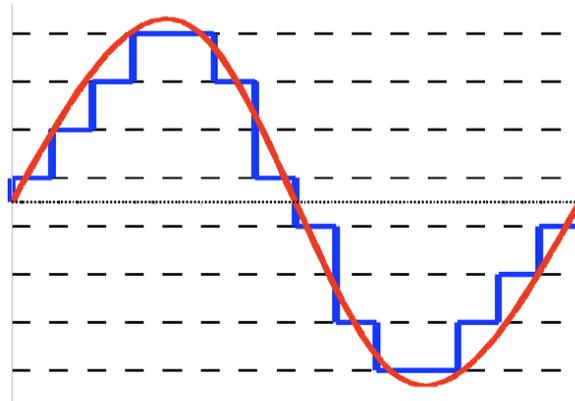


Figura 2.14. Cuantificación de Señales

CODIFICACIÓN

Proceso en el cual, se asigna una palabra binaria a cada intervalo de cuantificación; de esta manera se representa cada muestra con una palabra PCM de n bits, como se ve en la figura 2.15

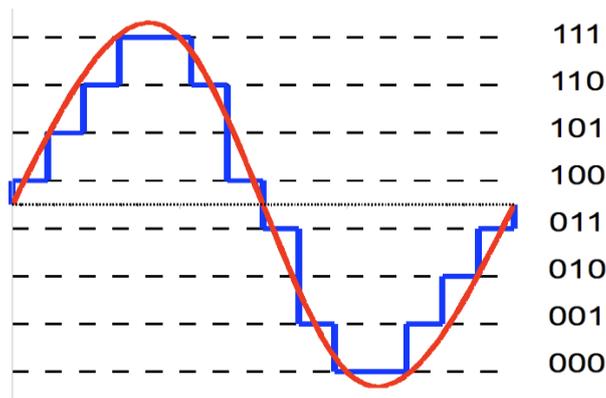


Figura 2.15. Codificación de Señales

2.1.4.3 Trabajo Preparatorio

1. Consultar el modo de uso de los bloques: *Quantizer*, *Unpack K bits* y *Pack K bits*.
2. Consultar el concepto de submuestreo y sobremuestreo, así como sus efectos en la digitalización de las señales.

2.1.4.4 Procedimiento

1. Generar en GNU Radio una onda sinusoidal de frecuencia F KHz, de amplitud A . Posteriormente, muestrear la señal a una frecuencia N veces la frecuencia de la señal. Considere N un número entero.
2. Mediante el uso del bloque *Quantizer* y *Unpack k bits* obtener gráficas de la señal cuantificada y codificada. Previo a esto, configurar el número de bits y la frecuencia de muestreo mediante el uso de variables.
3. Desarrollar un digitalizador de voz mediante el uso de los bloques *Unpack K bits* y *Multiply Const*.
 - a. Bloque *Multiply const*: cuantificador de 2^n niveles.
 - b. Bloque *Unpack K bits*: codificador de señales en formato PCM de k bits.

2.1.4.5 Informe

1. Analizar los resultados obtenidos de la digitalización de señales, variando el número de bits y utilizando una secuencia de números como fuente de información (se recomienda 10 números como máximo), mediante el uso del bloque *Vector Source*.
2. En base al flujograma desarrollado en el Procedimiento, elaborar un menú de usuario, el cual permita visualizar la señal en sus diferentes etapas, es decir: muestreo, cuantificación y codificación.
3. Crear dos entradas de datos al menú realizado en el numeral 2, éstas deben permitir ingresar al usuario la secuencia de datos a digitalizar y el número de bits para la codificación y cuantificación.
4. Además del menú de usuario desarrollado en los numerales anteriores, crear un *slider* que permita variar la frecuencia de muestreo de la señal a digitalizar, esto con el objetivo de analizar el comportamiento de la señal a diferentes frecuencias de muestreo.

2.1.5 PRÁCTICA 5: CUANTIFICACIÓN NO UNIFORME

2.1.5.1 Objetivo

Analizar la cuantización no uniforme dentro del proceso de digitalización de señales.

2.1.5.2 Marco Teórico

Es una técnica de cuantificación, en la cual, con el fin de tener una mejor relación SNQ (Señal a Ruido de cuantificación), es necesario tener niveles de cuantización más cercanos

(más pequeños) para niveles de señal de menor amplitud y niveles más espaciados para muestras de mayor amplitud.

Una cuantización no uniforme se logra enviando la señal a través de dispositivos no lineales, seguido de un cuantizador uniforme, es muy común el uso de compansores logarítmicos.

Dos de los esquemas de cuantificación no uniforme más utilizados en la actualidad vienen definidos en la recomendación de la UIT G.711.

Cuantificador LEY A

Es un esquema de cuantificación no uniforme utilizado en Europa, África y Sudamérica, el mismo que se puede aproximar por una característica de transferencia de 13 segmentos. Dicha característica, posee una respuesta lineal para valores inferiores a cierto umbral; y, una respuesta no lineal para valores superiores a ese margen.

Este cuantificador utiliza 256 intervalos de cuantificación, es decir, cada palabra PCM utiliza 8 bits para su representación.

El intervalo de cuantificación para determinada muestra se obtiene mediante la Ecuación 2.3.

$$F_A(x) = \begin{cases} \frac{A \cdot |x|}{1 + \ln(A)}, & 0 \leq x < \frac{1}{A} \\ \frac{1 + \ln(A \cdot |x|)}{1 + \ln(A)}, & \frac{1}{A} \leq x \leq 1 \end{cases} \quad (2.3)$$

Siendo:

A : constante de cuantificación = 87.6 $|x|$: muestra normalizada

La curva que corresponde a la ley A se muestra en la figura 2.16.

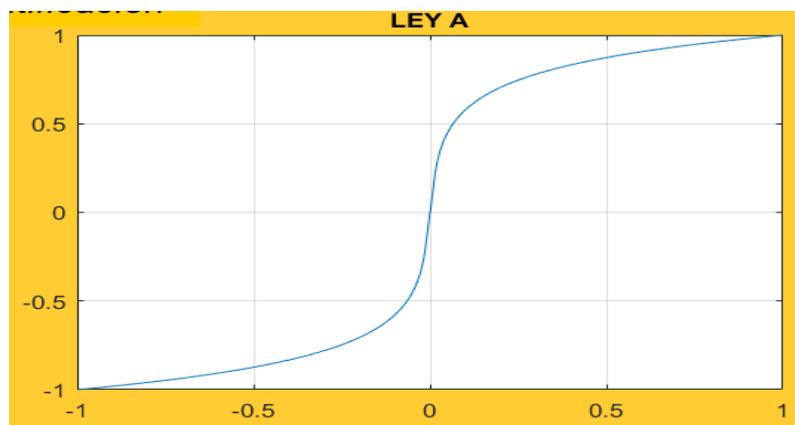


Figura 2.16. Gráfica ley A

Cuantificador LEY μ

Es un esquema de cuantificación no uniforme utilizado en EE. UU., Japón y Canadá, el mismo que se puede aproximar por una característica de transferencia de 15 segmentos. Dicha característica, posee una respuesta logarítmica para todos los valores de la señal a comprimir.

Este cuantificador utiliza 256 intervalos de cuantificación, es decir, a cada palabra PCM se le asignan 8 bits para su representación. La curva de cuantificación que corresponde a esta ley se muestra en la figura 2.17.

La ecuación 2.4 indica el intervalo de cuantificación de acuerdo con la ley μ que corresponde a un determinado valor de muestra.

$$f(x) = (\ln(1 + \mu * |x|))/\ln(1 + \mu) \quad (2.4)$$

Siendo:

μ : constante de cuantificación = 255 $|x|$: muestra normalizada

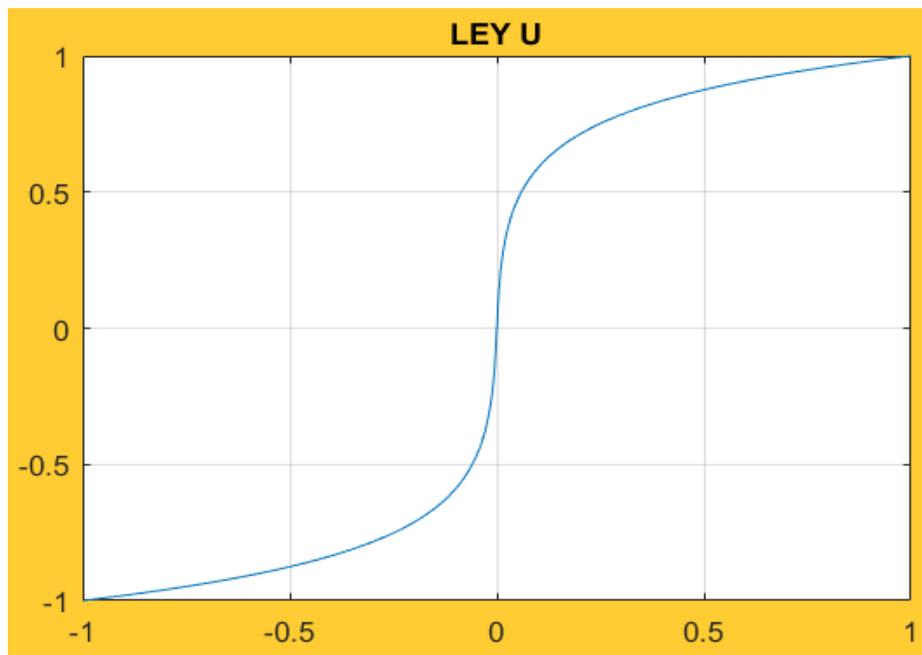


Figura 2.17. Gráfica Ley μ

2.1.5.3 Trabajo Preparatorio

1. Consultar el método de determinación del intervalo de cuantificación por aproximación según la ley A.

2. Consultar el proceso de determinación del intervalo de cuantificación a partir de un valor de muestra y un valor máximo. Además, consultar el modo de cálculo del porcentaje de error de cuantificación en el receptor.
3. Calcular el intervalo de cuantificación al cual pertenece la muestra $x = 0.481$ con un valor máximo de muestra $m = 1$. Adjuntar el proceso.
4. Consultar los comandos básicos de las librerías *Math* y *Numpy* de Python, para operaciones matemáticas y operaciones entre matrices.

2.1.5.4 Procedimiento

1. Importar las librerías *Numpy* y *Math*, necesarias para el cálculo de los distintos parámetros de la cuantificación, mediante el bloque *import* de GNU Radio.
2. Asignar variables para el valor de la muestra y el valor máximo, así como para las constantes de cuantificación. Posteriormente, calcular el valor de la muestra normalizada, de acuerdo con lo consultado en el trabajo preparatorio.
3. Calcular las $F(X)$ de acuerdo con la ley A, para ello es necesario el uso de comandos de la librería *Math*, consultados en el trabajo preparatorio. Realizar los cálculos haciendo el uso de variables.
4. En base al proceso consultado en el trabajo preparatorio, determinar el intervalo de cuantificación correspondiente a la muestra. Además, realice el cálculo del porcentaje de error en recepción.

2.1.5.5 Informe

1. Analizar los resultados obtenidos del procedimiento del cálculo del intervalo de cuantificación y compararlos con los obtenidos en el trabajo preparatorio.
2. Realizar el mismo procedimiento indicado en la práctica enfocado en la cuantificación según la ley μ .
3. En base al flujograma realizado en la práctica, realizar un graficador de las curvas de ley A y ley μ , utilizando bloques jerárquicos. Además, presentar sus resultados en una sola gráfica. Para ello, se utilizarán los bloques:
 - *Log*: bloque utilizado para graficar las curvas de ley A y ley μ .
 - *Divide*: bloque utilizado para realizar las operaciones correspondientes a la $F(X)$ de ley A y ley μ .
 - *Pad Sink*: Bloque utilizado como salida del bloque creado.

Adjuntar el flujograma del bloque graficador realizado.

4. Elaborar un menú de usuario que permita ingresar diferentes parámetros para el cálculo del intervalo de cuantificación, tales como: valor de muestra, valor máximo, constante de cuantificación; el flujograma debe mostrar mediante viñetas los resultados de las curvas de la ley A y la ley μ , así como, los resultados del intervalo de cuantificación. Adjuntar el flujograma desarrollado.

2.1.6 PRÁCTICA 6: MODULACIÓN M-PSK

2.1.6.1 Objetivo

Comprender el proceso de Modulación Digital de múltiples estados. Caso de estudio: Modulación BPSK (*Binary Phase Shift Key*).

2.1.6.2 Marco Teórico

MODULACIÓN DIGITAL

Modulación digital, o transmisión en banda ancha, es una técnica en la cual los datos son transmitidos usando una onda portadora, a la cual se modula de acuerdo con la señal digital que se quiere transmitir. Es decir, la modulación digital permite transmitir la información digital en una portadora analógica.

La característica fundamental de esta clase de modulación es que, a diferencia de la modulación analógica, la fuente de información debe ser digital; es decir, la señal de datos debe ser un flujo de bits. Ahora, para hablar de una modulación de M estados se debe tener en cuenta la forma en la cual se agrupan los bits en un estado M-PSK, tal como se muestra en la ecuación 2.5.

$$M = 2^n \quad (2.5)$$

Siendo n, el número de bits agrupados por estado. Por lo tanto, se tienen modulaciones de M estados, dependiendo del número de bits que tenga cada estado. Existen tres clases principales de modulación digital: ASK, FSK y PSK.

MODULACIÓN POR DESPLAZAMIENTO DE AMPLITUD (ASK)

Modulación en la cual la onda portadora varía su amplitud de acuerdo con la señal de información. Es decir, para la modulación de amplitud de dos estados, la portadora toma valores de A o 0 (Figura 2.18), dependiendo de la señal de información. Sin embargo, pueden existir casos en los cuales la señal tome M valores de amplitud en función de la señal de información, a este caso se le conoce como modulación M-ASK.

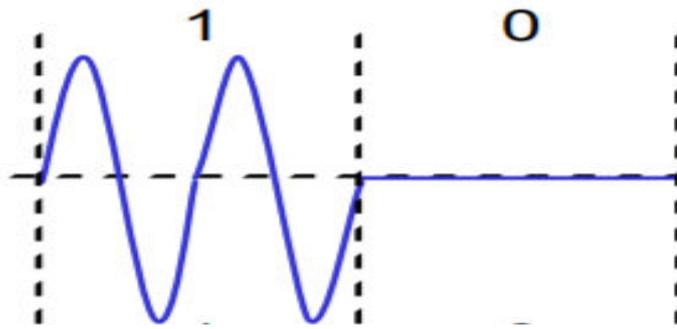


Figura 2.18. Modulación ASK

MODULACIÓN POR DESPLAZAMIENTO DE FRECUENCIA (FSK)

Modulación en la cual la onda portadora varía su frecuencia de acuerdo con la señal de información. Es decir, para la modulación de frecuencia de dos estados, la portadora toma valores de f_1 o f_2 (Figura 2.19), dependiendo de la señal de información. Sin embargo, pueden existir casos en los cuales la señal tome M valores de frecuencia en función de la señal de información, a este caso se le conoce como modulación M-FSK.

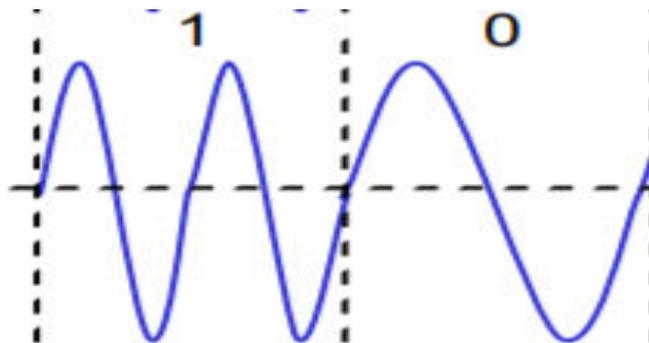


Figura 2.19. Modulación FSK

MODULACIÓN POR DESPLAZAMIENTO DE FASE (PSK)

Modulación en la cual la onda portadora varía su fase de acuerdo con la señal de información, en tanto que, la amplitud y la frecuencia permanecen constantes. El caso más típico es el modulador BPSK, en el cual, la portadora toma dos valores de fase, siendo la separación entre fases adyacentes de 180° (Figura 2.20). La separación entre fases adyacentes se la determina de acuerdo con la ecuación 2.6.

$$\Delta\theta = 2\pi/M = 2\pi/2^n \quad (2.6)$$

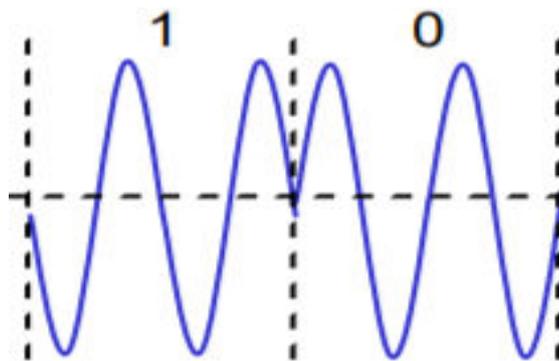


Figura 2.20. Modulación BPSK

BLOQUES DE GNU Radio PARA IMPLEMENTAR MODULACIÓN DIGITAL

Para la implementación de cada una de las etapas del modulador digital MPSK, se tomarán en cuenta diferentes bloques, tanto para digitalización de señales, como para la modulación. A manera de ejemplo, se presenta una modulación BPSK y una fuente de información 10101010 generada mediante el bloque *Vector Source*.

Digitalización

1. Packed to unpacked

- a. Al ser una modulación BPSK, se coloca en *Bits per Chunk* el número de bits que ocupa un estado BPSK, en este caso 1 (Figura 2.21); este bloque toma la señal entrante y la transforma en una señal de pulsos con valores de amplitud entre 0 y $M - 1$ (Figura 2.22). Por ejemplo, se tiene el número 170 como entrada y $M = 2$ (2 – PSK).

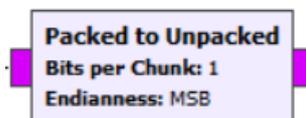


Figura 2.21. Bloque *Packed to unpacked*

$170 = 10101010$ (*representacion binaria*)

$170 = 1 - 0 - 1 - 0 - 1 - 0 - 1 - 0$

Por lo tanto, la salida del bloque representa cada bit en formato de byte, es decir:

$170 = 00000001\ 00000000\ 00000001\ 00000000\ 00000001\ 00000000\ 00000001\ 00000000$

$170 = 1 - 0 - 1 - 0 - 1 - 0 - 1 - 0$

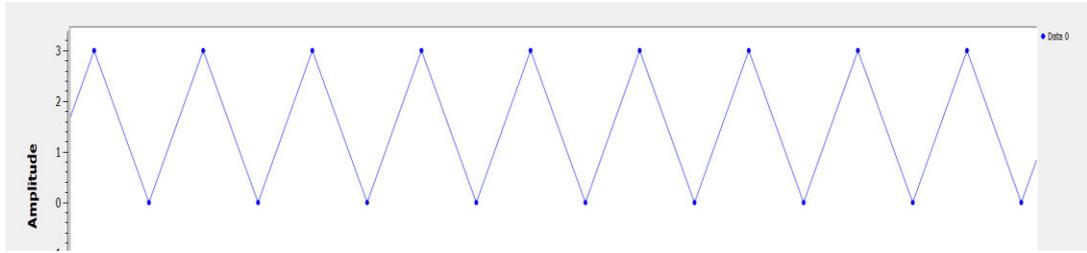


Figura 2.22. Resultado de Packed to unpacked

2. *Map*

- a. Realiza un mapeo de la señal entrante a los valores del vector de mapeo, es decir mapea los símbolos de binario a código Gray (figura 2.23), en el caso de BPSK los valores de código Gray son 0 y 1, como se muestra en la figura 2.24.

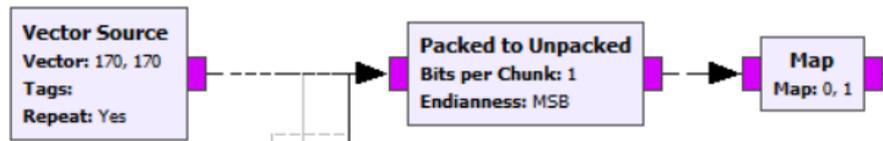


Figura 2.23. Uso del bloque Map

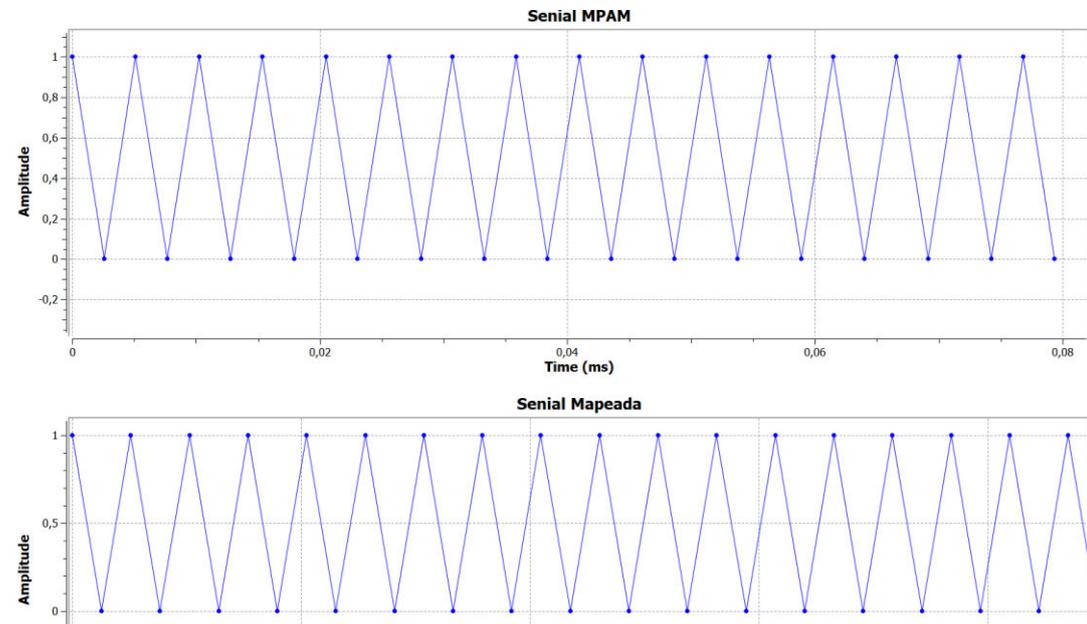


Figura 2.24. Resultado del bloque Map

Como se puede apreciar en las figuras, la señal antes y después del mapeo son iguales, esto debido a que, los valores 0 y 1 en código Gray son los mismos en código binario.

Codificación

3. Differential Encoder

- a. Realiza una codificación diferencial NRZ-I, la cual indica que la codificación de un símbolo depende del estado anterior, debido a que es un modulador B-PSK, es decir, de 2 estados, se tiene un codificador de 2 estados. El uso de éste se detalla en la figura 2.25.

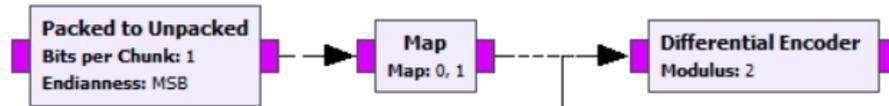


Figura 2.25. Uso del bloque Differential Encoder

Posteriormente, el resultado se presenta en las figuras 2.26 y 2.27.

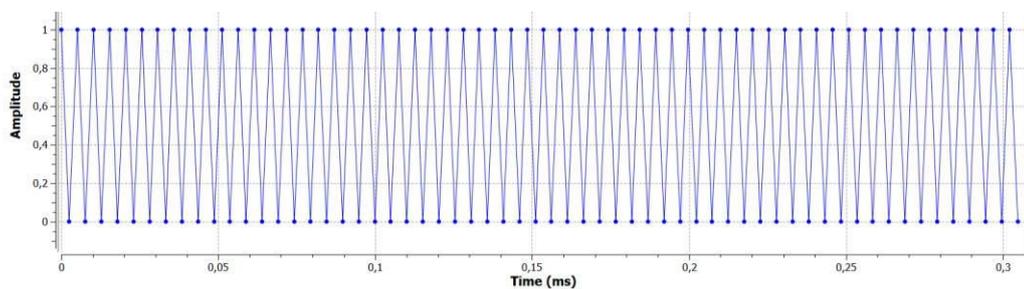


Figura 2.26. Señal antes de codificar

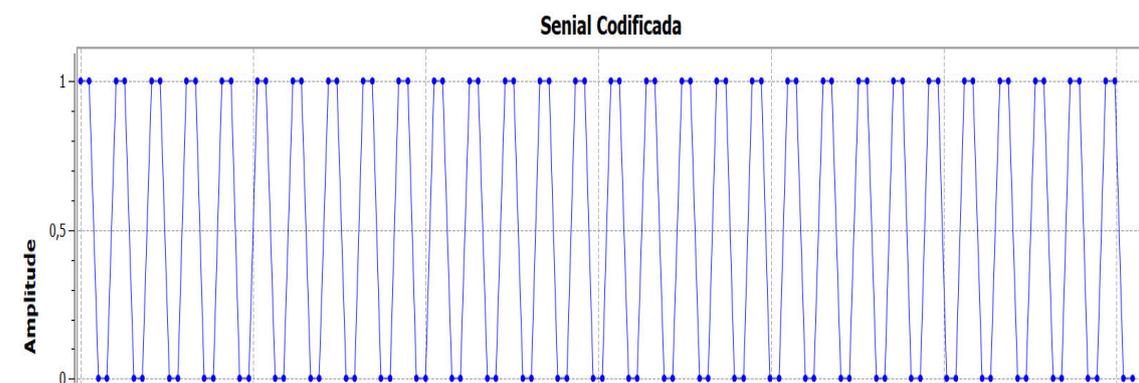


Figura 2.27. Resultado de codificación diferencial

Modulación

4. Chunks to symbols

- a. Este bloque realiza un mapeo de los datos desempaquetados a un conjunto de símbolos de constelación, es decir, traduce los valores de amplitud recibidos a símbolos complejos de acuerdo con los puntos de constelación.

