PROYECTO FINAL COMUNICACIONES 2

VALIDACION DE UN SISTEMA DE COMUNICACIÓN DIGITAL

Diego Calderón – 2170478, Magda Cuadros - 2165546, Sebastián Guerrero– 2190435

Introducción:

En este proyecto se tiene como objetivo validar mediante la transmisión y recepción un sistema de comunicación digital que funciona con las modulaciones QPSK y GMSK. Para ello se cuenta como base el proyecto de grado “Diseño e implementación del prototipo de un sistema de comunicaciones satelital” [1]. El cual fue realizado con un radio HackRF. En este caso se usó el radio USRP2920 de National Instruments.

Se implementaron flujogramas desde GNU Radio [2] mediante enlace cableado y enlace inalámbrico.

Marco Teórico:

Es posible comparar dos equipos mediante características físicas básicas y es necesario conocer ciertos conceptos como: ganancia, ancho de banda, potencia, tasa de muestreo, para forjar un criterio que de preferencia en esta comparativa.

Ganancia: Expresa la magnitud entre dos señales (salida respecto a entrada) y está dada en dB.

Ancho de Banda: Capacidad máxima y cantidad de datos que es posible transmitir en una conexión, diferencia de frecuencias (máxima menos mínima).

Potencia: Cantidad de energía por segundo de una señal, esta expresada en W, dB o dBm.

Tasa de Muestreo: Es la cantidad de muestras que se toma de una señal en una unidad de tiempo.

Teniendo claridad en dichos conceptos podemos representar estas características de ambos radios en una tabla que nos facilita su comparación, en la tabla 1 podemos observar lado a lado estos y más parámetros de los radios *Hackrf ONE* y *NI2920* respectivamente.

Comparativa Radios Definidos por Software		
Características	***HackRF ONE***	***USRP NI2920***
Rango de Frecuencias	1 MHz a 6 GHz	50 MHz a 2.2 GHz
Ancho de Banda	20 MHz	20 MHz
Duplexación	Half	Full
ADC	MAX5864 ADC 2 canales	ADC de 2 canales
	Resolución de 8 bits	Resolución de 16 bits
	/	100MS/s
DAC	MAX5864 DAC 2 canales	DAC de 2 canales
	Resolución 10 bits DAC	Resolución de 16 bits

	22MS/s	400MS/s
Potencia de Transmisión	+10 dBm	(50 MHz a 1.2 GHz) 17 dBm a 20 dBm
		(1.2 GHz a 2.2 GHz) 15 dBm a 18 dBm
Potencia de Recepción	Max hasta -5 dBm, segura hasta 10dBm	Max 0 dBm
Rango de Ganancia	RX 0dB a 62dB	RX 0dB a 31.5 dB
	TX 0dB a 47 dB	TX 0dB a 31dB
Tasa de muestreo máxima	20MS/s	25 MS/s con un muestreo de 16 bits
		50MS/s con un muestreo de 8 bits

Tabla 1. comparativa entre equipos de radio.

1. *Implementación*

En este proyecto se hace la modulación tanto en GMSK y QPSK, sin embargo, la generación del mensaje y su respectiva codificación es realizada para ambas modulaciones, con lo que cuentan con bloques principales que permiten generar el mensaje, codificar y decodificar

1.0 Bloques de generación, codificación y decodificación.

1.0.1 Generación y codificación

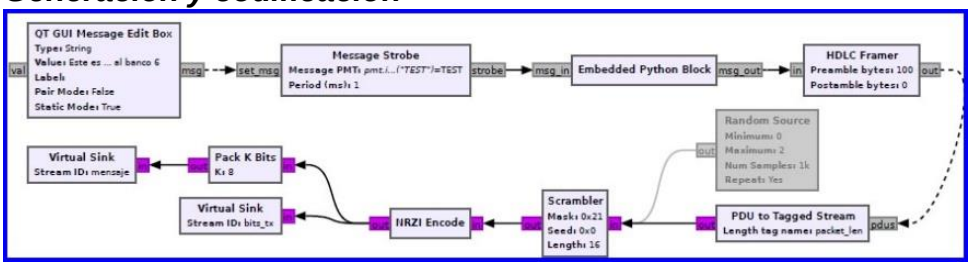


Imagen 1: Generación y Codificación del mensaje

Se lee en un cuadro de texto para que el usuario la ingrese (**Qt GUI message edit text box**). (**Message strope**) toma el mensaje y lo repite, cada 1 milisegundo. El bloque de python convierte el string en formato vector uint8 vectors, Luego **HDLC**[1] aplica un protocolo de enlace con el fin de evitar errores en la transmisión, es decir que agrega bits que permiten conocer y corregir errores, luego pasa de

[1] El bloque HDLC Framer y Deframer usado no es el que viene en GNU Radio, es la librería gr-satelites, específicamente el protocolo AX.25.

ser un mensaje a un flujo de datos **PDU to Tagged Stream** especificando el largo del paquete, **Scrambling** codificada de manera que aleatoriza el flujo de bits, luego existe una codificación que cambia la representación de bits; 1 son representados como cambios y 0 como mantenerse. Finalmente se empaquetan el flujo bits en flujo de bytes.