Tabla 2.1. Asignación de puntos de constelación

Información (binaria)	Señal MPAM	Puntos de constelación
0	0	-1
1	1	1

A manera de ejemplo, se tiene la tabla 2.1 y su correspondiente asignación de puntos de constelación. Los puntos de constelación se asignan de acuerdo con el diagrama de constelación de la figura 2.28.

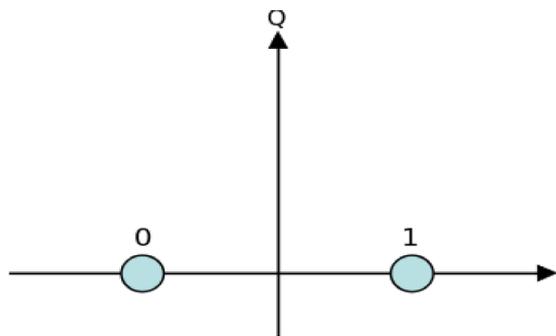


Figura 2.28. Diagrama de constelación BPSK

- b. Entre los parámetros a configurar, se tiene el parámetro *Symbol Table* (figura 2.29), el cual se puede ingresar manualmente o importar desde una variable mediante el bloque *variable*. A manera de ejemplo, se tiene BPSK.
 - i. Ingreso manual: 1,-1.
 - ii. Ingreso desde variable: `variable_name.points()`

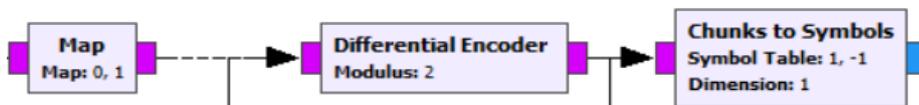


Figura 2.29. Uso del bloque *Chunks to Symbol*

Finalmente, se muestra el resultado del bloque en la figura 2.30.

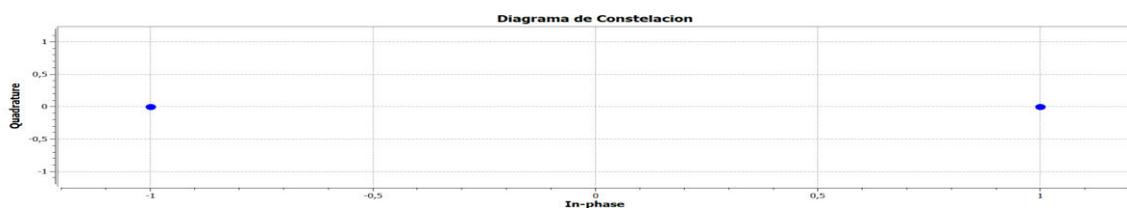


Figura 2.30. Resultado del bloque *Chunks to Symbol*

Hasta este punto, se tiene la señal resultante del bloque *Chunks to symbols* conocida como envolvente compleja, o en ocasiones denominada señal I Q. Sin embargo, ésta no es la señal que se conoce como modulada; ésta es generada mediante el USRP, el mismo que multiplica la envolvente compleja con una onda portadora. Sin embargo, la frecuencia a la que se transmite la señal modulada es demasiado elevada, por lo que para observar una aproximación de ésta se utiliza el bloque *Up_Converter* [14], el cual realiza una aproximación de lo que el USRP transmite, pero a una frecuencia que puede ser observada por el ojo humano.

A manera de ejemplo, en la figura 2.31 se visualiza una cadena de siete 0L y un 1L, con una aproximación de la onda modulada BPSK.

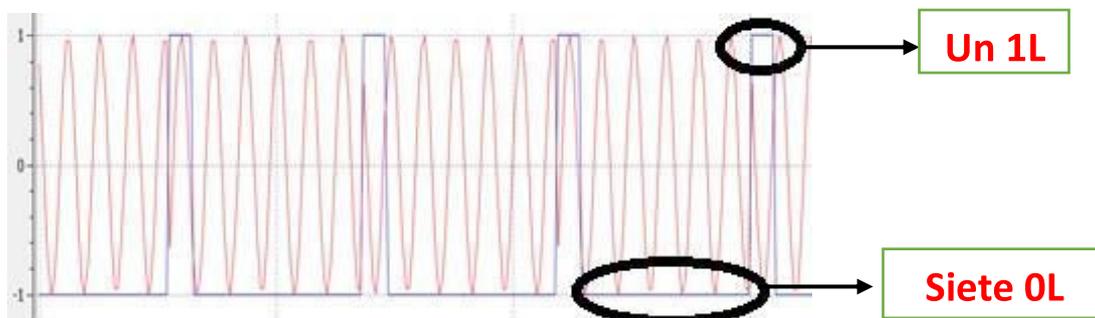


Figura 2.31. Aproximación de la señal modulada mediante el bloque *Up_Converter*

Finalmente, se tendría el modulador completo mediante la conexión del USRP con el computador que tiene GNU Radio.

Adaptación de la señal a USRP

Un aspecto importante para la transmisión de la señal mediante el USRP es la forma de la señal, en GNU Radio las señales que se manejan son en banda base, por lo que, se necesita convertir la señal a un formato compatible con el equipo, para ello se utiliza el bloque de adaptación de la señal *Adaptador GNU USRP* (Figura 2.32).



Figura 2.32. Bloque *Adaptador GNU USRP*

Posteriormente, se tiene la señal adaptada y el espectro de la señal BPSK, como muestran las figuras 2.33 y 2.34.

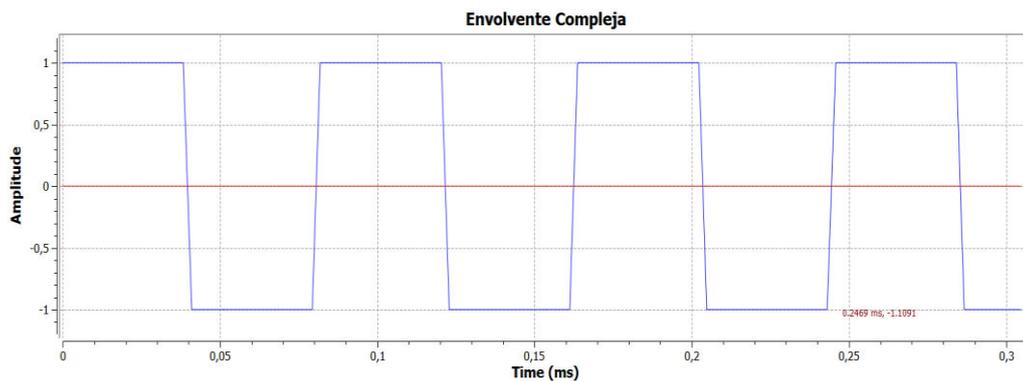


Figura 2.33. Señal adaptada mediante el bloque *Adaptador GNU USRP*

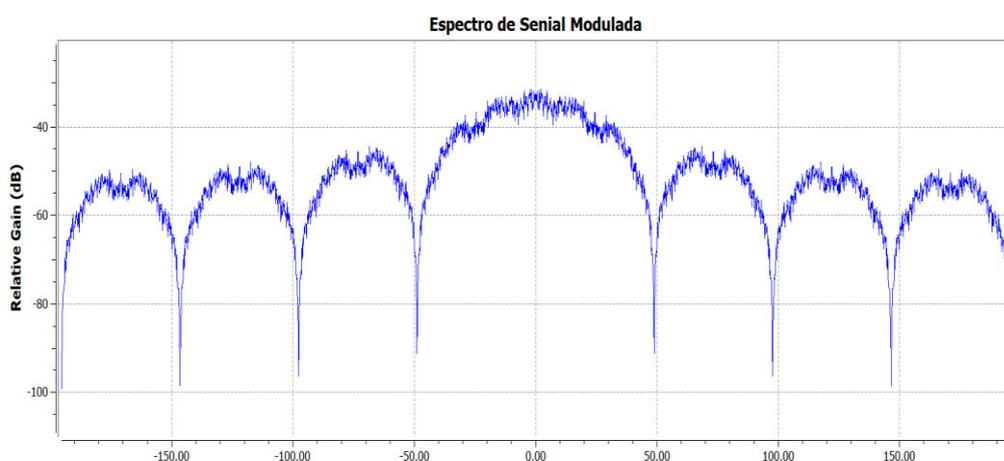


Figura 2.34. Espectro de señal BPSK

2.1.6.3 Trabajo Preparatorio

1. Preparar un archivo de texto, esto para verificar el correcto funcionamiento del sistema.
2. Consultar los comandos de la librería *digital* de Python para modulaciones digitales.
3. Consultar el modo de uso del bloque *Interpolating FIR Filter*.

2.1.6.4 Procedimiento

1. Cargar el archivo de texto preparado en el trabajo preparatorio, mediante el bloque de GNU Radio *File Source*. Posteriormente, digitalizar el archivo previo a la modulación utilizando el bloque *Packed to Unpacked*.
2. Importar las librerías *gr*, *digital* y *math*, necesarias para el desarrollo de la práctica.
3. Crear un modulador digital BPSK a partir de bloques jerárquicos, siguiendo las diferentes etapas que se indican en el diagrama de bloques de la figura 2.35.

Posteriormente, crear variables que indiquen el tipo de modulación y el número de símbolos.



Figura 2.35. Diagrama de bloques del modulador a implementar

4. Mediante el uso de visualizadores, mostrar los resultados de la señal original, la señal digitalizada, el diagrama de constelación de la onda modulada y una aproximación de la onda modulada RF.

2.1.6.5 Informe

1. Analizar los resultados obtenidos usando un archivo de audio como fuente de información, mediante el uso del bloque *Wav File Source*. Además, añadirle una fuente de ruido gaussiano y visualizar el comportamiento de la señal.
2. A partir del flujograma desarrollado en el Procedimiento, crear un menú interactivo con el usuario, el cual debe mostrar las siguientes señales: señal original, señal digitalizada, señal modulada, diagrama de constelación. Además, el menú permitirá ingresar una secuencia de datos a modular y controlar la ganancia del equipo USRP NI 2920.
3. Utilizando el modulador desarrollado en el Procedimiento, modificarlo con el fin de tener un modulador QPSK. Además, realizar el menú descrito en el apartado anterior para visualizar los resultados. Adjuntar los cambios realizados.
4. En el flujograma del modulador QPSK realizado en el apartado anterior, añadirle una fuente de ruido gaussiano con el fin de visualizar sus efectos en las señales y en el diagrama de constelación. Adjuntar y comentar los resultados.

2.1.7 PRÁCTICA 7: DEMODULACIÓN M-PSK

2.1.7.1 Objetivo

Comprender el proceso de Demodulación Digital de múltiples estados. Caso de estudio: Demodulación BPSK (*Binary Phase Shift Key*).

2.1.7.2 Marco Teórico

Modulación M-PSK

En la modulación por desplazamiento de fase (PSK), los símbolos de transmisión modulan la fase de una portadora. Si hay M símbolos diferentes, la portadora puede simplemente mantener una de las M fases para cada intervalo de símbolo.

Diagrama de constelación

El diagrama de constelación es muy similar al diagrama vectorial, y es una manera de representar los estados de los símbolos de la portadora en términos de amplitud y fase [14]. La distribución de los puntos de constelación se da en función del tipo de modulación que se esté utilizando, tal como se muestra en la figura 2.36.

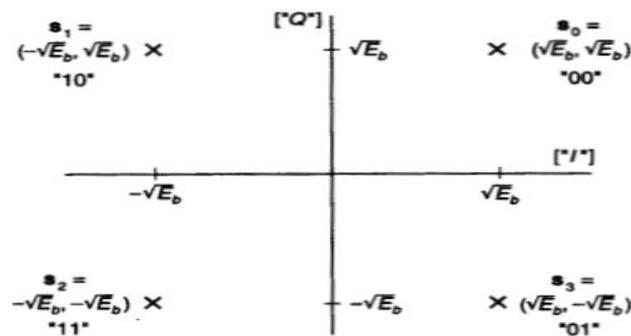


Figura 2.36. Diagrama de constelación QPSK [14]

Corrección de Fase

Muchas veces, cuando se analizan modulaciones digitales en base a constelaciones en una transmisión digital, suele aparecer un fenómeno en dicha constelación, en la que la constelación puede llegar girada (desfasada), como se ve en la figura 2.37, esto, debido a las anomalías introducidas en el canal de comunicaciones.

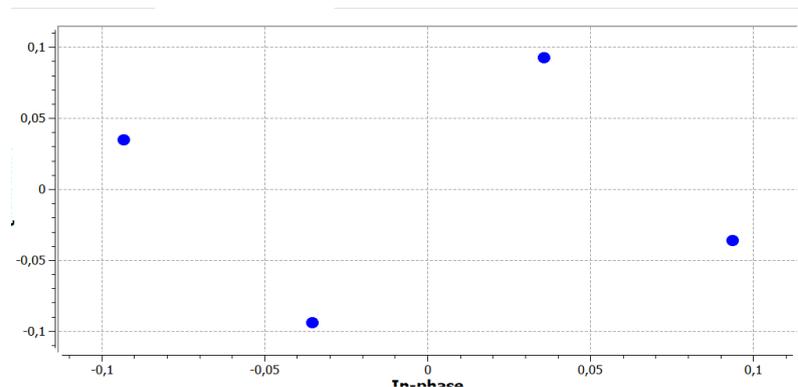


Figura 2.37. Constelación luego de atravesar el canal.

Una forma de solucionar este problema es mediante un corrector de fase, en este caso el lazo de Costas, el cual, es uno de los circuitos más utilizados en el caso de demoduladores digitales. Por esta razón, se pretende utilizar el circuito *Costas Loop*, bloque que pertenece a la plataforma de GNU Radio, el cual se mencionará en la sección 2.1.7.3.

Implementación en GNU Radio

A manera de ejemplo se presenta un demodulador BPSK, el cual complementa la práctica anterior, para ello se procede a implementar el flujograma indicado en la figura 2.38.

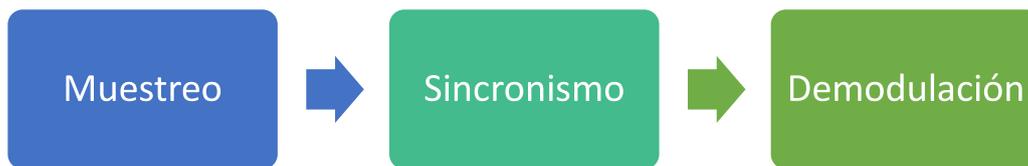


Figura 2.38. Diagrama de bloques del demodulador

Muestreo

Para el caso de muestreo se utiliza el bloque *diezmador_cc*, el cual toma cierta cantidad de muestras de la señal entrante; esto se realiza debido a que la señal generada por el USRP se muestrea a una frecuencia en el orden de los MHz. La forma de configurar el bloque se indica en la figura 2.39, donde *Sps* indica la cantidad de muestras a analizar.



Figura 2.39. Bloque *e_diezmador_cc*

Sincronismo

Al momento de que la señal atraviese el canal, ésta se ve expuesta a diferentes fenómenos que la podrían alterar, por ejemplo, un cambio en la fase o en la frecuencia, lo que ocasiona que la constelación transmitida gire, tal como se muestra en la figura 2.40.

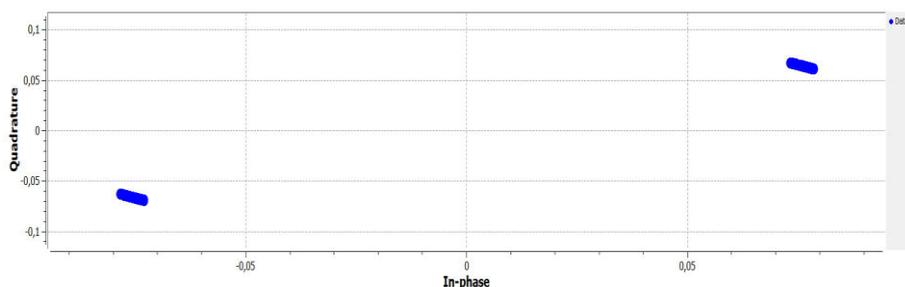


Figura 2.40. Constelación girada debido a un cambio en la fase

Para solucionar este inconveniente se utiliza el bloque *Costas Loop*, el cual se mencionó anteriormente; el resultado de este bloque se presenta en la figura 2.41.

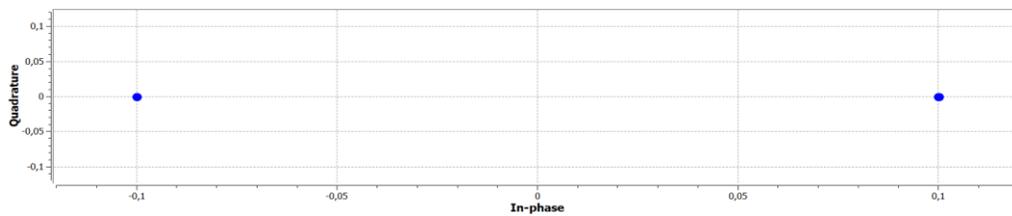


Figura 2.41. Constelación corregida por el bloque *Costas Loop*

Constellation Receiver

Bloque propio de la plataforma de GNU Radio, el cual realiza el proceso inverso al bloque *Chunks to Symbols*. Entre los diferentes parámetros a configurar, se tienen:

1. *Constellation Object*: indica el objeto de constelación utilizado, es decir, se coloca el nombre de la variable *constelacion* utilizada en el flujograma, como se indica a continuación:

constelacion.base()

2. *Loop Bandwidth*: indica el ancho de banda del bloque *Costas Loop*, se recomienda mantener un valor por defecto de:

$2 \cdot \pi / 100$

Demodulación

En esta etapa se realiza el proceso contrario al indicado en la práctica anterior, es decir se realiza la decodificación, mapeo y desempaquetamiento de bits. Para ello se utilizan los siguientes bloques:

- *Differential Decoder*: realiza el proceso contrario al bloque *Differential Encoder*, el resultado se muestra en las figuras 2.42 y 2.43.

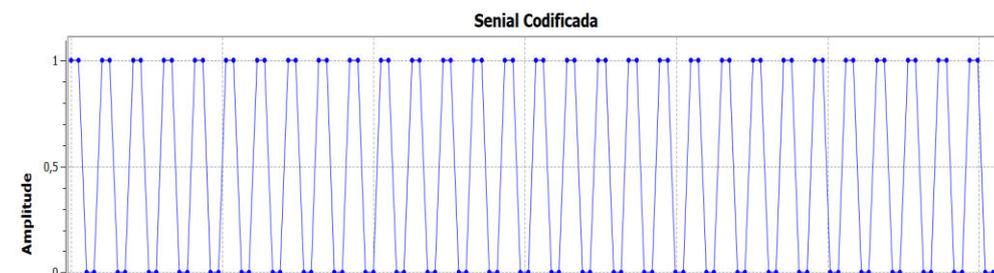


Figura 2.42. Recuperación de la señal codificada.

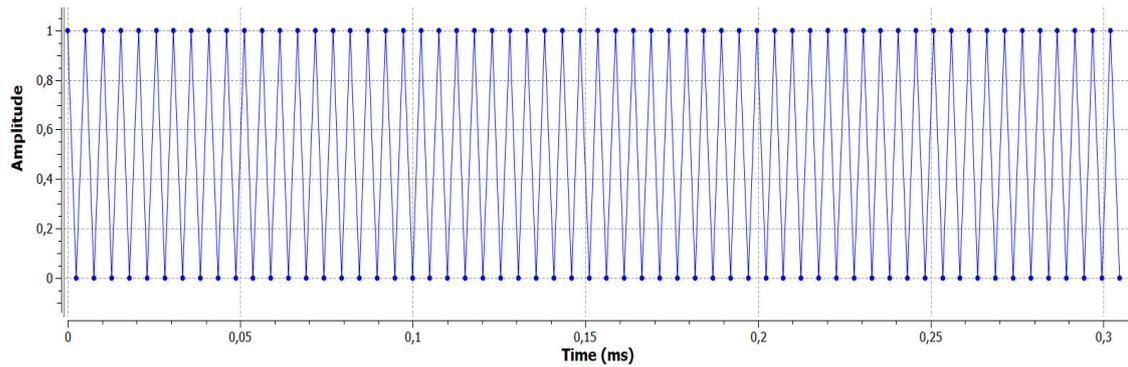


Figura 2.43. Señal decodificada

- *Map*: Realiza el proceso contrario al indicado en la práctica 6, es decir, mapea los símbolos de Código Gray a código binario. Como se mencionó anteriormente, el mapeo a Código Gray en el caso de BPSK no toma ningún efecto, ya que la representación de 0 y 1 en Código Gray es la misma, como se muestra en la figura 2.44.

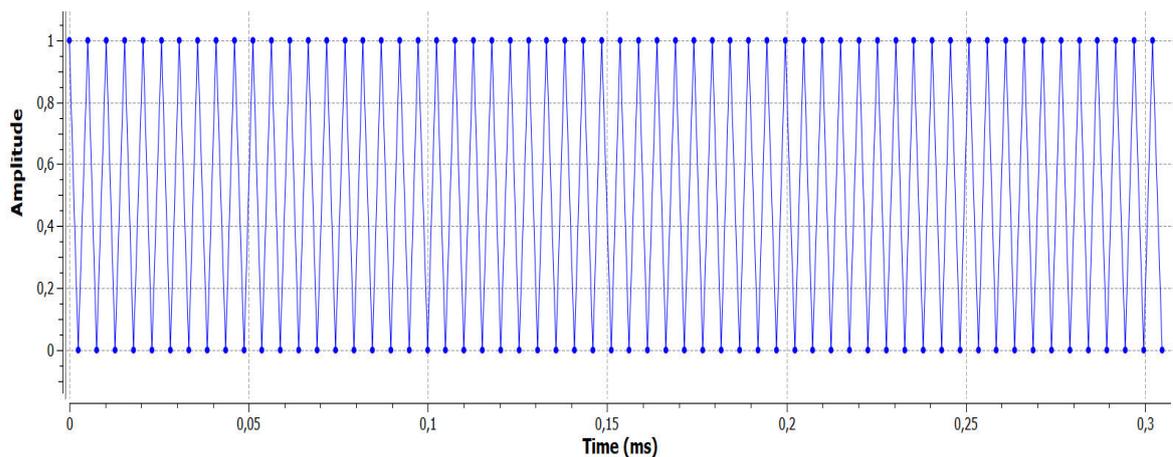


Figura 2.44. Señal mapeada de Código Gray a binario

2.1.7.3 Trabajo Preparatorio

1. Preparar el modulador PSK desarrollado en la práctica anterior, así como también el archivo de texto utilizado.
2. Consultar el modo de uso de los siguientes bloques:
 - a. *FLL Band-Edge*
 - b. *Costas Loop*
 - c. *Unpacked to Packed*
 - d. *Wav File Sink*

e. File Sink

2.1.7.4 Procedimiento

1. Implementar un demodulador B-PSK en base al modelo indicado en la figura 2.45.



Figura 2.45. Modelo del receptor

- a. Muestreo: utilizar el bloque *e_diezmador_cc*, el cual será puesto a disposición de los estudiantes el día de la práctica.
 - b. Sincronismo: como se mencionó en el trabajo preparatorio, para esta etapa se utiliza el bloque *Costas Loop*, el cual corrige la desviación de fase del diagrama de constelación, ocasionado por los efectos negativos del canal.
 - c. Demodulación: para esta etapa se utilizará el bloque *Constellation Receiver*, el cual realiza un proceso opuesto al bloque *Chunks to Symbols*.
2. Una vez realizada la conexión de los bloques mencionados en el numeral anterior, configurar el número de muestras y la variable de constelación, la cual, es la misma que el modulador PSK. Además, realizar la decodificación de la señal demodulada, previo a mostrar los resultados; es decir, realizar el proceso inverso al mostrado en la etapa previa a la modulación en la práctica anterior.
 3. Realizar la transmisión del archivo de texto mediante el modulador desarrollado en la práctica 6. Posteriormente, realizar la demodulación del archivo y guardarlo en un archivo txt. Capturar gráficas de los resultados de cada etapa, tanto en transmisión, como en recepción. Variar distintos parámetros hasta obtener un resultado favorable y anotarlos. Entre éstos, se encuentran:
 - a. *Delay* en el receptor: bloque necesario con el fin de tener una mejor sincronización en tiempo.
 - b. Ganancia del transmisor y receptor.
 - c. Amplitud de las señales producidas por el USRP.

2.1.7.5 Informe

1. Analizar los resultados obtenidos utilizando un archivo de audio como fuente de información. Además, añadir ruido gaussiano y observar los resultados.

2. A partir del flujograma desarrollado en el Procedimiento, crear un menú interactivo con el usuario, el cual debe mostrar las siguientes señales: recibida, señal filtrada, señal demodulada, diagrama de constelación antes y después del sincronismo. Además, el menú permitirá controlar la ganancia del equipo USRP NI 2920.
3. Utilizando el demodulador desarrollado en el Procedimiento, modificarlo con el fin de tener un demodulador QPSK. Posteriormente, crear un bloque del modulador y otro bloque del demodulador QPSK, esto con el fin de tener una mejor visualización del flujograma. Adjuntar los resultados.
4. A manera de simulación, desarrollar un sistema de transmisión-recepción de datos mediante QPSK, utilizando como fuente de información un archivo de audio. Además, desarrollar el menú de visualización mencionado en el numeral 2 anterior.

2.1.8 PRÁCTICA 8: ANÁLISIS DE CANALES CON ISI (*INTERSYMBOL INTERFERENCE*)

2.1.8.1 Objetivos

Analizar diferentes conceptos relacionados a la transmisión digital de señales, tales como factor de *roll-off*, *jitter*, ISI y diagrama del ojo.

2.1.8.2 Marco Teórico

ISI (*INTERSYMBOL INTERFERENCE*)

Es una anomalía de la señal, mediante la cual, algunas componentes de un pulso específico interfieren sobre los pulsos adyacentes a éste (figura 2.46). Este fenómeno es producido debido a varios factores.

- Ancho de banda finito del canal.
- Superposición de pulsos.
- Efectos negativos del canal: ruido, desvanecimiento, entre otros.

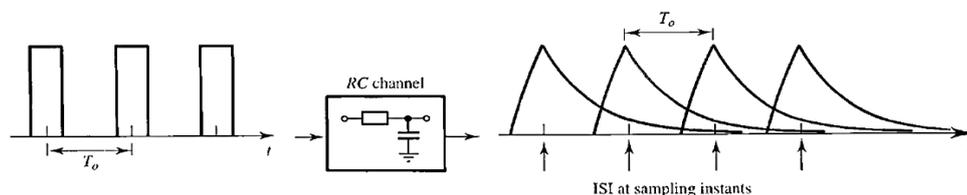


Figura 2.46. Superposición de pulsos ocasionado por ISI [13]

Filtro coseno levantado

Es un modelo de filtro electrónico, utilizado en sistemas de transmisión digital, debido a que es capaz de reducir al mínimo anomalías de la señal, tales como la ISI (interferencia entre símbolos). Es denominado coseno alzado porque la representación de su espectro es una señal coseno.

- Factor de *roll-off*. El factor de *roll-off* muestra el porcentaje de ancho de banda que excede la señal del filtro respecto ancho de banda que ocuparía el pulso rectangular. Es decir, a un mayor factor de *roll-off*, se tendría una mejor representación de los símbolos transmitidos, pero también se necesitaría más ancho de banda para implementar un filtro con un factor de *roll-off* alto, como se muestra en la figura 2.47.

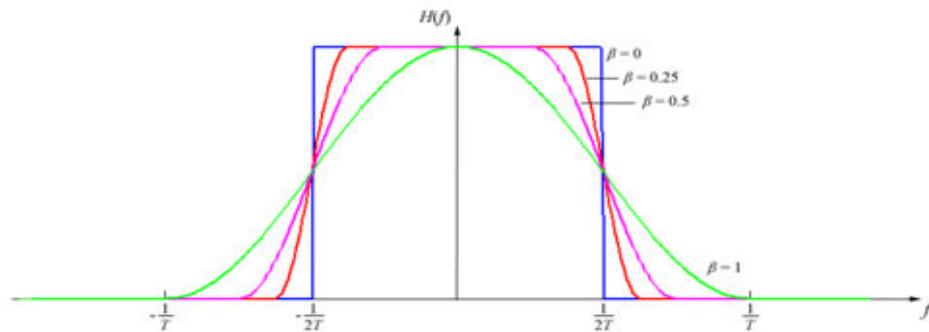


Figura 2.47. Influencia del factor de *Roll-Off* en el filtro RC

DIAGRAMA DEL OJO

Es una técnica de medición experimental, desarrollada con el fin de evaluar ciertos parámetros que definen la calidad de un sistema de transmisión. Entre los parámetros que se puede analizar en este diagrama se tienen:

- **ISI**
 - o Generalmente se da por un mayor número de muestras en el filtro RC (*Raised Cosine*).
 - o Existe ISI cuando el ojo comienza a cerrarse
 - o Como se puede apreciar en la figura 2.48 cuando se tiene un *roll-off* de 1 pueden existir problemas de ISI. Sin embargo, el problema no es tan serio como cuando se tiene un *roll-off* de 0, esto porque las colas son más pequeñas que cuando se tiene un *roll-off* de 0. Sin embargo, el ancho de banda necesario aumenta para un *roll-off* mayor [15].
 - o Se da una distorsión en el cruce por cero.

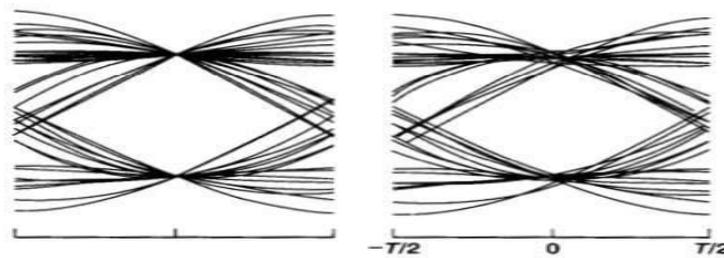


Figura 2.48. Señal con ISI debido a $roll\ off = 0.3$ [14]

- JITTER

- Se puede apreciar mediante las fluctuaciones por el cruce por cero
- Esto depende mucho del factor de $roll-off$ del filtro
 - A un mayor factor de $roll-off$ se tiene menos presencia de $jitter$ (figura 2.49)

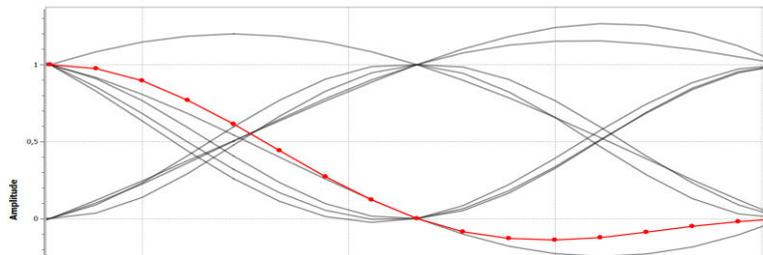


Figura 2.49. Señal con jitter ($roll\ off = 0.3$)

2.1.8.3 Trabajo Preparatorio

1. En un sistema de comunicación digital señalar y justificar en qué etapa es conveniente la utilización del diagrama de ojo.
2. Consultar cómo analizar anomalías como jitter e ISI en el diagrama de ojo.
3. Consultar el modo de uso del bloque *Random Source*, así como su configuración.
4. Configurar el modulador MPSK de la práctica 6, con el fin de visualizar los diagramas de ojo de diferentes estados.

2.1.8.4 Procedimiento

1. Para la presente práctica se usará como fuente de información una fuente binaria, generada mediante el bloque *Random Source*.
2. Una vez digitalizada la señal, se filtra utilizando un filtro coseno alzado, para ello, se utilizará el bloque *b_Raised_cosine_ff*, el cual será puesto a disposición de los estudiantes el día de la práctica.

3. Crear un analizador de diagramas de ojo a partir de bloques jerárquicos, para ello se hará uso de los bloques: *delay*, *QT GUI Sink*.
4. Presentar los diagramas de ojo con un factor de *roll off* de 0.75 correspondientes a modulaciones digitales BPSK y QPSK, para esto, se utilizará el modulador PSK desarrollado en la práctica 6.

2.1.8.5 Informe

1. Analizar los resultados obtenidos durante la práctica, variando el factor de *roll off* y el tipo de modulación de la señal.
2. A partir del flujograma desarrollado en el Procedimiento, crear un menú interactivo con el usuario, el cual debe permitir visualizar la señal original, la señal digitalizada, la señal filtrada y el diagrama de ojo correspondiente. Además, implementar *sliders* que permitan variar el número de muestras del filtro y el factor de *roll off*.
3. En el flujograma desarrollado en el Procedimiento, implementar un canal AWGN con el objetivo de analizar los efectos de éste en el diagrama de ojo. Además, implementar una *Slider* que permita variar el nivel de ruido en el canal. Adjuntar los resultados.
4. Crear una entrada de datos en el flujograma, la cual permitirá al usuario ingresar una secuencia de bits para ser digitalizada, filtrada y analizada mediante el diagrama de ojo.

2.1.9 PRÁCTICA 9: DSSS (DIRECT SEQUENCE SPREAD SPECTRUM)

2.1.9.1 Objetivo

Comprender el sistema DSSS mediante la utilización de un sistema de modulación BPSK.

2.1.9.2 Marco Teórico

SPREAD SPECTRUM

Espectro expandido es una técnica utilizada para transmitir datos digitales a través de un ancho de banda más amplio (ensanchado/expandido), con el fin de que su interceptación sea mucho más complicada.

Existen dos formas de implementar esta técnica de espectro expandido:

- FHSS (*Frequency Hopping Spread Spectrum*)

- Técnica mediante la cual, la señal se transmite mediante una serie pseudoaleatoria de frecuencias, saltando de frecuencia en frecuencia en intervalos establecidos.
- DSSS (*Direct Sequence Spread Spectrum*)
 - Técnica mediante la cual, cada bit de la señal es representado por una combinación de bits, usando un código de expansión.

DSSS (*Direct Sequence Spread Spectrum*)

En esta técnica, se considera que cada bit de la señal original a transmitir es representado por un conjunto de bits de una secuencia de expansión. Estos bits, son denominados chips, ya que son más pequeños que los bits de la señal a transmitir.

Para combinar los bits de la señal a transmitir con los chips de la secuencia de expansión, se emplea la operación lógica XOR, como se muestra en la figura 2.50.

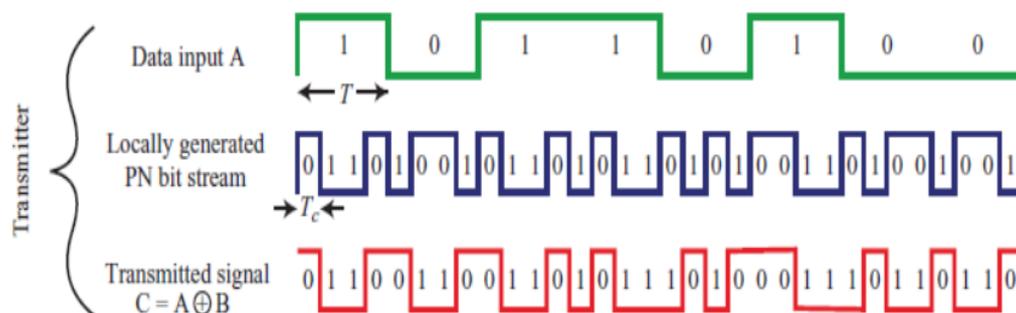


Figura 2.50. Operación XOR entre datos y señal PN

2.1.9.3 Trabajo Preparatorio

1. Consultar el diagrama de bloques de un sistema DSSS utilizando una modulación BPSK.
2. Consultar el modo de uso del bloque GLFSR.
3. Configurar el modulador y demodulador BPSK desarrollado en prácticas anteriores.
4. Preparar un archivo de texto para su transmisión y recepción.

2.1.9.4 Procedimiento

1. Cargar el archivo de texto mencionado en el preparatorio mediante el bloque *File Source* y digitalizarla mediante el bloque *Packed to Unpacked*.

2. Obtener la secuencia DSSS mediante la operación XOR entre la señal digitalizada y la secuencia de PseudoRuido generada por el bloque GLFSR.
3. Con la señal DSSS, utilizar el modulador BPSK desarrollado en prácticas anteriores para modular esta señal con BPSK. Posteriormente, transmitirla con los bloques *UHD: USRP Sink* y recibirla con el bloque *UHD: USRP Source*.
4. Utilizar el demodulador BPSK para recibir la secuencia transmitida y decodificarla empleando el bloque GLFSR mencionado en el numeral 2 de este Procedimiento.
5. En el receptor, tener en cuenta los diferentes parámetros a configurar, con el fin de recibir de manera correcta el archivo transmitido, tal como se realizó en la práctica 7.

2.1.9.5 Informe

1. Analizar los resultados obtenidos durante la práctica utilizando dos secuencias diferentes de datos para transmitir en el ambiente de GNU Radio.
2. A partir del flujograma desarrollado, crear un menú interactivo con el usuario, el cual debe permitir visualizar la señal original, la secuencia DSSS generada, la señal modulada. Además, realizar el mismo menú para el receptor.
3. A manera de simulación, implementar un sistema DSSS en un ambiente con ruido, para esto se utilizará el bloque *Channel Model*. Además, tanto el modulador como el demodulador BPSK deben ser representados como bloques creados. Adjuntar los resultados.

3. RESULTADOS Y DISCUSIÓN

En este apartado se presentan los resultados correspondientes a cada una de las 9 prácticas, tanto en la parte de transmisión como de recepción, así como también los resultados de los ejercicios de los informes de cada práctica. Para ello, se utilizan los dispositivos USRP NI 2920 para montar un sistema de transmisión y recepción de datos, esto en caso de que la práctica lo requiera. Por otra parte, se realizarán simulaciones en la plataforma de GNU Radio para validar los resultados de lo requerido en el informe de cada una de las prácticas de laboratorio.

3.1 RESULTADOS DE LAS PRÁCTICAS

En este apartado se presentan las gráficas de los resultados de cada práctica, éstas con su respectivo análisis.

3.1.1 PRÁCTICA 1: INTRODUCCIÓN A GNU RADIO Y USRP NI 2920

3.1.1.1 Creación del bloque *Calculadora*

Para la creación de bloques se siguen los mismos pasos indicados en el marco teórico, para ello se requieren de los bloques *Pad Source* y *Pad Sink*, así como los bloques que realicen las diferentes operaciones.

Para realizar las operaciones indicadas en numeral 1 del procedimiento se utilizan los bloques *Multiply*, *Add* y *Divide*, como se muestra en la figura 3.1.

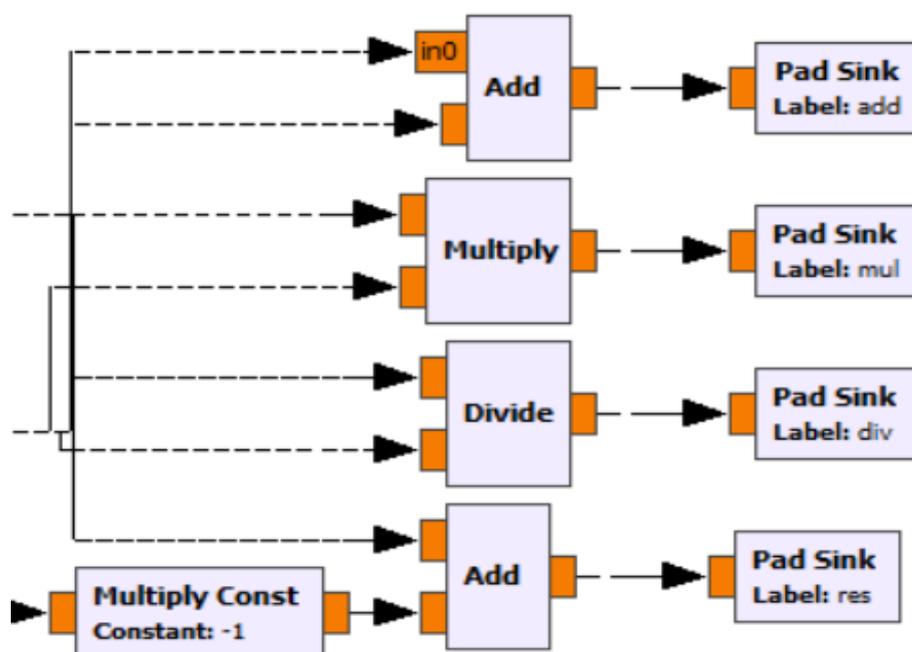


Figura 3.1. Bloques que simulan la calculadora

Posteriormente, se configuran los valores a ingresar en la calculadora mediante el bloque *Parameter*, tal como se muestra en la figura 3.2.

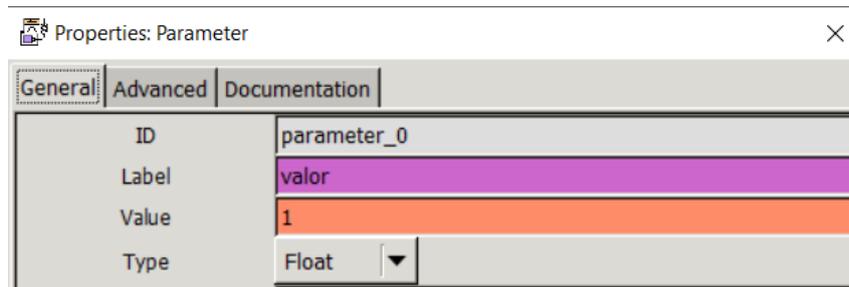


Figura 3.2. Configuración del bloque *Parameter*

Finalmente, se configuran las entradas y salidas del bloque mediante los bloques *Pad Source* y *Pad Sink*, como se muestra en la figura 3.3.

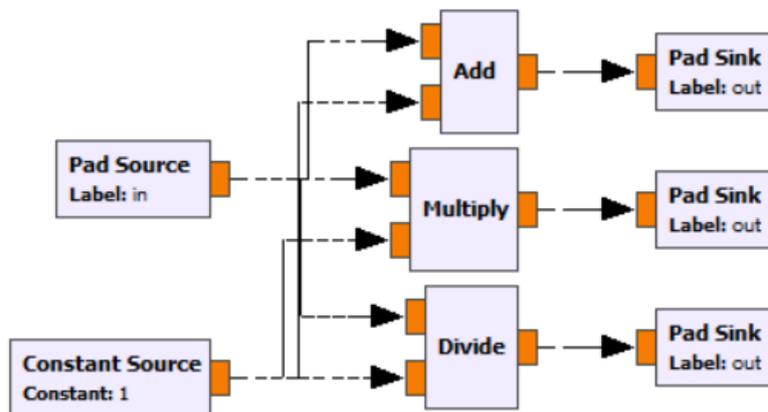


Figura 3.3. Uso de los bloques *Pad Source* y *Pad Sink*

3.1.1.2 Creación del menú de visualización de resultados

Para la creación del menú de usuario se utilizarán los bloques *QT GUI Tab Widget*, tal como se muestra en la figura 3.4.



Figura 3.4. Menú de usuario

3.1.2 PRÁCTICA 2: ANÁLISIS DE SEÑALES

3.1.2.1 Generación de onda sinusoidal

En este literal, se genera una señal definida por el instructor, en este caso, una señal seno de frecuencia 1 KHz y de amplitud 1. Para ello, se configura el bloque *Signal Source*, tal como se muestra en la figura 3.5.

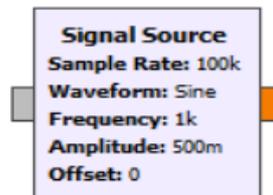


Figura 3.5. Bloque *Signal Source*

Posteriormente, el resultado se muestra en la figura 3.6.

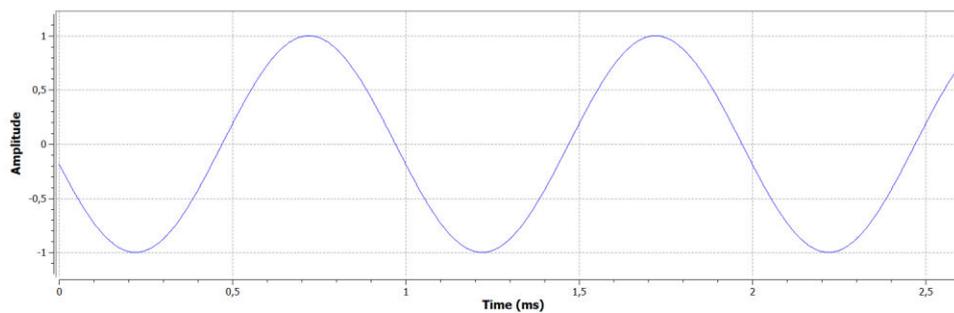


Figura 3.6. Señal en el dominio del tiempo

Como se puede apreciar, el período de la señal es 1 ms, correspondiente a 1 KHz. Además, la representación de la señal en el dominio de la frecuencia se lo puede visualizar mediante la FFT (Transformada Rápida de Fourier), esto mediante el bloque QT *Frequency Sink* en la figura 3.7.

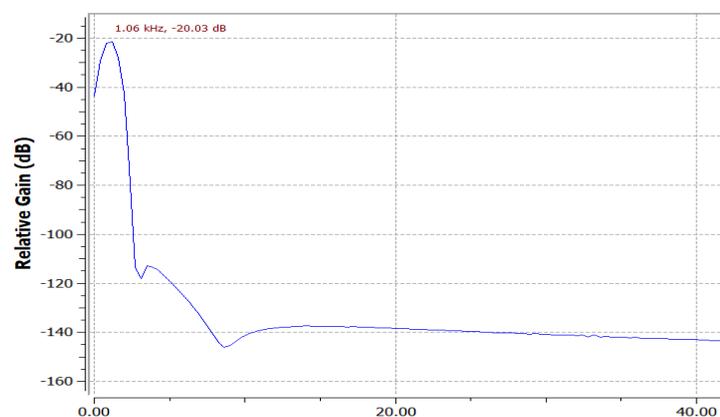


Figura 3.7. Señal en el dominio de la frecuencia

3.1.2.2 Generación de onda cuadrada

En este punto, se realiza el mismo proceso que en el numeral anterior, pero cambiando el tipo de onda, la amplitud y frecuencia. A manera de ejemplo, se usa la misma frecuencia y amplitud que la señal generada en el literal anterior, cuyo resultado se aprecia en las figuras 3.8 y 3.9.

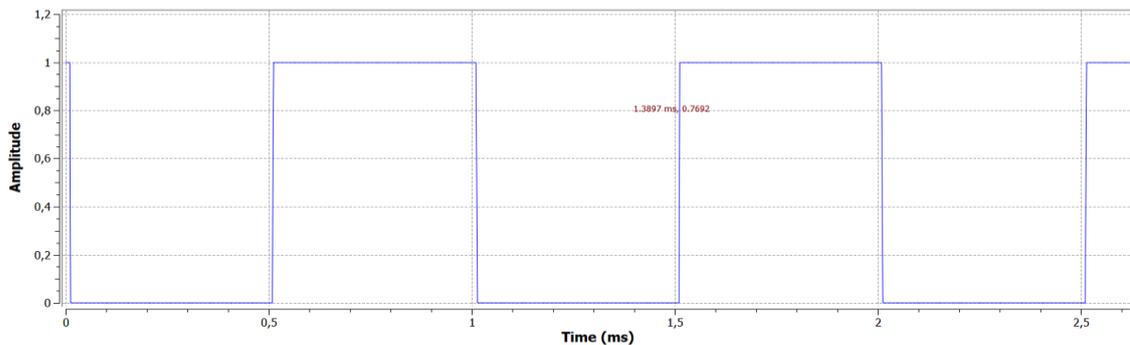


Figura 3.8. Señal 2 en el dominio del tiempo

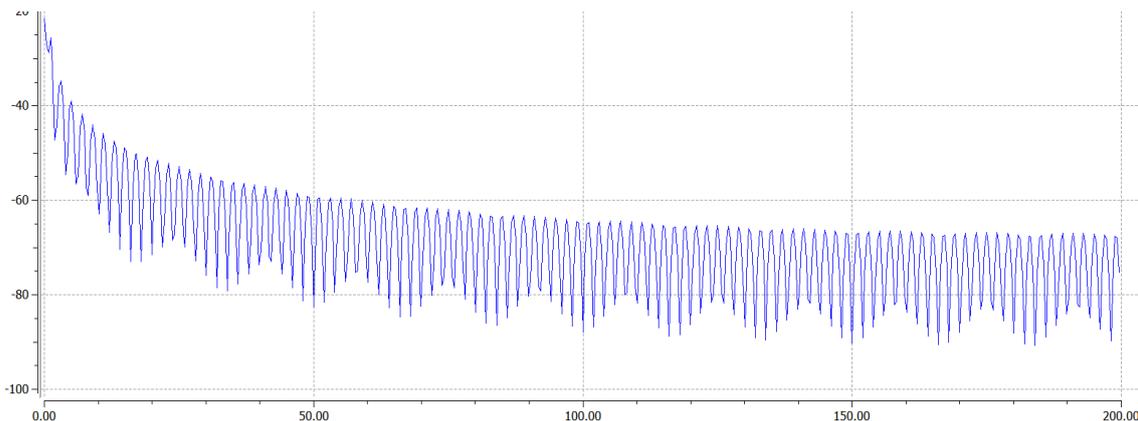


Figura 3.9. Señal 2 en el dominio de la frecuencia

3.1.2.3 Remuestreo de las señales generadas en los numerales 1 y 2

En este apartado se procede con el cambio de frecuencia de muestreo mediante el bloque *Rational Resampler* (figura 3.10), para ello se utiliza una frecuencia de muestreo que el USRP soporte. A manera de ejemplo, se utiliza 390.625 KHz [16].



Figura 3.10. Cambio de frecuencia de muestreo de la señal

3.1.2.4 Transmisión y recepción de señales mediante los bloques *WBFM Transmit* y *WBFM Receive*

En este numeral, se procede con la transmisión y recepción de las señales mediante los bloques *WBFM Transmit* y *WBFM Receive*. En las figuras 3.11 y 3.12 se muestra la señal a transmitir, tanto en el dominio del tiempo y de la frecuencia.

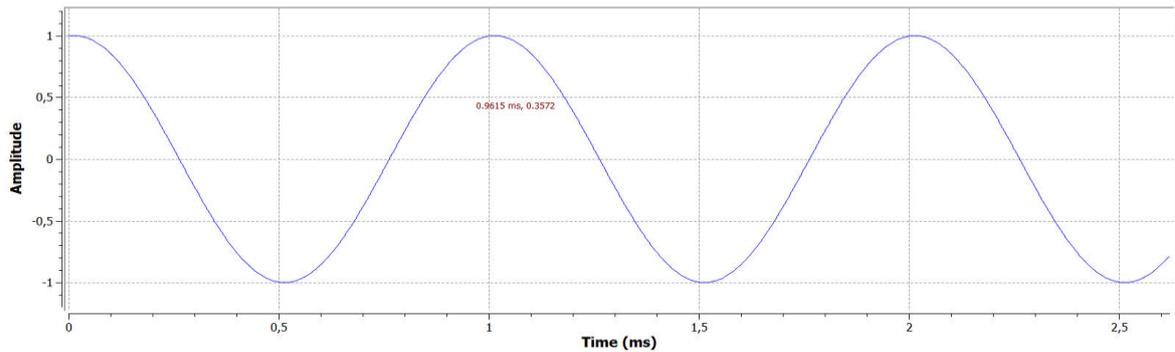


Figura 3.11. Señal transmitida

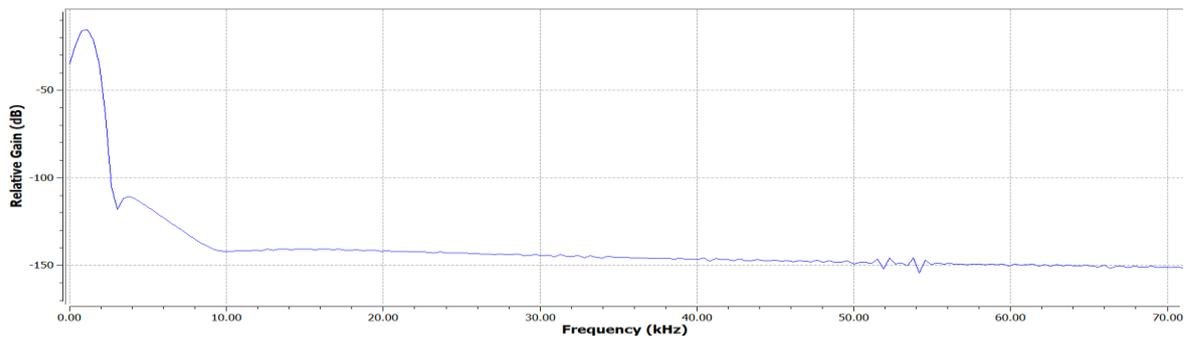


Figura 3.12. Señal transmitida en el dominio de la frecuencia

Una vez transmitida la señal se procede a verificar la correcta recuperación de la señal en el receptor, como se muestra en las figuras 3.13 y 3.14.

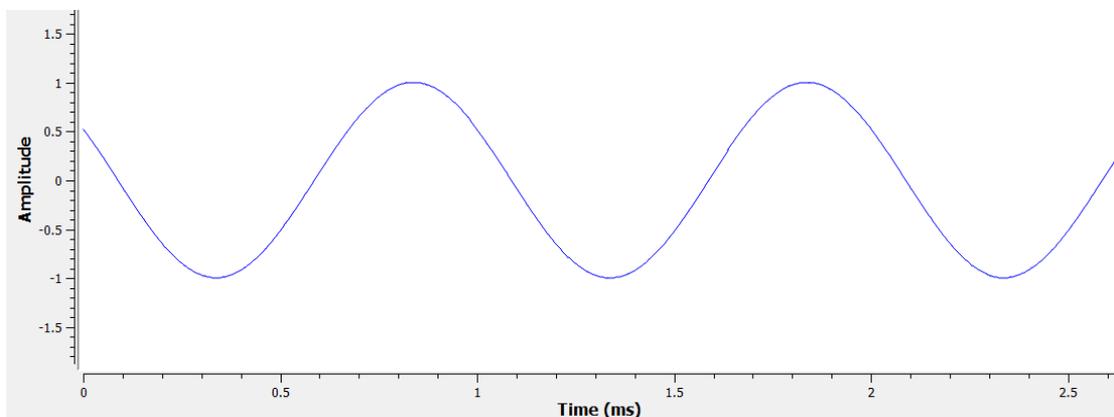


Figura 3.13. Señal recibida en el dominio del tiempo

Como se puede apreciar en la figura 3.14, la señal recibida posee una frecuencia de 1 KHz, tal como la señal original.

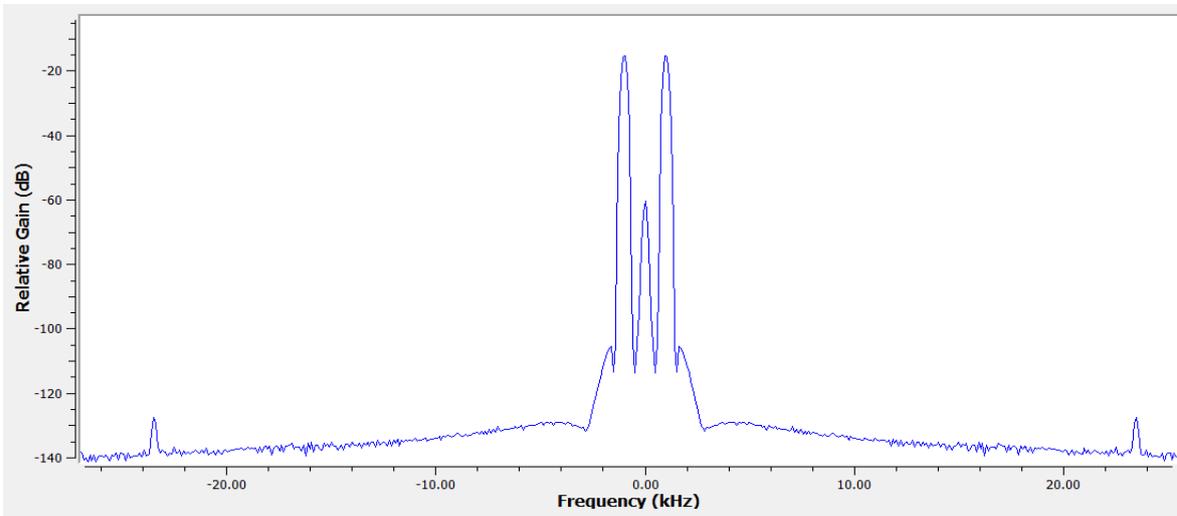


Figura 3.14. Señal recibida en el dominio de la frecuencia

A continuación, se tiene la señal cuadrada generada en el numeral 2, tal como se muestra en las figuras 3.15 y 3.16.

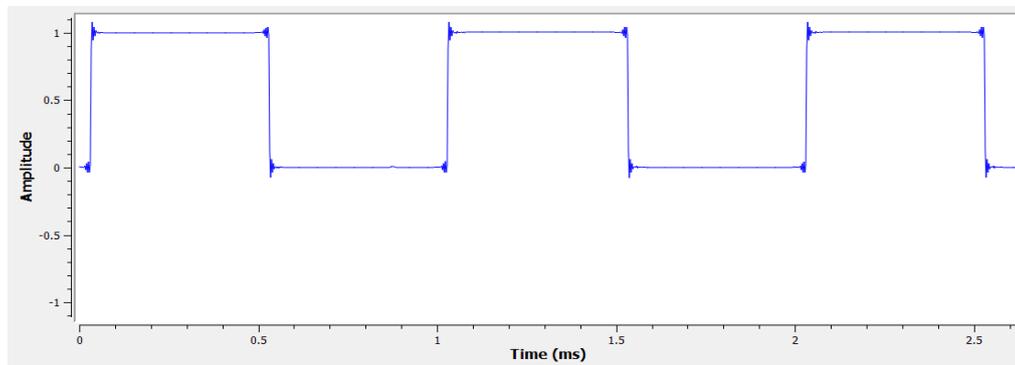


Figura 3.15. Señal cuadrada recibida

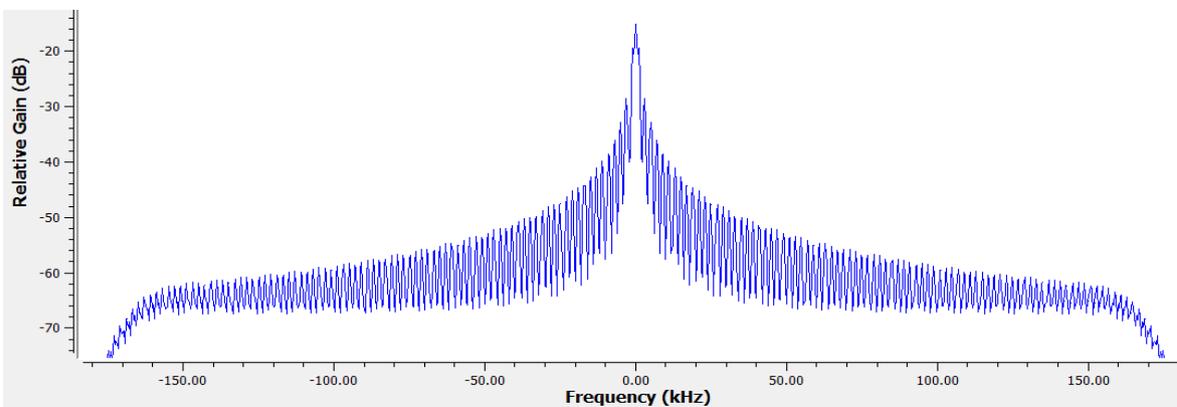


Figura 3.16. Espectro de frecuencia de onda cuadrada

3.1.2.5 Transmisión y recepción del producto de dos señales mediante los bloques *UHD: USRP Sink* y *UHD: USRP Source*

En este caso, se transmite el resultado de la multiplicación de una señal sinusoidal por una señal cuadrada, como se muestra en las figuras 3.17 y 3.18.

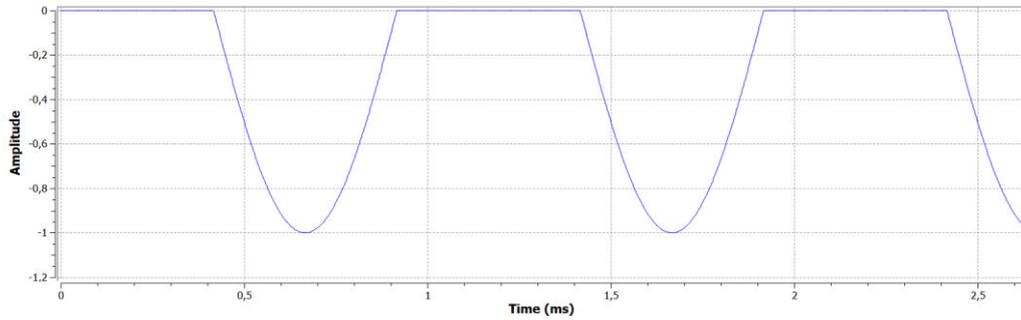


Figura 3.17. Señal transmitida en el dominio del tiempo

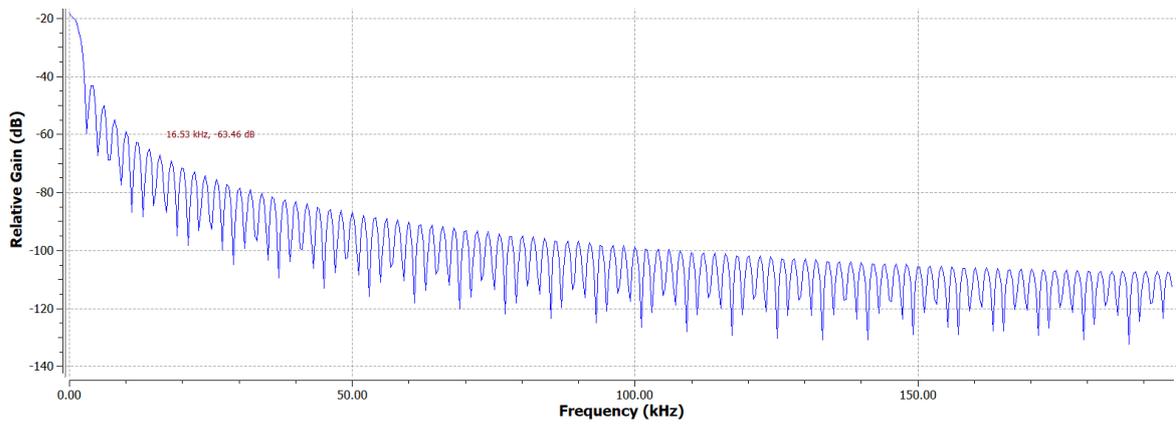


Figura 3.18. Señal transmitida en el dominio de la frecuencia

En las figuras 3.19 y 3.20 se presentan los resultados que se tienen en el receptor.

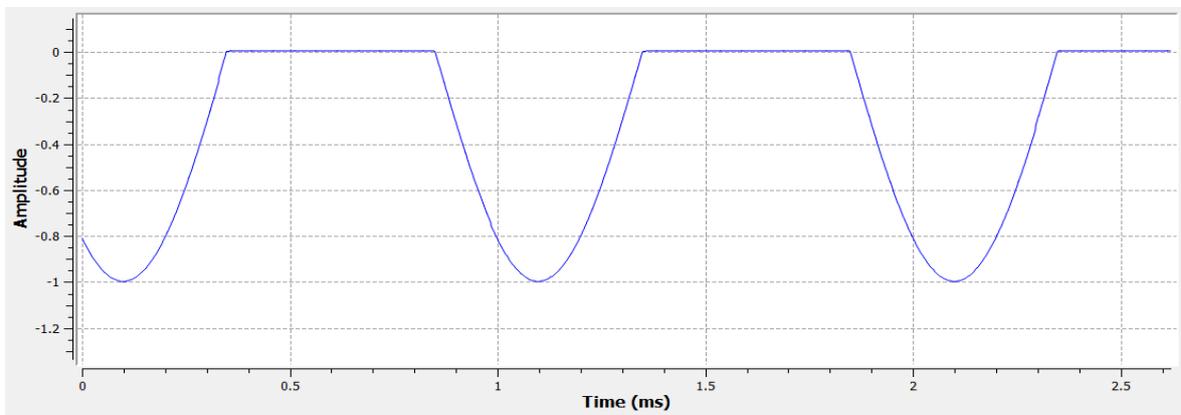


Figura 3.19. Señal recibida en el dominio del tiempo

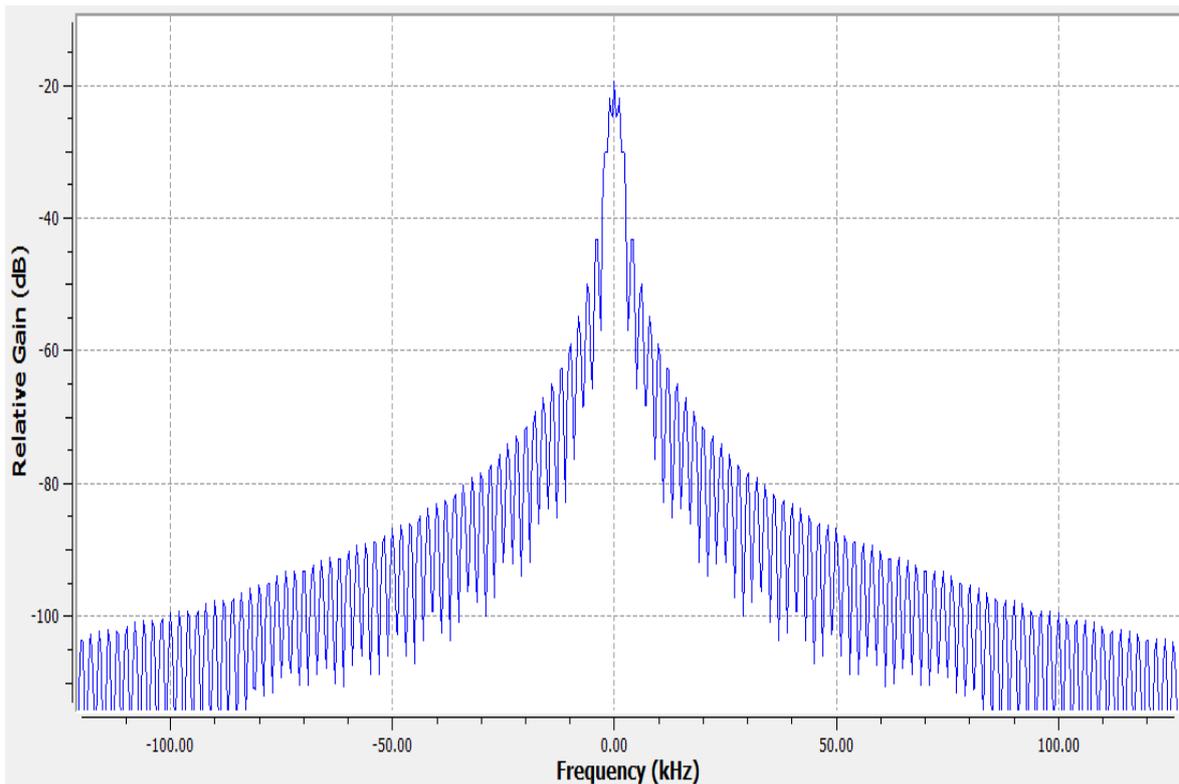


Figura 3.20. Señal recibida en el dominio de la frecuencia

Como el transmisor no está a una distancia considerable del receptor, la señal no es afectada, en gran medida, por los efectos del canal.

3.1.3 PRÁCTICA 3: ANÁLISIS DEL CANAL DE COMUNICACIONES

En esta práctica se pretende analizar el comportamiento de las señales ante las diferentes anomalías que introduce el canal.

3.1.3.1 Generación de onda sinusoidal

A manera de ejemplo, se utiliza el bloque *Signal Source* para generar la onda sinusoidal de frecuencia 1 KHz y amplitud 1, tal como se ve en las figuras 3.21 y 3.22.

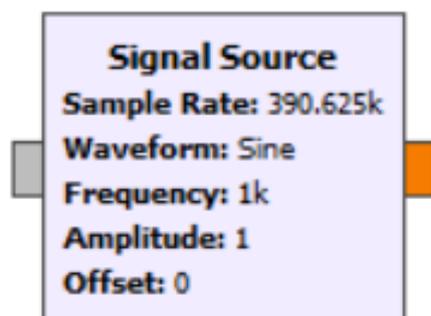


Figura 3.21. Configuración de *Signal Source* para señal sinusoidal

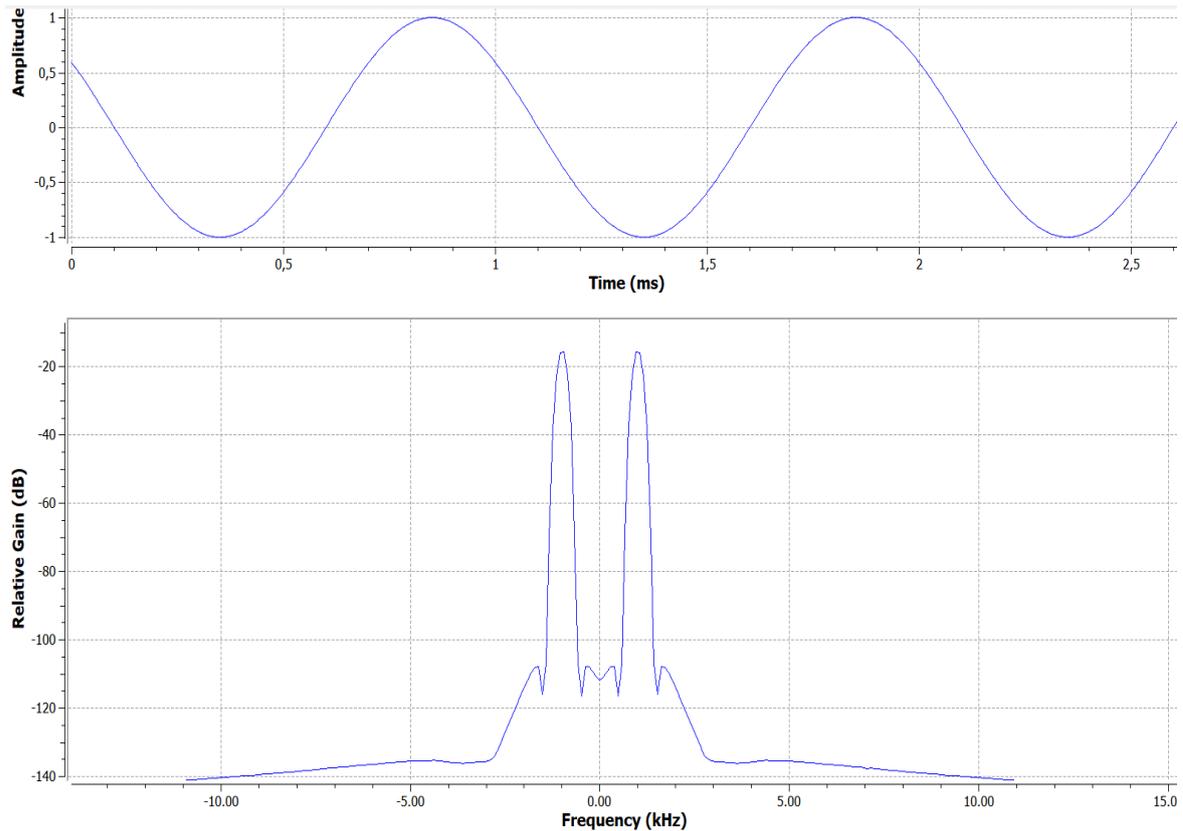


Figura 3.22. Señal generada mediante el bloque *Signal Source* en el dominio del tiempo y en el dominio de la frecuencia

3.1.3.2 Remuestreo de la señal a una frecuencia compatible con el USRP

A continuación, se cambia la frecuencia de muestreo de la señal, para ello en el bloque *Rational Resampler* se coloca en *Decimation* la frecuencia de la señal original y en *Interpolation* la frecuencia deseada, en este caso 390.625 KHz (figura 3.23), la cual es una frecuencia de muestreo aceptada por el USRP.

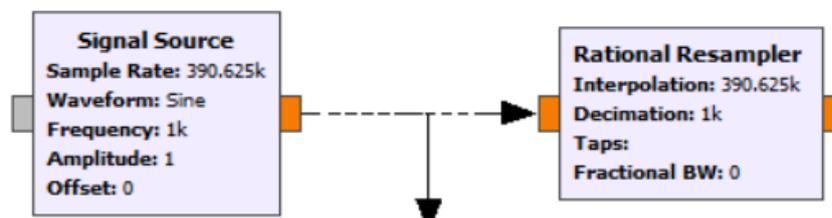


Figura 3.23. Cambio de frecuencia de muestreo de la señal

3.1.3.3 Transmisión y recepción de señales con ruido mediante los bloques *WBFM Transmit* y *WBFM Receive*

A continuación, se procede con la modulación de la señal, ésta es necesaria para transmitirla correctamente. En este caso, se utiliza modulación FM, ya que es una forma

de modulación que presenta una mejor calidad que AM, aunque de menor alcance. Sin embargo, al trabajar en un ambiente de laboratorio, el alcance no es un parámetro crítico.

El bloque utilizado en este caso es *WBFM Transmit* (figura 3.24) y *WBFM Receive*, en el transmisor y receptor, respectivamente.

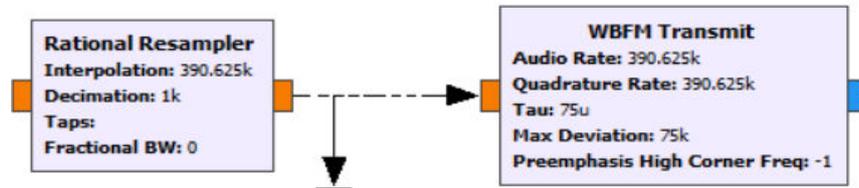


Figura 3.24. Modulación FM a la señal previa a transmitir

Con el fin de simular un canal con ruido, se añade una fuente de ruido mediante el bloque *Noise Source* y el bloque *Add*. Posteriormente, en el receptor se realiza el proceso contrario, es decir, se utiliza el bloque *WBFM Receive* y el bloque *Rational Resampler* para regresar a la frecuencia de muestreo original de la señal.

En las figuras 3.25 y 3.26 se presentan los resultados de la señal recibida, después de añadirle 0.5 V de ruido.

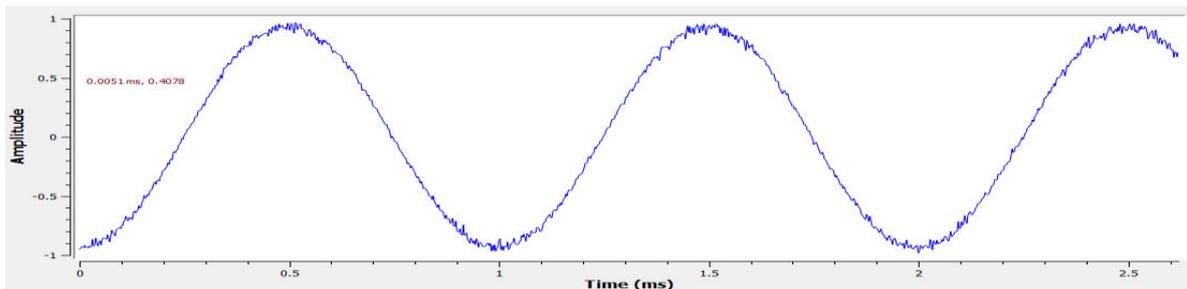


Figura 3.25. Señal recibida con ruido $n=0.5$

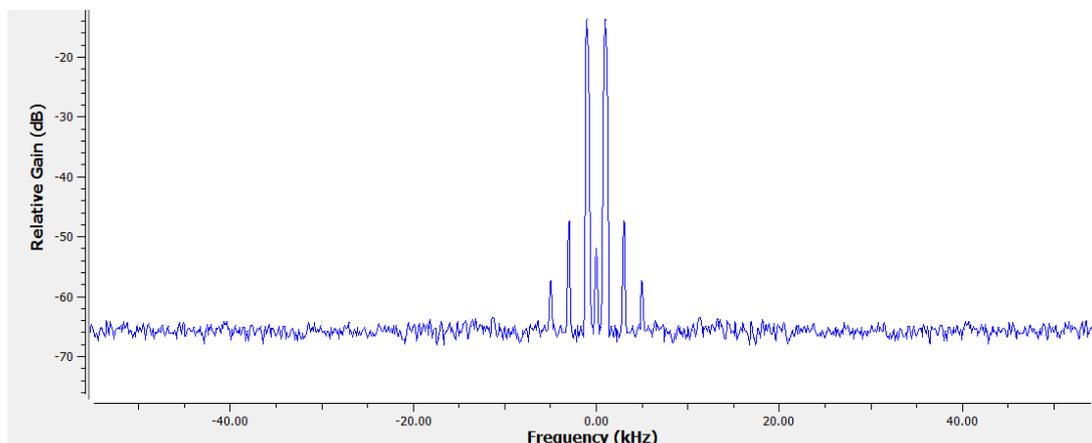


Figura 3.26. Señal recibida con ruido $n=0.5$ (Dominio de la frecuencia)

3.1.3.4 Transmisión y recepción de señales de audio en canales con ruido

En este numeral, se realiza el mismo procedimiento que en los numerales anteriores, pero teniendo como fuente de información a un archivo de audio .wav (figura 3.27).

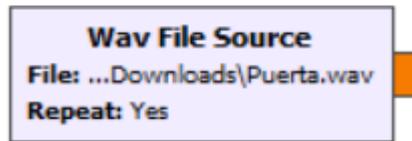


Figura 3.27. Archivo .wav cargado mediante el bloque *Wav File Source*

En este caso el cambio de frecuencia de muestreo depende de la frecuencia de muestreo a la cual ha sido grabada la canción. A manera de ejemplo se utiliza una canción de 44.1 KHz. Posteriormente, se realiza el mismo proceso que en los numerales anteriores, teniendo como resultados lo mostrado en las figuras 3.28 y 3.29.

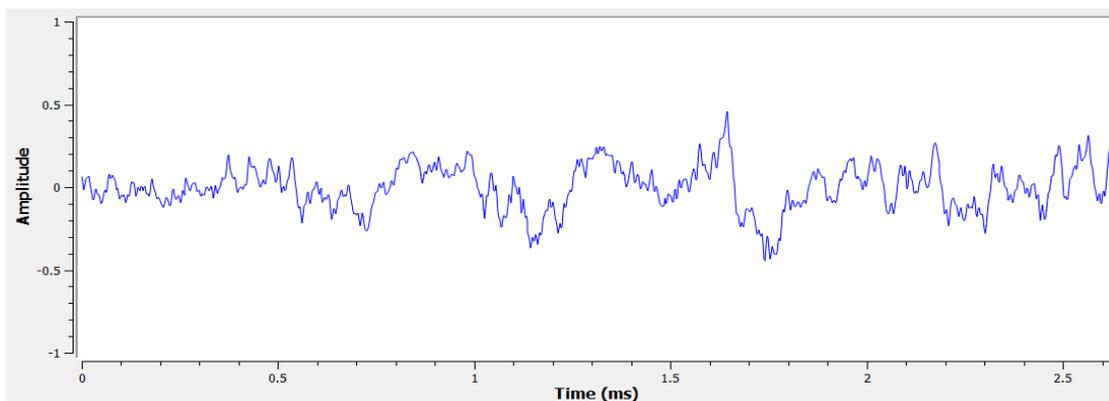


Figura 3.28. Archivo de audio a transmitir

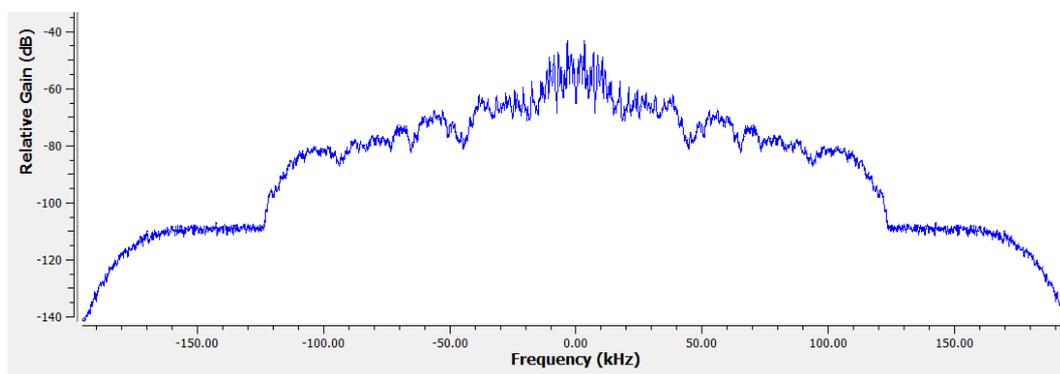


Figura 3.29. Espectro de frecuencia de la canción a transmitir

Finalmente, se añade un cierto nivel de ruido, esto, con el fin de analizar el comportamiento de la señal a través del canal, teniendo como resultado el mostrado en las figuras 3.30 y 3.31.

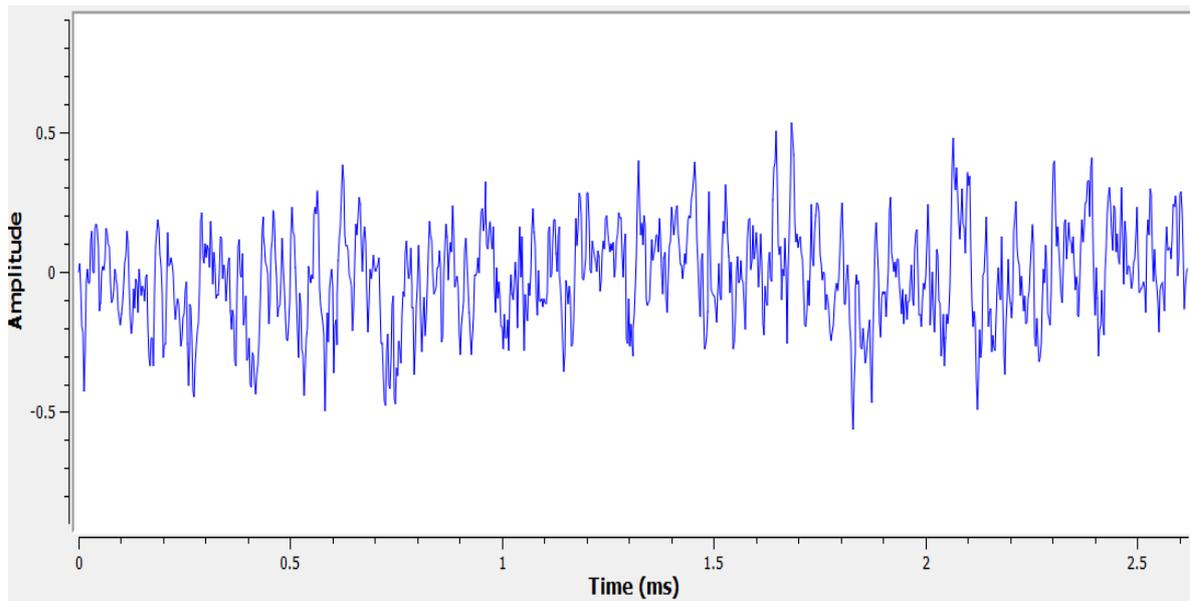


Figura 3.30. Señal recibida con un nivel de ruido $n=3$

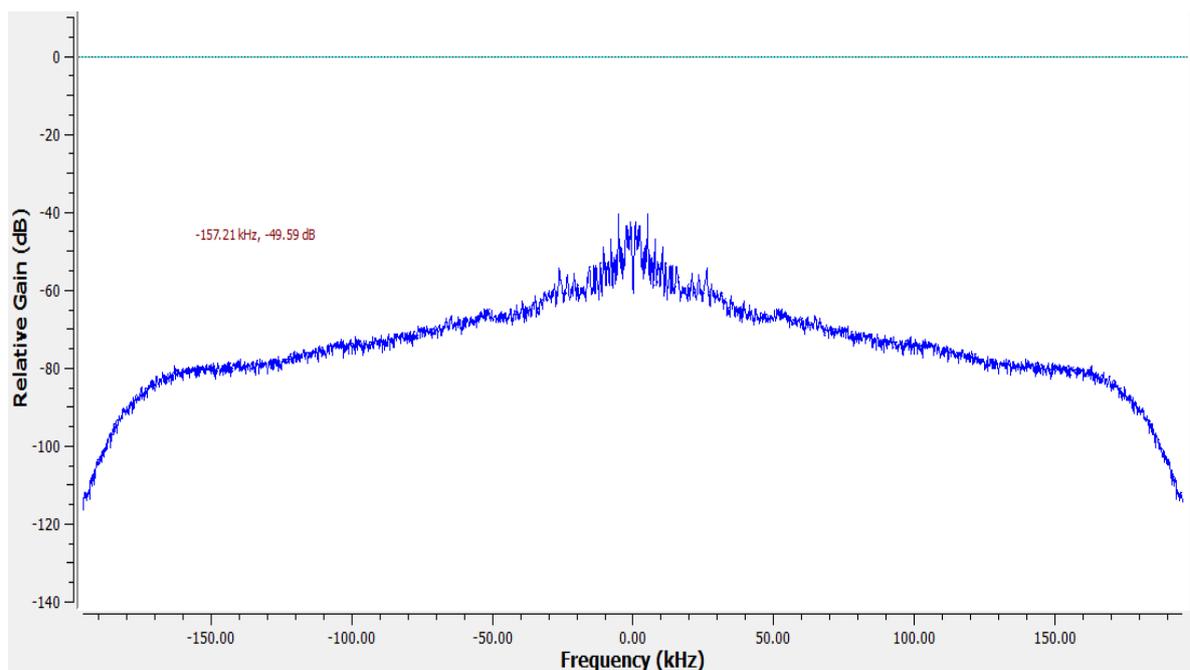


Figura 3.31. Espectro de la señal recibida con un nivel de ruido $n=3$

3.1.4 PRÁCTICA 4: DIGITALIZACIÓN DE SEÑALES

La presente práctica pretende reforzar el conocimiento de los estudiantes en cuanto al tema de digitalización de señales. Para ello, se ha implementado un flujograma interactivo con el usuario, el cual permite analizar cada una de las etapas de la digitalización.

Como se menciona en el capítulo 2, el flujograma presenta cada una de las etapas de la digitalización, es decir, muestreo, cuantificación y codificación.

3.1.4.1 Generación de onda sinusoidal

Para generar la onda sinusoidal se procede de la misma manera que en la práctica 2, es decir, mediante el bloque *Signal Source*. Para ello, se genera una senoide de 1 KHz de frecuencia y amplitud 1, como se muestra en la figura 3.32.

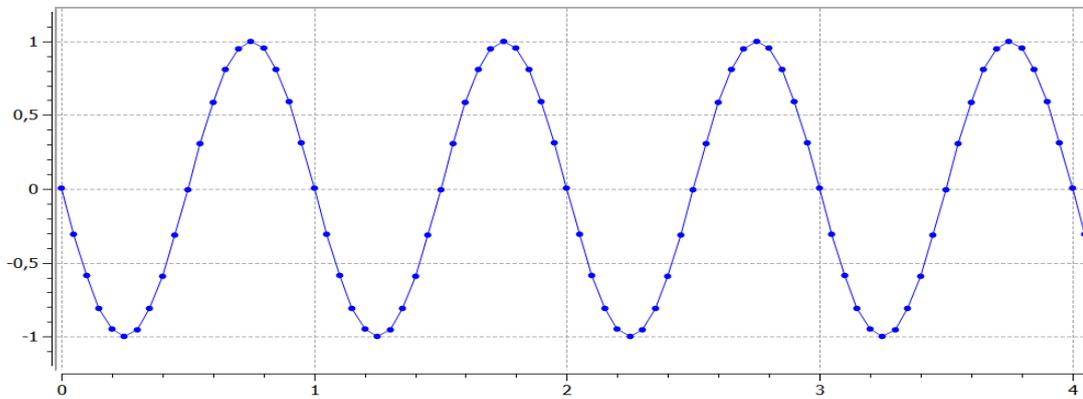


Figura 3.32. Señal de 1KHz generada mediante el bloque *Signal Source*

Ahora, para el muestreo se selecciona una frecuencia de muestreo $f_s > 2 \cdot f_{max}$, con el objetivo de tener una mejor representación de la señal mediante las muestras, como se ve en la figura 3.33.

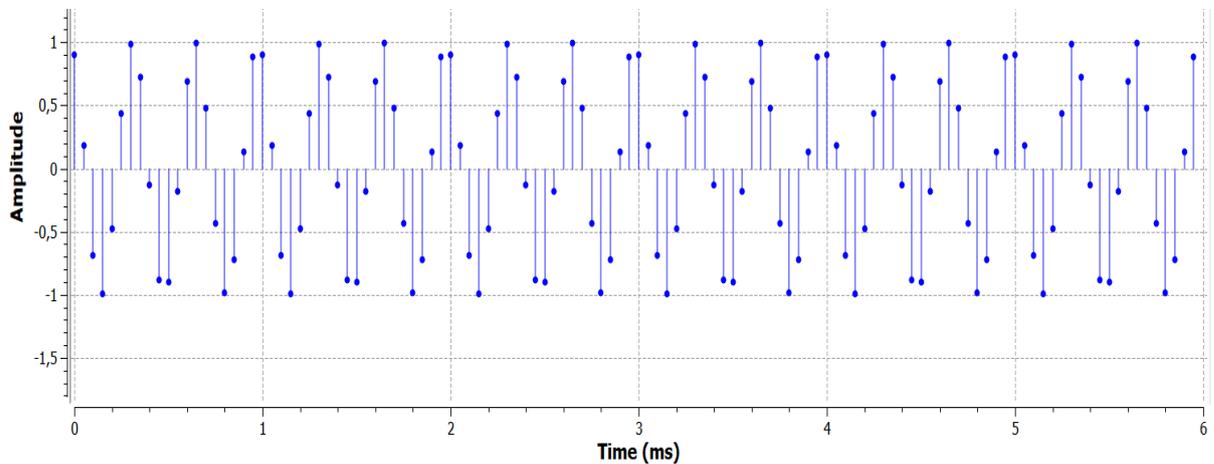


Figura 3.33. Señal muestreada a $f_s > 2 \cdot f_{max}$

Posteriormente, con el objetivo de analizar el teorema de Nyquist se plantea emplear una frecuencia de muestreo $f_s < 2 \cdot f_{max}$, produciendo una señal similar a la que se observa en a figura 3.34.

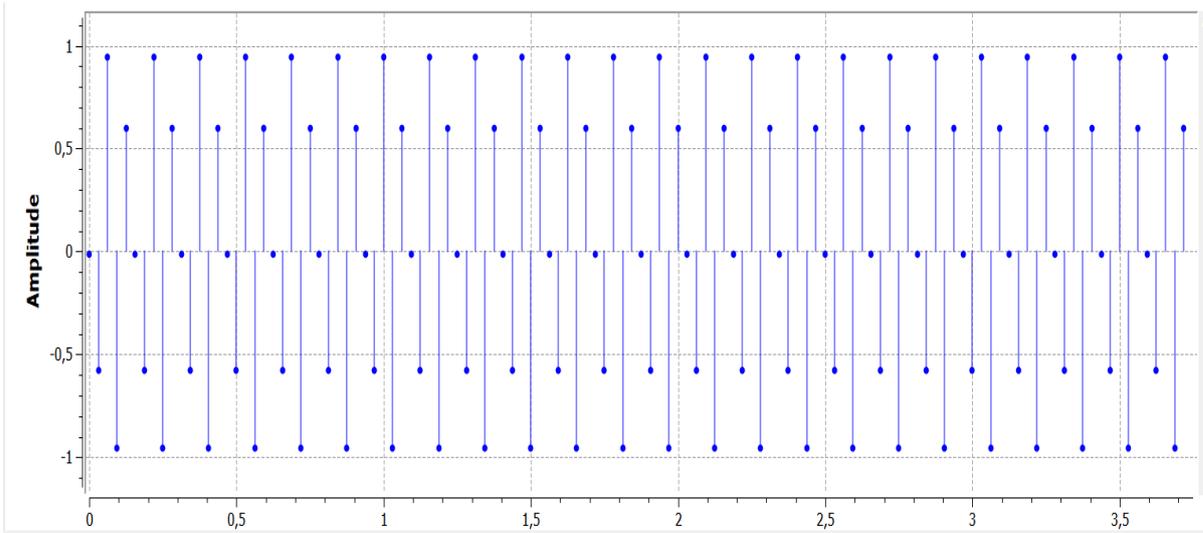


Figura 3.34. Señal muestreada a $f_s < 2 \cdot f_{max}$

Como se puede visualizar en la figura 3.34, el uso de una frecuencia de muestreo por debajo de $2 \cdot f_{max}$ ocasiona una distorsión de la señal, tanto en tiempo como en frecuencia.

3.1.4.2 Cuantización de la onda generada en el numeral 1

Para esta etapa se requiere el uso del bloque *Quantizer*, el cual realiza una cuantificación uniforme en función de "n" número de bits, con lo que se originan 2^n intervalos de cuantificación, tal como se muestra en la figura 3.35.

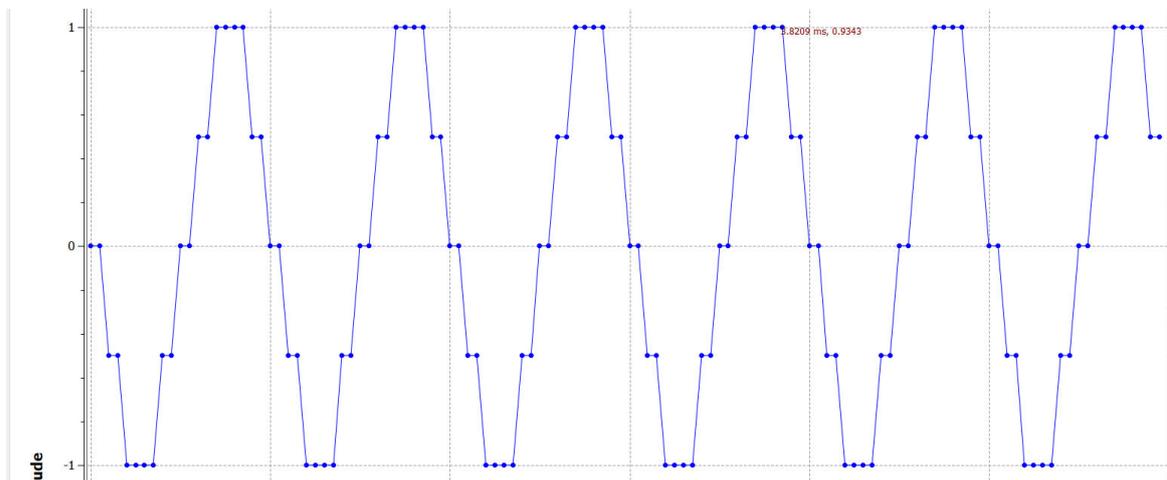


Figura 3.35. Señal cuantificada con 2 bits (4 intervalos de cuantificación)

Como se puede observar en la figura 3.36, un mayor número de bits presenta una mejor cuantificación, es decir, la señal se acerca mucho más a su forma original.

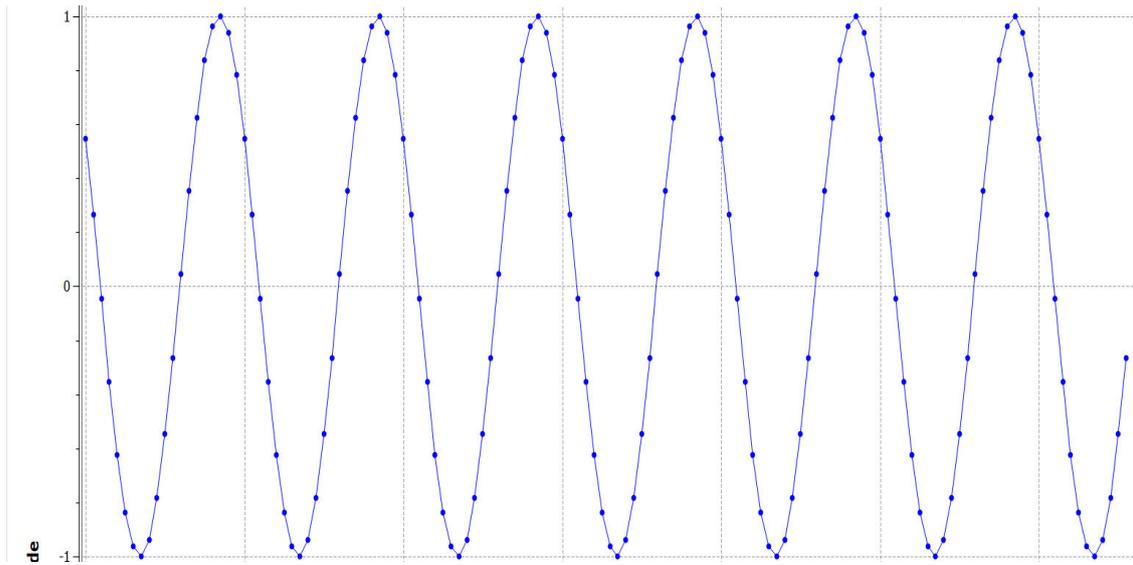


Figura 3.36. Señal cuantificada con 8 bits

A continuación, se procede con la digitalización. Para ello, se utiliza el bloque *unpack k bits*, el cual toma un símbolo entrante (muestra cuantificada) y lo representa en un formato de k bits, es decir, se lo representa en una palabra PCM de k bits, como se puede apreciar en la figura 3.37.

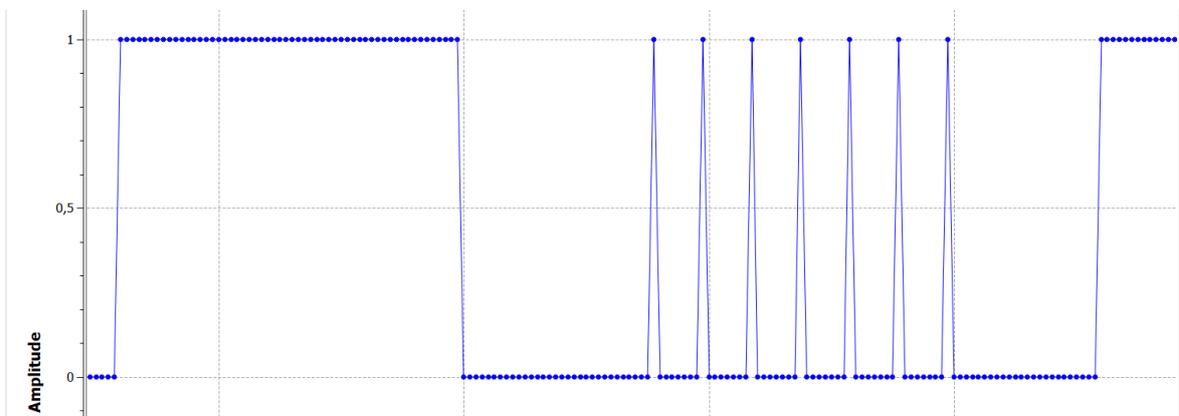


Figura 3.37. Señal seno digitalizada

3.1.4.3 Creación de un digitalizador de señales de audio

Para implementar un digitalizador de señales de voz se procede con los siguientes bloques:

- *Multiply const*: bloque que amplifica la señal para las amplitudes más pequeñas
- *Unpack K bits*: bloque que realiza la codificación en formato de 8 bits cada una de las amplitudes de la señal.
- *Float to char*: conversor de datos flotantes a caracteres.

El resultado se muestra en la figura 3.38.



Figura 3.38. Digitalizador de voz

3.1.5 PRÁCTICA 5: CUANTIFICACIÓN NO UNIFORME

3.1.5.1 Importación de librerías *Math* y *Numpy*

Para la implementación del cuantizador se necesita importar algunas librerías, necesarias para trabajar con logaritmos y vectores, para ello se utilizará el bloque *import*, tal como se muestra en la figura 3.39.



Figura 3.39. Librerías importadas

- Librería *Numpy*: necesaria para trabajar con vectores y matrices.
- Librería *Math*: necesaria para trabajar con operaciones matemáticas, tales como: logaritmos, divisiones y funciones exponenciales.

3.1.5.2 Creación de variables para los parámetros de cálculo de la muestra normalizada

Para la asignación de variables, se utilizará el bloque *Variable*. A manera de ejemplo, se utilizarán los datos indicados en la tabla 3.1.

Tabla 3.1. Valores de parámetros

Parámetro	Valor
Valor de la muestra	0.48
Valor máximo	1
Constante A	87.6
Constante U	255

Posteriormente, se incluyen los datos en el programa, mediante el bloque indicado anteriormente, como se puede apreciar en la figura 3.40.

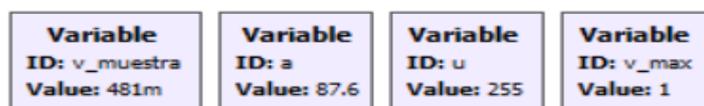


Figura 3.40. Variables con los parámetros indicados

3.1.5.3 Cálculo de la F(X) de acuerdo con la ley A

El cálculo de la F(X), requiere determinar el valor normalizado de la muestra, esto se logra dividiendo el valor de la muestra para el valor máximo, como se observa en la figura 3.41.

<u>ID</u>	v_normal
<u>Value</u>	(v_muestra/v_max)

Figura 3.41. Cálculo de la muestra normalizada

Se procede a calcular la F(x) para la ley A y ley μ mediante el uso de *variables*, como se observa en las figuras 3.42 y 3.43.

<u>ID</u>	y
<u>Value</u>	$(1+(\text{math.log}(a*v_normal)))/(1+\text{math.log}(a))$

Figura 3.42. Cálculo de F(X) según ley A

<u>ID</u>	y2
<u>Value</u>	$(\text{math.log}(1+u*v_normal))/(\text{math.log}(1+u))$

Figura 3.43. Cálculo de F(X) según ley μ

Cabe recalcar, que, para generar las funciones matemáticas como logaritmos, potencias o aproximaciones, se utilizan los comandos propios de la librería *Math*.

3.1.5.4 Cálculo del intervalo de cuantificación y del porcentaje de error en recepción

Para obtener el intervalo de cuantificación al que pertenece la muestra ingresada, se multiplica el número total de intervalos de cuantificación por el valor de la F(x) previamente calculada, como se puede observar en la figura 3.44.

<u>ID</u>	int_cuan
<u>Value</u>	128*y

Figura 3.44. Cálculo del intervalo de cuantificación

Una vez que se tiene el valor del intervalo de cuantificación, se procede a aproximarlos (figura 3.45), ya que la multiplicación antes mencionada da como resultado un número decimal.

<u>ID</u>	IC
<u>Value</u>	<code>math.floor(int_cuan)</code>

Figura 3.45. Determinación del intervalo de cuantificación con aproximación

Ahora, para estimar el valor de la muestra en la recepción, se procede con los siguientes pasos:

1. Dividir el intervalo de cuantificación para 128, como se observa en la figura 3.46.

<u>ID</u>	IC_RX_A
<u>Value</u>	<code>IC/128</code>

Figura 3.46. Normalización del Intervalo de cuantificación

2. Se obtiene la característica inversa, como se puede apreciar en la variable *nivel_rx_LEY_A* de la figura 3.47.

<u>ID</u>	nivel_rx_LEY_A
<u>Value</u>	<code>math.exp(IC_RX_A*(1+math.log(a))-1)/a</code>

Figura 3.47. Cálculo de la característica inversa

3. Se desnormaliza la muestra, multiplicando la muestra recibida por el valor máximo, como se puede apreciar en la figura 3.48.

<u>ID</u>	nivel_rx_desnormal_LEY_A
<u>Value</u>	<code>v_max*nivel_rx_LEY_A</code>

Figura 3.48. Desnormalización de la muestra

4. Se obtiene el porcentaje de error dividiendo la variable *v_normal-nivel_x_LEY_A* para la variable *v_normal*, que corresponde al valor normalizado de la muestra (figura 3.49).

<u>ID</u>	porc_errprA
<u>Value</u>	<code>(v_normal-nivel_rx_LEY_A)/v_normal</code>

Figura 3.49. Cálculo del porcentaje de error en el receptor

3.1.6 PRÁCTICA 6: MODULACIÓN M-PSK

3.1.6.1 Carga de archivo de texto mediante el bloque *File Source*

Para realizar la carga de un archivo de texto se utiliza el bloque *file source*, como se muestra en la figura 3.50.

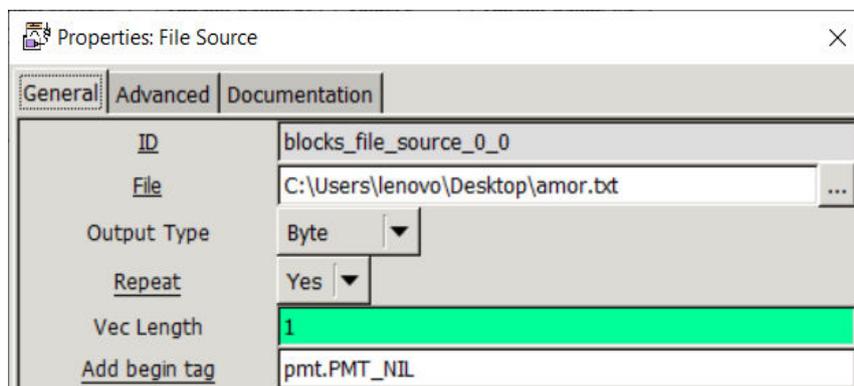


Figura 3.50. Archivo de texto cargado mediante *File Source*

Para la digitalización del archivo se utiliza el bloque *Packed to Unpacked*, tal como se muestra en la figura 3.51.

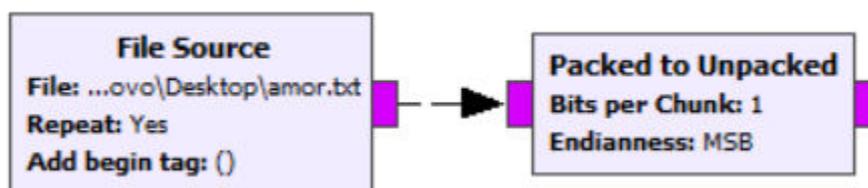


Figura 3.51. Digitalización del archivo de texto mediante *Packed to Unpacked*

3.1.6.2 Importación de las librerías *Digital*, *gr* y *Math*

La importación de las librerías, se lo hace mediante el bloque *Import*, tal como se observa en la figura 3.52.

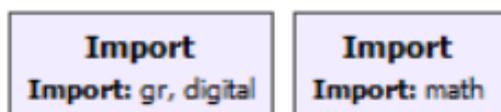


Figura 3.52. Librerías importadas en GNU Radio

3.1.6.3 Creación del modulador digital BPSK usando bloques jerárquicos

Una vez digitalizada la señal, se procede con el mapeo de símbolos a Código Gray mediante el bloque *Map*, tal como se presenta en la figura 3.53.

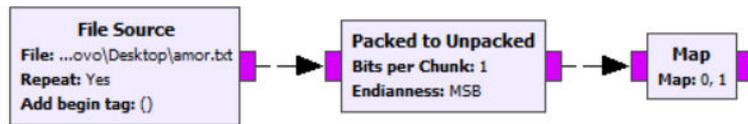


Figura 3.53. Mapeo de símbolos mediante el uso del bloque *Map*

Posteriormente, se realiza la codificación de la señal mediante el bloque *Differential Encoder* y su modulación, para la cual se utilizará la variable *constelacion* (Figura 3.54), la cual será creada de acuerdo con el procedimiento indicado en el trabajo preparatorio.

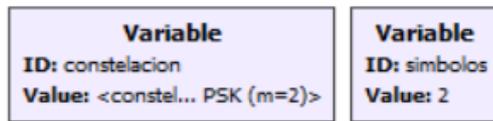


Figura 3.54. Variable *constelacion*

Finalmente, el modulador se muestra en la figura 3.55.

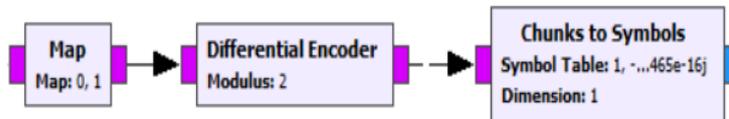


Figura 3.55. Modulador BPSK implementado

3.1.6.4 Visualización de los resultados de cada etapa del modulador

Para la visualización de los resultados se emplearán los bloques utilizados en prácticas anteriores, tales como *QT Time Sink* o *QT Frequency Sink*, como se muestra en las figuras 3.56, 3.57 y 3.58.

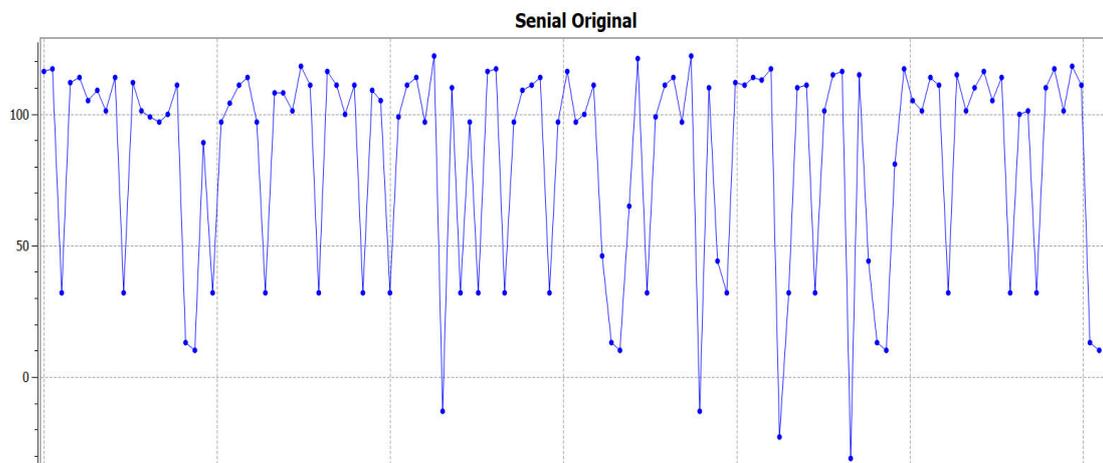


Figura 3.56. Señal original (archivo de texto)

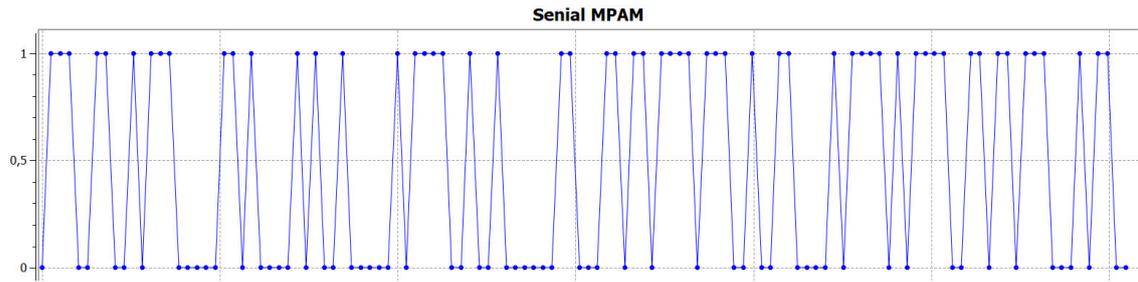


Figura 3.57. Señal digitalizada

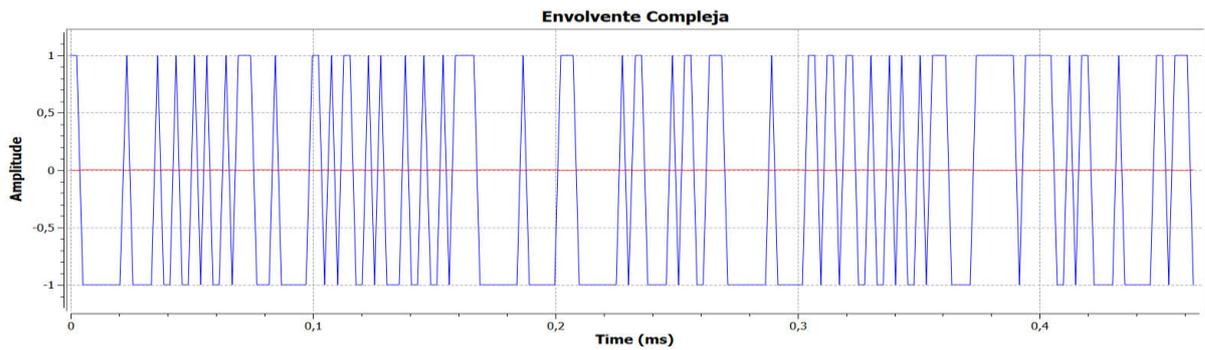


Figura 3.58. Señal envolvente compleja BPSK

A continuación, se presenta el diagrama de constelación y una aproximación de la señal RF modulada, como se muestra en las figuras 3.59 y 3.60, respectivamente.

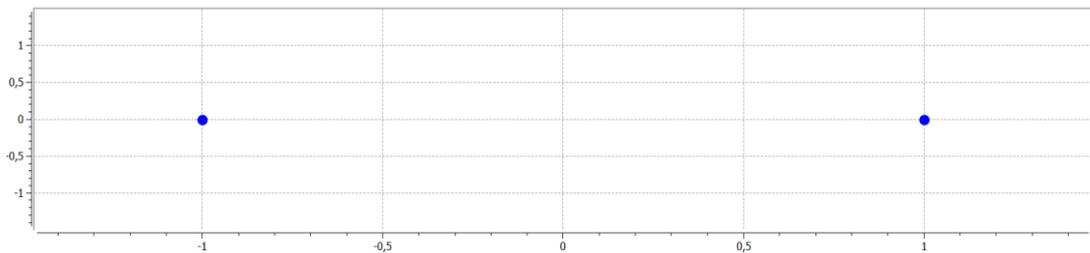


Figura 3.59. Diagrama de constelación BPSK

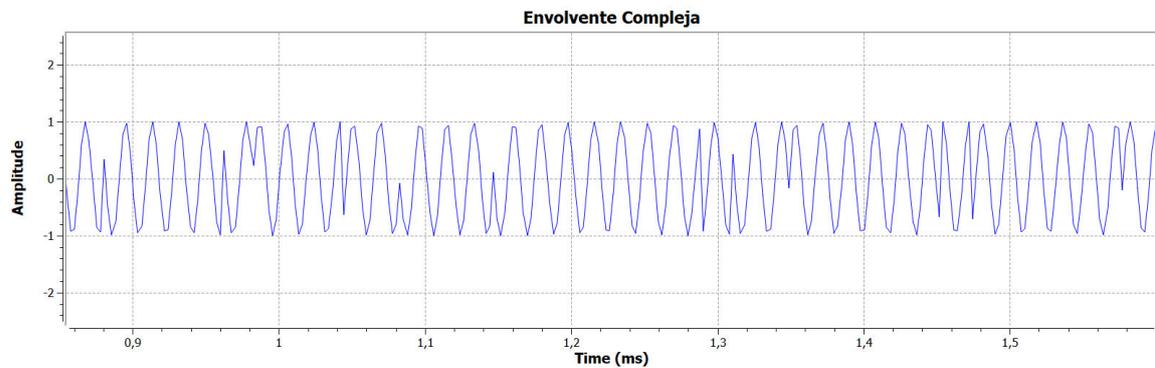


Figura 3.60. Aproximación de la onda modulada RF

3.1.7 PRÁCTICA 7: DEMODULACIÓN M-PSK

3.1.7.1 Creación del demodulador digital BPSK

Para la elaboración del demodulador se toma en cuenta las etapas mostradas en el diagrama de bloques de la hoja guía.

Para el caso del muestreo, se utiliza el bloque *e_diezmador_cc*, el cual toma un cierto número de muestras de la señal recibida.

Posteriormente, se utiliza el bloque *Costas Loop*, el cual corrige la desviación de frecuencia de la señal recibida, representando de mejor manera el diagrama de constelación.

Seguido de esto, se sigue el mismo proceso de demodulación indicado en el capítulo II, teniendo como resultado el indicado en la figura 3.61.

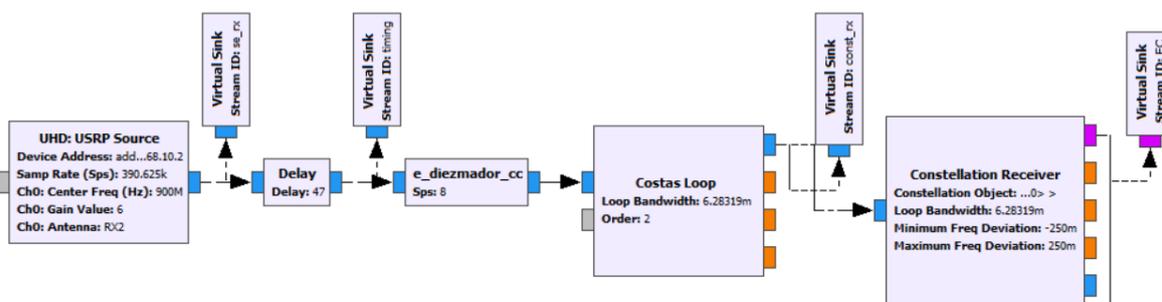


Figura 3.61. Demodulador BPSK

3.1.7.2 Asignación y configuración de variables en el demodulador

Al emplear un número determinado de muestras en el transmisor (en este caso 8), se utiliza el mismo número en el receptor (figura 3.62), esto mediante la variable *Sps*, el cual determina, además, el número de muestras a tomar mediante el bloque *e_diezmador_cc*.

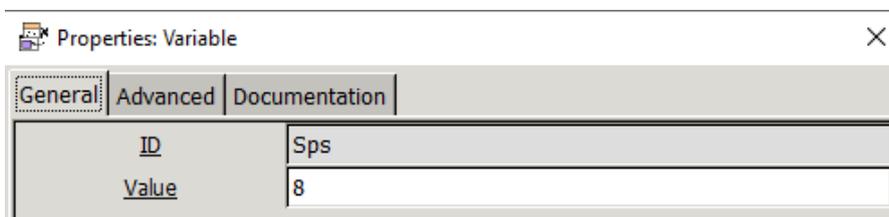


Figura 3.62. Configuración de número de muestras

Posteriormente se realiza el proceso inverso al modulador, es decir, la decodificación diferencial, el mapeo, tal como se muestra en la figura 3.63.

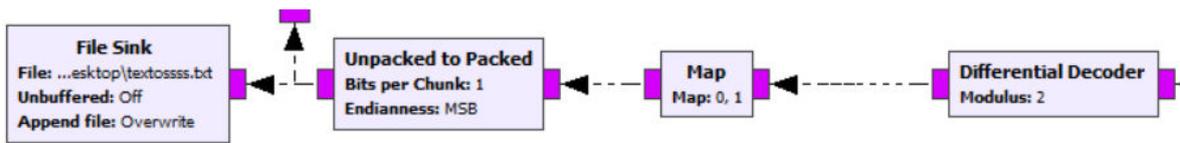


Figura 3.63. Decodificación y mapeo en el receptor

3.1.7.3 Transmisión y recepción de un archivo de texto utilizando modulación BPSK

Para la visualización de los resultados se emplean los bloques utilizados en prácticas anteriores, tales como *QT Time Sink* o *QT Frequency Sink*.

En la figura 3.64 se presenta la señal recibida por el equipo USRP en el dominio del tiempo, en tanto que en la figura 3.65 se presenta la señal en el dominio de la frecuencia, así como su diagrama de constelación en la figura 3.66.

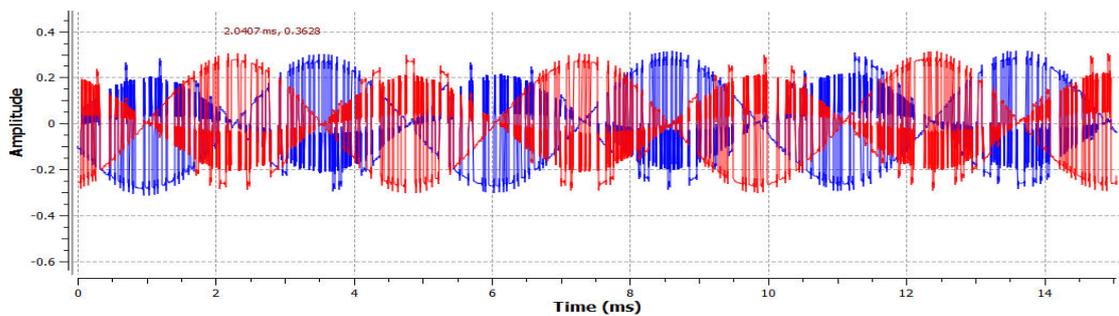


Figura 3.64. Señal RF recibida

Como es notorio en la figura 3.64, la señal recibida no se parece a la señal modulada BPSK, esto es debido a que la señal emitida por el USRP está a una frecuencia de muestreo muy elevada, por lo que presenta esta forma sinusoidal.

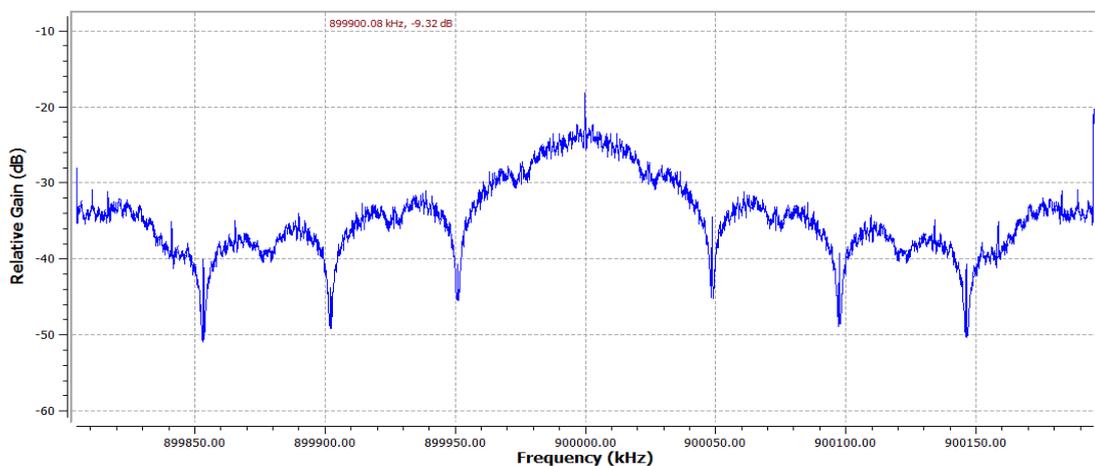


Figura 3.65. Espectro de señal recibida

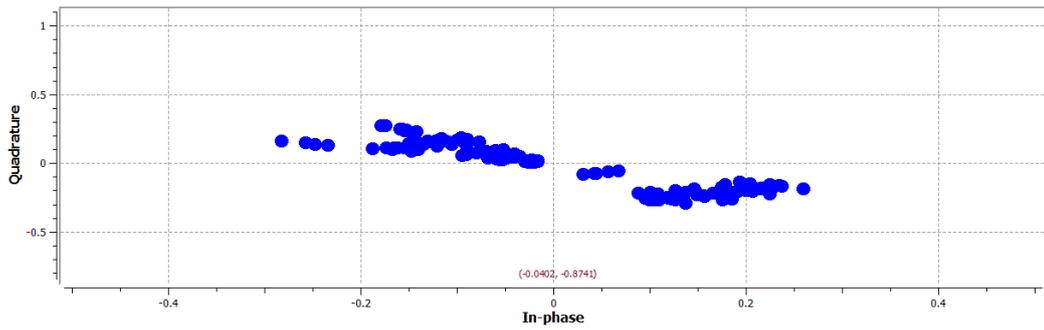


Figura 3.66. Diagrama de constelación recibido

Posteriormente, se tiene el procesamiento de la señal mediante los bloques *e_diezmador_cc*, *Costas Loop* y *Constellation Receiver*, dando como resultado el mostrado en las figuras 3.67 y 3.68.

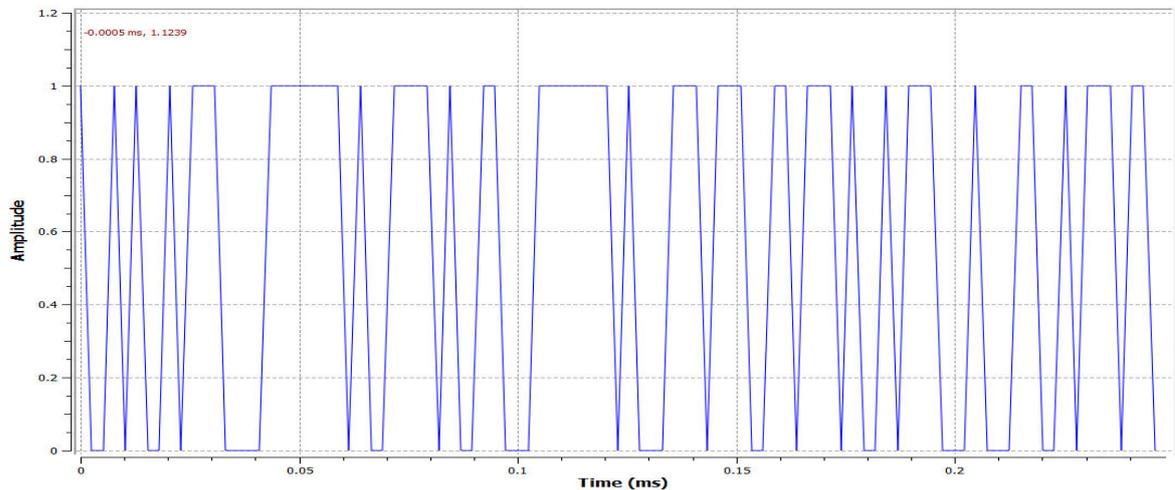


Figura 3.67. Recuperación de la EC (bloque *Constellation Receiver*)

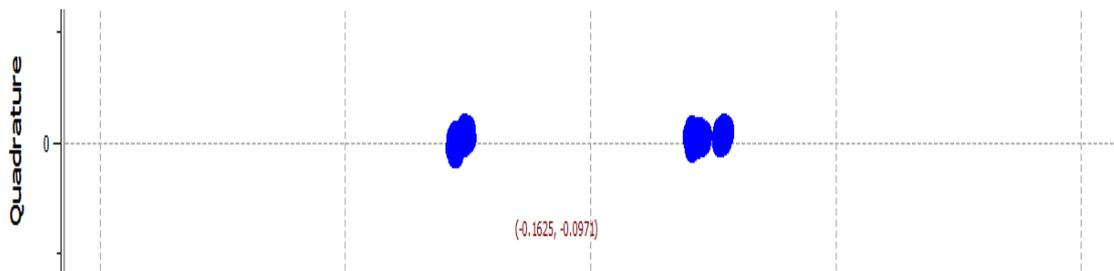


Figura 3.68. Diagrama de constelación recibido (*Costas Loop*)

Finalmente, hay un parámetro a tomar en cuenta para la correcta recuperación de datos, este es el *Time Alignment*, el cual se corrige mediante el parámetro *Delay*. Finalmente se muestra el resultado en las figuras 3.69 y 3.70.

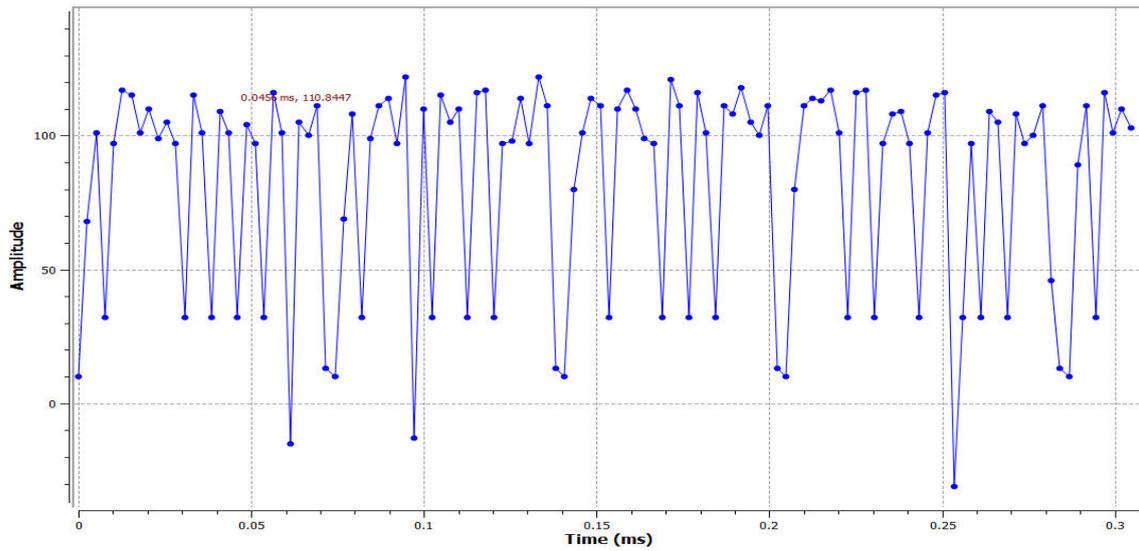


Figura 3.69. Datos recuperados

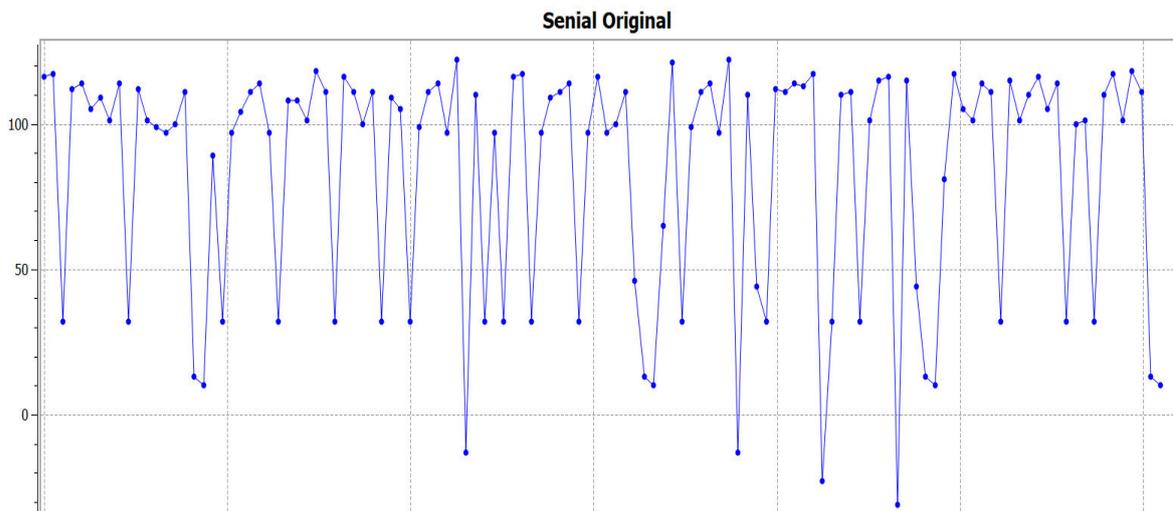


Figura 3.70. Señal original

Se observa que el archivo original es igual al recibido; sin embargo, para poder observar de mejor manera si el archivo llegó bien, se abre el archivo txt recibido, tal como se lo presenta en la figura 3.71.

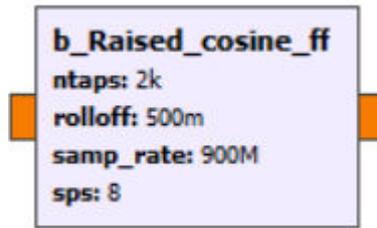


Figura 3.73. Bloque *b_Raised_cosine_ff*

3.1.8.3 Creación de un analizador de diagramas de ojo mediante bloques jerárquicos

Para el desarrollo del analizador de diagrama de ojo es necesario el uso de los bloques *QT GUI Time Sink*, *Delay*, *Pad Source* y *Pad Sink*, cuyas funciones son descritas:

- *QT Time Sink*: visualizador del diagrama resultante.
- *Delay*: bloque encargado de representar cada copia de la salida del filtro.
- *Pad Source & Pad Sink*: representan las entradas y salidas del analizador.

Finalmente, en la figura 3.74 se presenta el graficador de diagrama de ojo.

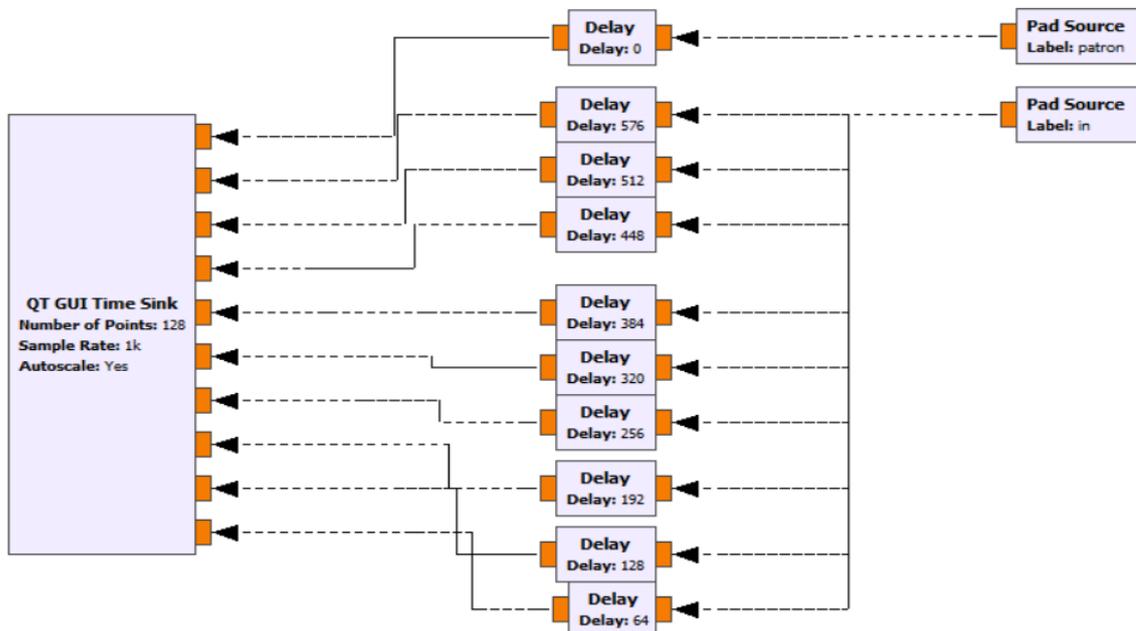


Figura 3.74. Analizador de diagramas de ojo

3.1.8.4 Visualización de diagramas de ojo correspondientes a modulaciones digitales

El flujograma necesario para esta práctica tendrá 3 etapas, como se observa en la figura 3.75.

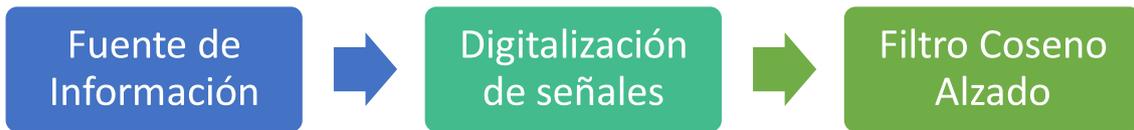


Figura 3.75. Etapas del flujograma a desarrollar

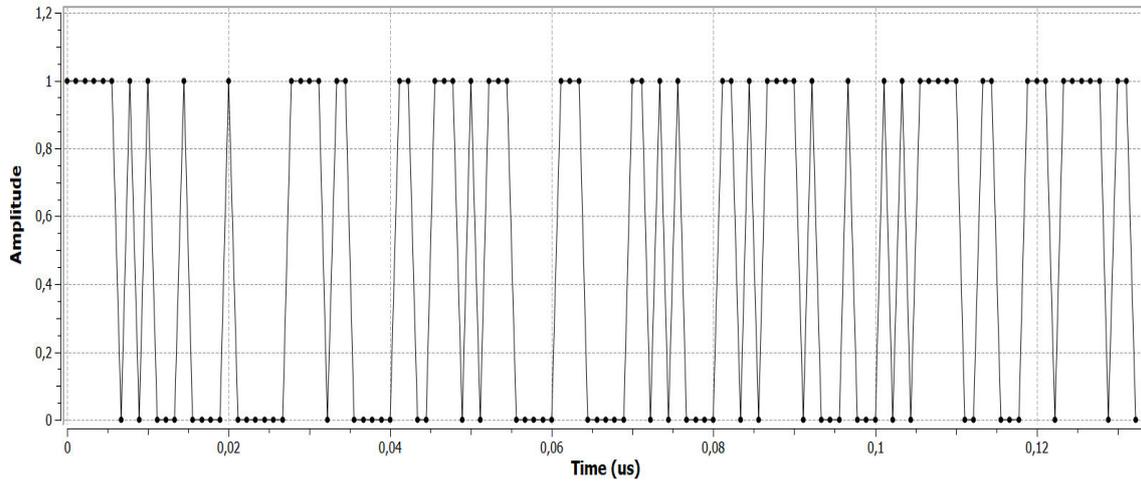


Figura 3.76. Secuencia binaria creada por el bloque *Random Source*

Luego de generar la señal binaria, se procede con la implementación del filtro RRC (*Raised Cosine*), con un factor de *roll off* de 0.75 y *sps* de 8, el cual presenta como resultado una señal *Sinc*, como se puede apreciar en las figuras 3.77 y 3.78.

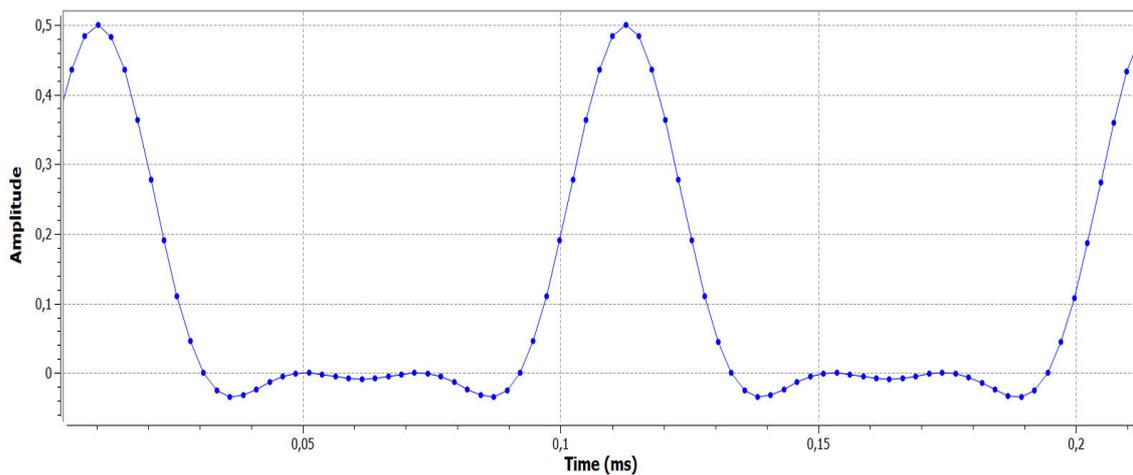


Figura 3.77. Respuesta del filtro coseno alzado (*roll off* = 0.75)

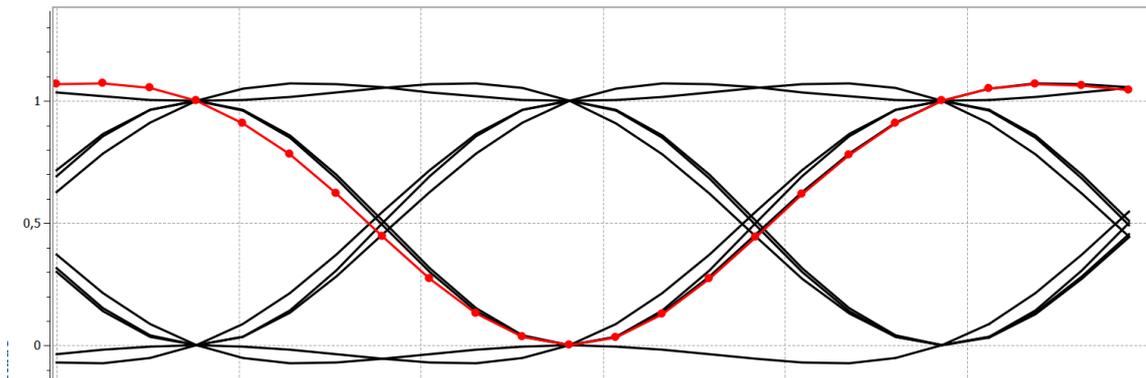


Figura 3.78. Diagrama de ojo correspondiente a fuente digital binaria (*roll off* = 0.76)

A continuación, se presentan los diagramas de ojo correspondientes a modulaciones digitales, para ello, se hará uso del modulador digital desarrollado en la práctica 6, tal como se muestra en las figuras 3.79 y 3.80.

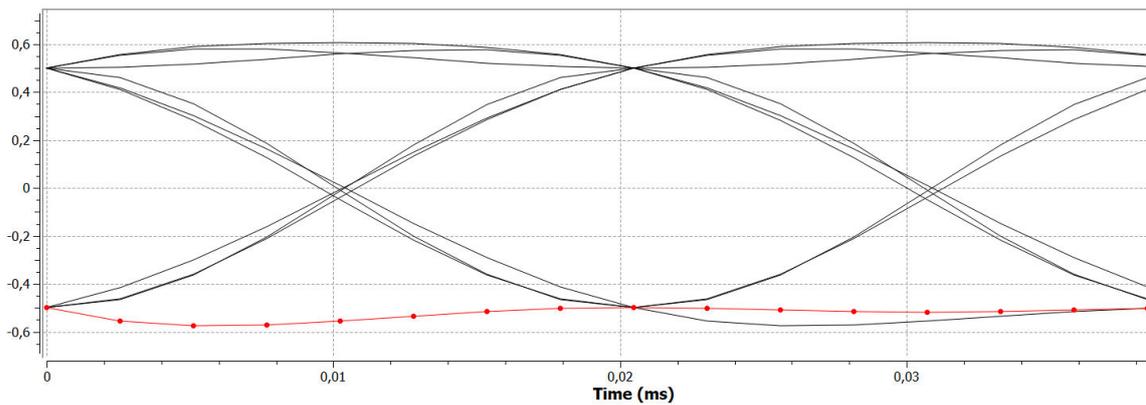


Figura 3.79. Diagrama de ojo de modulación BPSK (*roll off* = 0.75)

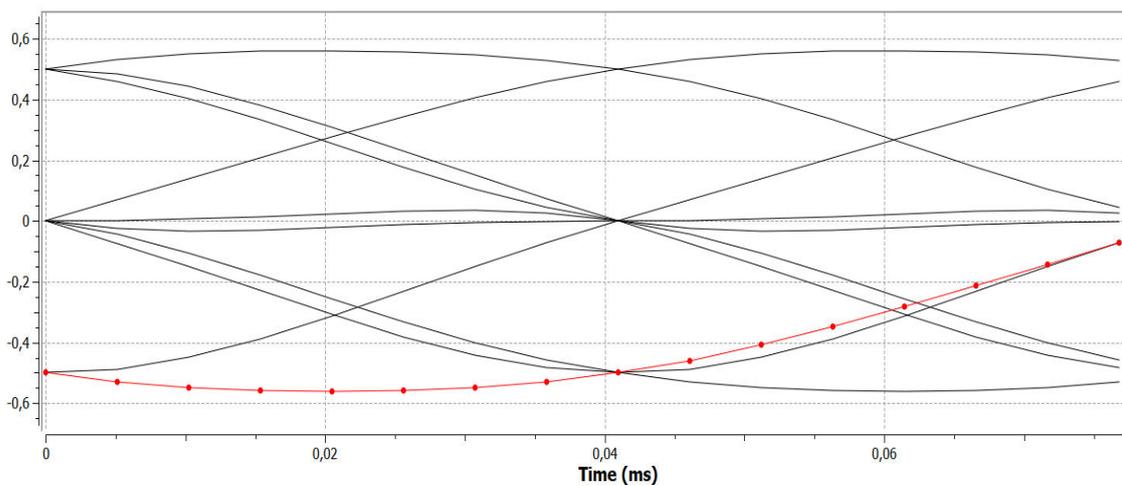


Figura 3.80. Diagrama de ojo de modulación QPSK (*roll off* = 0.75)

3.1.9 PRÁCTICA 9: DSSS (*DIRECT SEQUENCE SPREAD SPECTRUM*)

3.1.9.1 Carga y digitalización de un archivo de texto mediante los bloques *File Source* y *Packed to Unpacked*

Para cargar un archivo de texto se utiliza el bloque *file source*, el cual se empleó en la práctica 6, tal como se indica en la figura 3.81.

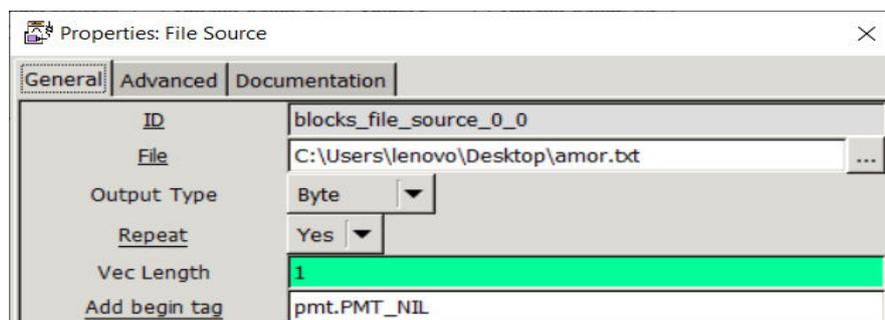


Figura 3.81. Archivo de texto cargado mediante *File Source*

Para la digitalización del archivo se utiliza el bloque *Packed to Unpacked*, tal como se muestra en la figura 3.82.

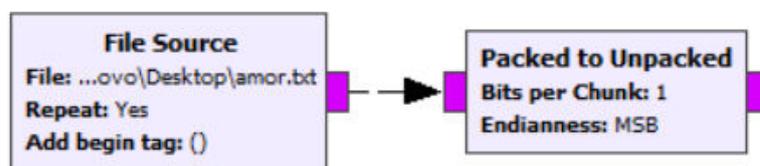


Figura 3.82. Digitalización del archivo de texto mediante *Packed to Unpacked*

3.1.9.2 Generación de la secuencia DSSS

Como se pudo observar en el capítulo 2, para implementar un sistema DSSS, es necesario realizar la operación XOR entre la secuencia de datos digitalizada y un código de PseudoRuido generado mediante el bloque *GLFSR Source*; el resultado se muestra en la figura 3.83.

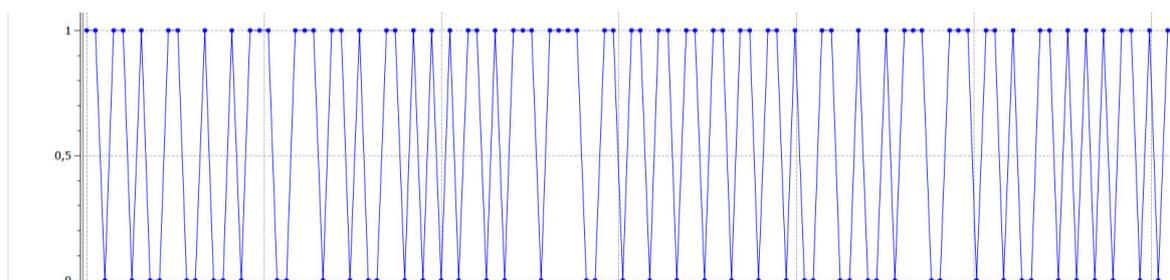


Figura 3.83. Secuencia DSSS generada

Sin embargo, para verificar que se trata de una secuencia DSSS, a manera de ejemplo, se utilizarán una secuencia aleatoria de bits y una secuencia de PseudoRuido generada por el bloque GLFSR de GNU Radio, a las cuales se les aplicará la operación lógica XOR.

- Secuencia de bits: 00101101
- Secuencia PseudoRuido: 01101101
- Resultado del XOR: 01000000

Esto al realizarlo en GNU Radio se tienen los resultados, mostrados en las figuras 3.84, 3.85 y 3.86.

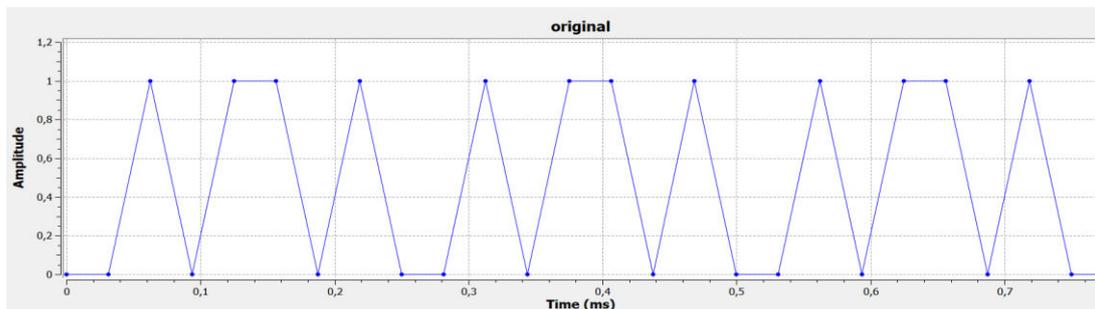


Figura 3.84. Secuencia de datos original

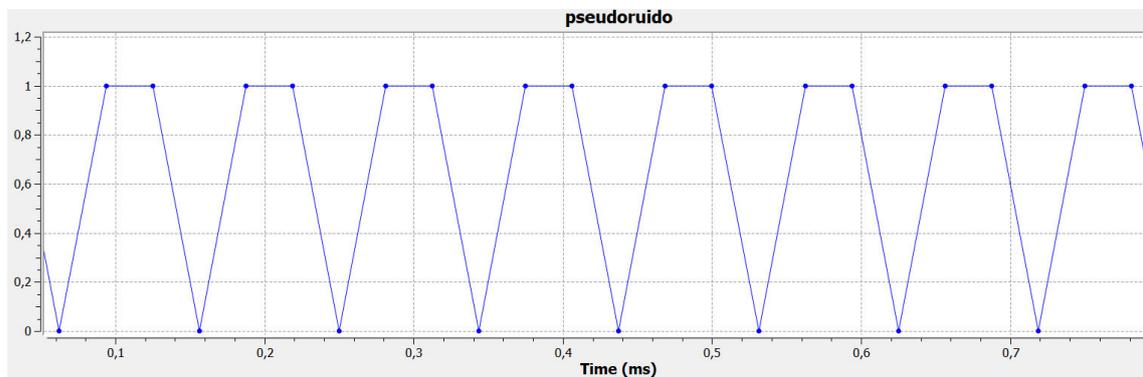


Figura 3.85. Secuencia de PseudoRuido generada mediante el bloque *GLFSR Source*



Figura 3.86. Secuencia DSSS generada mediante la operación XOR

3.1.9.3 Modulación y transmisión de la señal DSSS utilizando un sistema BPSK

Previo a la transmisión, la señal debe ser modulada. Por esta razón, el modulador BPSK desarrollado en prácticas anteriores es necesario utilizarlo en este apartado. Los resultados del modulador se presentan en las figuras 3.87, 3.88 y 3.89.

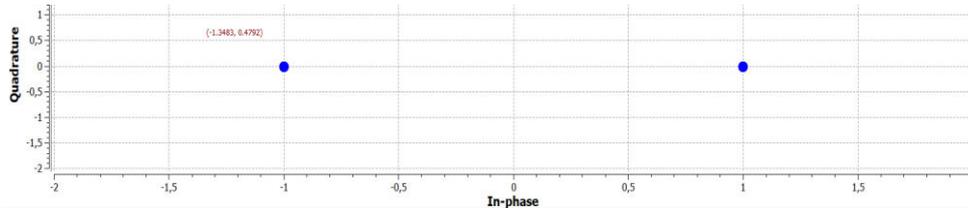


Figura 3.87. Diagrama de constelación BPSK

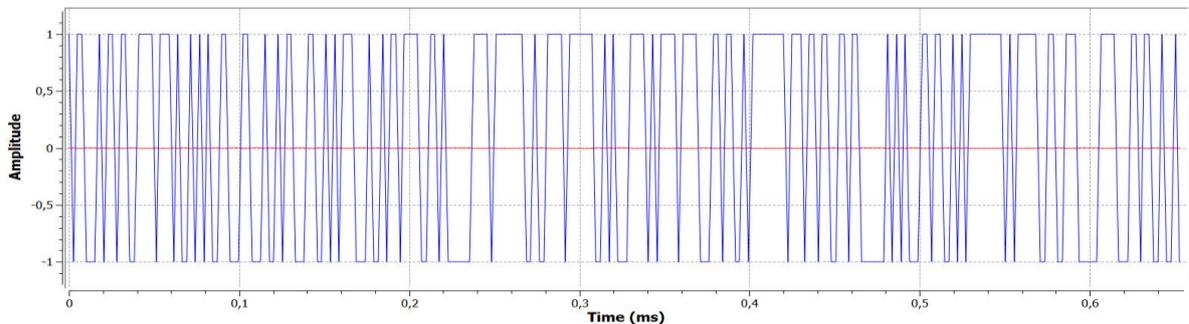


Figura 3.88. Envoltura compleja generada

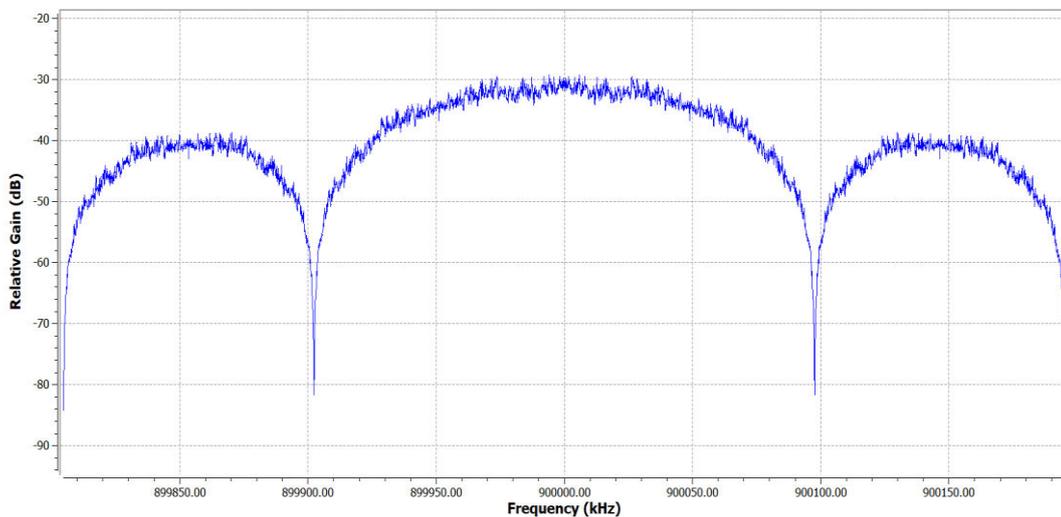


Figura 3.89. Espectro BPSK generado

3.1.9.4 Demodulación y recepción de la señal DSSS

Para la parte de recepción se realiza el mismo proceso que en la práctica 7. Como se puede apreciar en las figuras 3.90 y 3.91 se muestra la señal que recibe el USRP, en el dominio del tiempo y de la frecuencia.

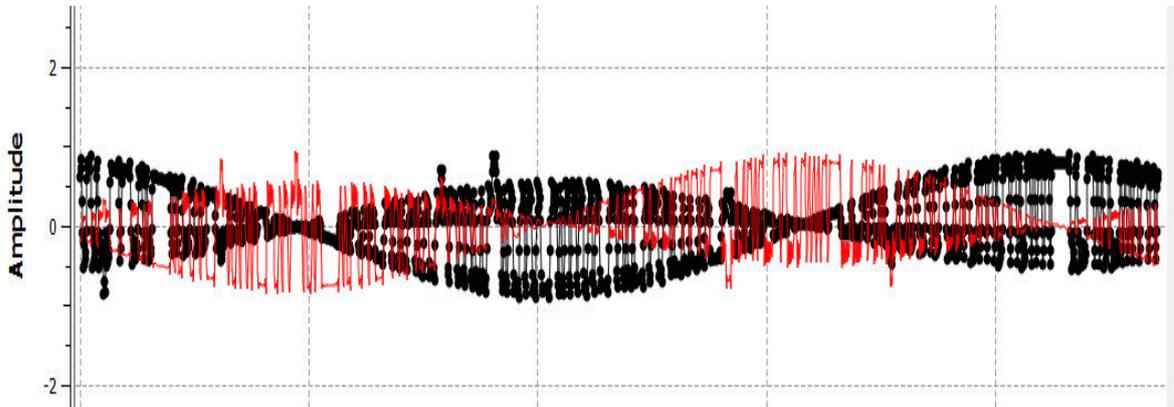


Figura 3.90. Señal RF recibida

Como se mencionó en la práctica 6, la forma que presenta la señal de la figura 3.90 no es similar a la señal modulada BPSK. Esto debido a que, la señal está muestreada a una frecuencia muy elevada, por lo que, es un poco complicado el apreciar su forma.

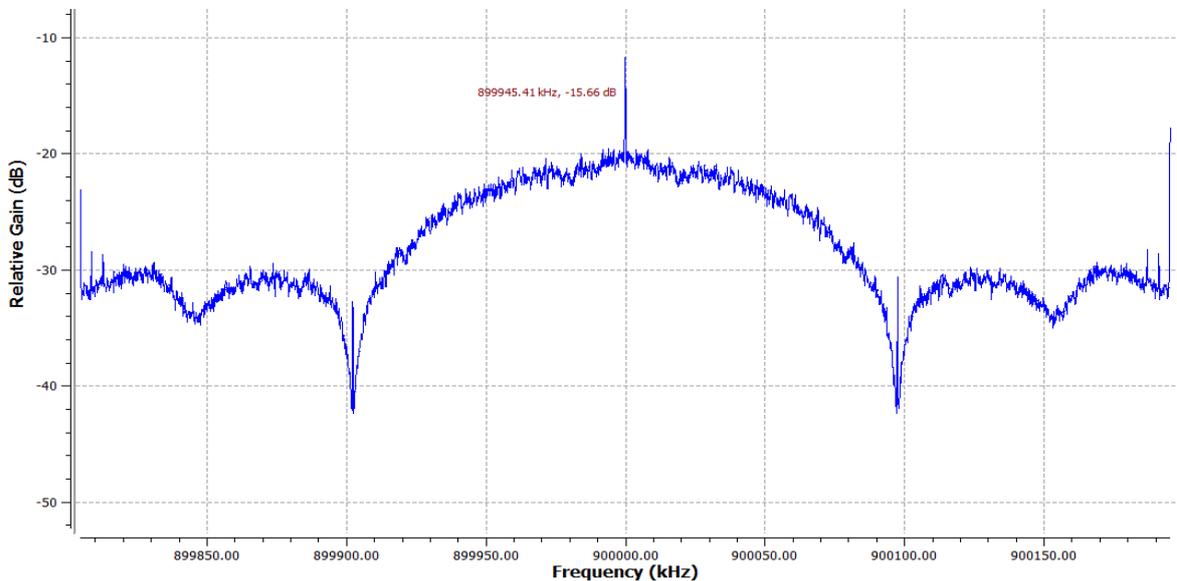


Figura 3.91. Espectro recibido por el USRP

Al igual que en prácticas anteriores, se utiliza el bloque *Costas Loop* para corregir la desviación de frecuencia; este resultado se muestra en el diagrama de constelación de la figura 3.92.

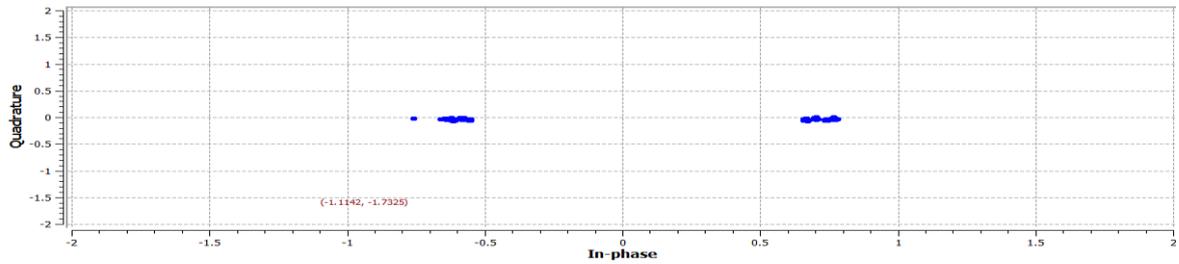


Figura 3.92. Diagrama de constelación (*Costas Loop*)

Como se puede apreciar en la figura 3.93, se tiene la envolvente compleja recuperada en el receptor, esto a la salida del bloque *Costas Loop*.

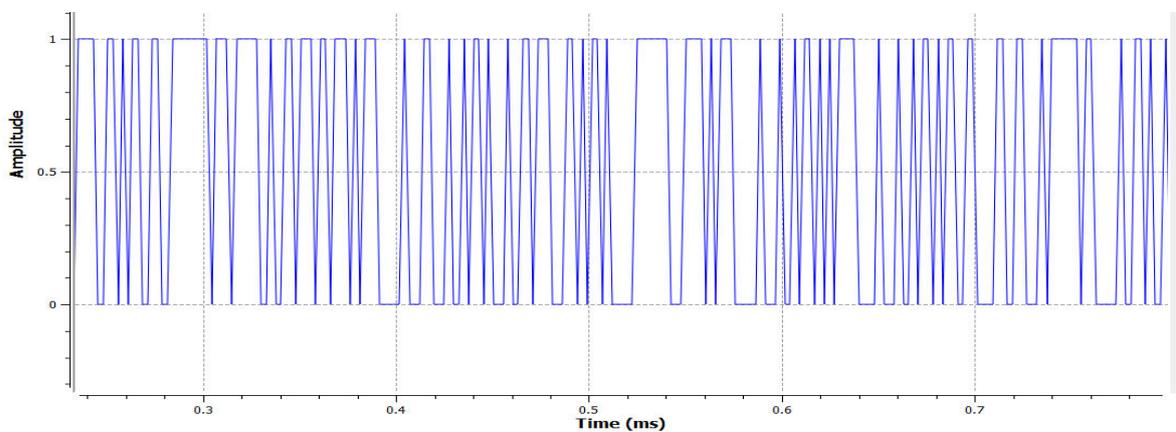


Figura 3.93. Envolvente compleja recuperada

Posteriormente, a la señal recibida se le aplica un paso más, es decir, se multiplica por el mismo código de PseudoRuido utilizado en el transmisor, esto para recuperar los datos transmitidos, tal como se indica en la figura 3.94.

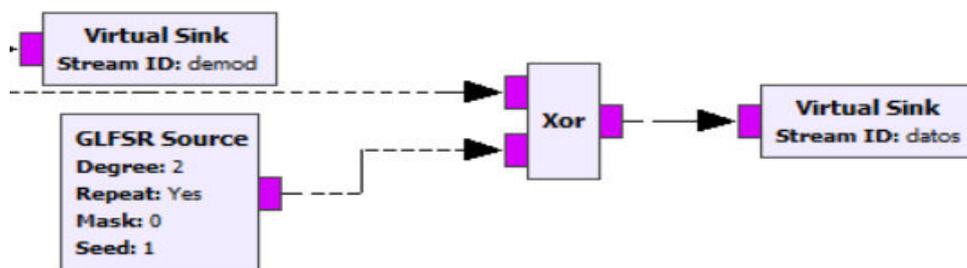


Figura 3.94. Recuperación de datos

Posteriormente, se tiene los datos recibidos en el dominio del tiempo (figura 3.95).

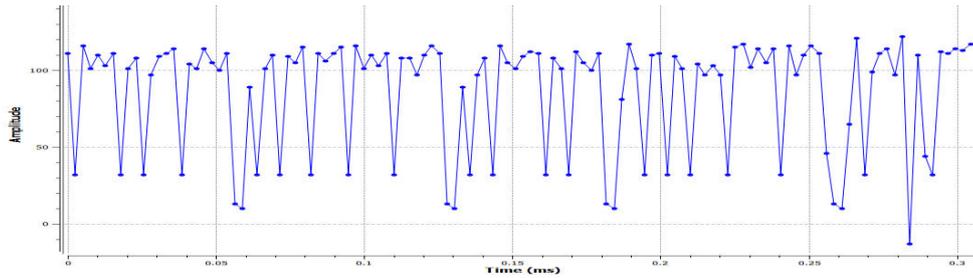


Figura 3.95. Datos recuperados

Para verificar que el texto recibido sea el mismo que el transmitido se abre el archivo de texto guardado mediante el bloque *File Sink*, tal como se muestra en la figura 3.96.

```

áT+#Ú0^A&á_&i[b]&iÚ&aÚ|èÐÏl}ŠKW,,K_0aZ
K] kÚ,,iú°^E↗oÔ iÔ+K[ b]ÉWÆbýnZ0Mk3ó× bÚ[b}
K[,,nZ]]èŠKQ0KW,,KV,`Z&âÚNkÚ,,iú°^M
K\0b]0çÚ ÈUbU↗/ýiy,,bxÆbú°^E↗oß0iyiÚ0EiÔ|é_&:
Æ%maçe0»o0Y00ç-€°e†·mÖ0m†%1;Z|0Yma0yÚm†²l0è
Z-€·l0°-€Ú-...·m†Ê< [aa»i;Z|0·aa°m00h0x%6'y0ç:i
Yo fui tu primer amigo,
También tu primer pecado
Y ahora llevo todo mi corazón a tu amor a
Ay corazón, porque no estás,
Quiero sentir de nuevo
Tu calor entre mis brazos.
Ay corazón, porque te vas,
Y por tu regreso mi amor,
Yo sigo esperando.
De ausencia se me ha teñido
El corazón sin tu abrazo
Pero nunca yo te olvido
Porque tu alma está a mi lado.
Yo tengo el amor herido
Y en mis ojos tengo llanto
Y al tiempo le pido
Que no me haga sufrir tanto.
Ay corazón, porque no estás,
Quiero sentir de nuevo
Tu calor entre mis brazos.
Ay corazón, porque te vas,
Y por tu regreso mi amor,
Yo sigo esperando.

Me muero porque te has ido,
Me muero porque te extraño
Y te llora mi corazón y te estoy amando.
Quisiera mirar tus ojos,
Quisiera sentir tus labios
Pero estás muy lejos

```

Figura 3.96. Texto recibido

Como se puede apreciar en la figura, al inicio del texto aparecen unos símbolos que no pertenecen al texto, esto es debido a los efectos del canal, los cuales son solucionados de la misma manera que en la práctica 7, es decir con un corrector de *Delay*.

4. CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

- Luego del trabajo realizado se puede apreciar que se cumplió con el objetivo general del mismo, es decir, implementar las 9 prácticas de laboratorio para la materia de Comunicación Digital utilizando Radio Definida por Software y GNU Radio
- Como se mencionó en el capítulo 2, el uso de bloques para adaptación de la señal es recomendable para tener una mejor transmisión ya que al trabajar con dispositivos USRP, éstos presentan ciertas condiciones que debe cumplir la señal para poder ser transmitida. Por ejemplo, debe tener una forma continua y ser muestreada a una frecuencia determinada por el equipo.
- Al momento de transmitir un archivo de audio o simplemente señales periódicas, se tienen dos opciones, la primera es transmitiendo sin ningún tipo de modulación previa; la segunda opción es realizando una modulación previa, como por ejemplo FM, tal como se mostró en el capítulo 3.
- Al momento de recibir información con los dispositivos USRP, ya sea archivos de texto o archivos de audio, se debe tener en cuenta un concepto muy importante que es el *Time Alingment*, parámetro que debe ser configurado en el receptor, con el fin de tener un sincronismo en tiempo, de esta manera la información puede ser recuperada de manera correcta en el receptor.
- Los dispositivos que funcionan con tecnología SDR presentan una gran versatilidad para desarrollar diferentes aplicaciones, tales como comunicaciones digitales o comunicaciones analógicas. Esto, sin alterar los componentes del hardware, únicamente se necesita modificar la parte de software, que en este caso es manejada por GNU Radio.
- El uso de dispositivos SDR en el ambiente académico permite a los estudiantes comprender de mejor manera un sistema de transmisión real, ya que, al utilizar equipos físicos se pueden apreciar las diferentes anomalías que afectan a la información al viajar por el canal. Por lo que, los estudiantes deberán buscar soluciones a estos inconvenientes.
- Al momento de trabajar con dispositivos USRP, existen diferentes plataformas de software con las que se puede interactuar. Sin embargo, se utilizó GNU Radio

debido a que, presenta una facilidad de manejo superior a otros, posee una interfaz muy intuitiva y permite desarrollar una gran cantidad de aplicaciones mediante una programación por bloques. Además, en el caso de no poseer algún bloque, el usuario puede crearlo.

4.2 RECOMENDACIONES

- Al momento de transmitir y recibir señales con los dispositivos USRP es importante tener en cuenta la distancia entre éstos. Ya que, al tener los equipos uno frente al otro, la potencia que emanan los equipos puede resultar en una distorsión en la señal recibida, por tanto, se recomienda una distancia prudente entre éstos, como por ejemplo 3 metros.
- Cuando se trata de un enlace de comunicaciones y se utiliza cualquier tipo de modulación, es recomendable que el archivo de audio que se emplea sea una canción ya grabada y no una grabación de voz. Esto debido a que una canción está mezclada y ecualizada de manera adecuada, lo que no sucede con una grabación de voz, ya que un celular al momento de grabar la voz capta no solo la voz, sino también el ruido presente en el ambiente, provocando de esta manera, que el ruido sea difícil de eliminar en el flujograma.
- Al momento de trabajar en un ambiente de laboratorio o simulación, se recomienda que el flujograma no tenga una cantidad excesiva de bloques, ya que esto podría ocasionar que el proceso se torne demasiado lento o que la máquina colapse por el proceso que se intenta llevar a cabo.
- Como una manera de verificar el funcionamiento adecuado de los flujogramas desarrollados en éste o en futuros proyectos de titulación, es recomendable realizar el cálculo del BER (*Bit Error Rate*), ya que este parámetro muestra el grado de eficiencia de las aplicaciones desarrolladas.
- Como una manera de ampliar el trabajo realizado en este proyecto, se recomienda realizar un analizador de diagramas de ojo mediante código Python, ya que al trabajar con bloques se tiene ciertas limitaciones, como por ejemplo el número de curvas que se utilizan para la representación del diagrama de ojo. Por otra parte, al trabajar con un bloque embebido mediante código Python, se puede fácilmente tomar más de 100 curvas para graficar el diagrama de ojo y tener una mejor visualización de las diferentes anomalías que se pueden analizar en él.

- Como una recomendación de un trabajo futuro, se propone complementar el análisis de canal de comunicaciones, es decir, analizar diferentes anomalías que presenta el canal inalámbrico, tales como desvanecimiento o multitrayecto; este proyecto complementarían el aprendizaje de las comunicaciones digitales inalámbricas.

5. REFERENCIAS BIBLIOGRÁFICAS

- [1] Federal Communications Commission, «Authorization and Use of Software Defined Radios,» 14 septiembre 2001. [En línea]. Available: https://transition.fcc.gov/Bureaus/Engineering_Technology/Orders/2001/fcc01264.pdf. [Último acceso: 23- noviembre- 2019].
- [2] GNU-Radio, «About GNU - Radio,» 2019. [En línea]. Available: <https://www.gnuradio.org/about/>.
- [3] Juan Pablo Montero Hidalgo, IMPLEMENTACION DE UN SISTEMA DE COMUNICACIONES BASADO EN SDR, MADRID: UNIVERSIDAD AUTONOMA DE MADRID, 2014.
- [4] Hiroshi. Harada y Ramjee Prasad, Simulation and Software Radio for mobile communications, Primera ed., Boston: Artech House, 2002.
- [5] W. Tuttlebee, Software Defined Radio, PRIMERA ed., Inglaterra: JOHN WILEY & SONS, LTD, 2002.
- [6] B. Technologies, «10 Popular Software Defined Radios (SDRs) of 2019,» 31-diciembre- 2018. [En línea]. Available: <https://blog.bliley.com/10-popular-sdrs-software-defined-radios-2018>. [Último acceso: 20- noviembre- 2019].
- [7] N. INSTRUMENTS, «GETTING START GUIDE NI USRP - 2920/2921/2922 Universal Software Radio Peripheral,» septiembre- 2015. [En línea]. Available: <http://www.ni.com/pdf/manuals/376358a.pdf>. [Último acceso: 20 octubre 2019].
- [8] N. INSTRUMENTS, «USRP - 2920 BLOCK DIAGRAM,» 27- junio- 2017. [En línea]. Available: <http://www.ni.com/documentation/en/usrp-software-defined-radio-device/latest/usrp-2920/block-diagram/>. [Último acceso: 20- septiembre- 2019].
- [9] N. Instruments, «NI-USRP Configuration Utility».
- [10] Wayne Tomasi, Sistemas de Comunicaciones Electrónicas, Cuarta ed., Edo. de México: Pearson Education, 2003.
- [11] P. S. Foundation, 2001. [En línea]. Available: <https://docs.python.org/3/library/math.html>. [Último acceso: 16- agosto- 2019].

- [12] Facundo Q.. [En línea]. Available: http://facundoq.github.io/courses/images/res/03_numpy.html. [Último acceso: 16 agosto- 2019].
- [13] Ian . A. Glover, Peter. M. Grant, Digital Communications, vol. 3rd Edition, Edinburgh Gate: Pearson Education, 2010.
- [14] Oscar Mauricio Reyes. Homero Ortega Boada, Comunicaciones Digitales basadas en Radio Definida por Software, Bucaramanga, Colombia: Editorial UIS, abril, 2016.
- [15] JOHN. B. ANDERSON, Digital Transmission Engineering, Canadá: IEEE Press, 2005.
- [16] B. SKLAR, DIGITAL COMMUNICATIONS Fundamentals and Applications, second edition ed., New Jersey: Prentice Hall PTR, 2001.
- [17] B. P. LATHI, «SISTEMAS DE COMUNICACIÓN,» California State University, Sacramento, McGRAW-HILL, 1986.
- [18] A. Miguel, 26 mayo 2014. [En línea]. Available: <https://repository.usta.edu.co/bitstream/handle/11634/2805/Avalomiguel-anex-2015.pdf?sequence=2&isAllowed=y>. [Último acceso: 25- julio- 2019].
- [19] G. Radio, «wiki.gnuradio,» [En línea]. Available: https://wiki.gnuradio.org/index.php/USRP_Source. [Último acceso: 10 julio 2019].
- [20] G. Radio, «wiki.gnuradio,» [En línea]. Available: https://wiki.gnuradio.org/index.php/USRP_Sink. [Último acceso: 10 julio 2019].
- [21] G. Radio, 2012. [En línea]. Available: https://www.gnuradio.org/doc/sphinx-3.6/gr/source_blk.html. [Último acceso: 12 julio 2019].
- [22] A. Mohar, 22 agosto 2015. [En línea]. Available: <http://www.andrej-mohar.com/resampling-and-throttling-in-gnuradio>. [Último acceso: 20 julio 2019].
- [23] X.-G. Xia, Modulated Coding for Intersymbol Interference Channel, Primera ed., London, New York: CRC Press, 2018.

ANEXOS

ORDEN DE EMPASTADO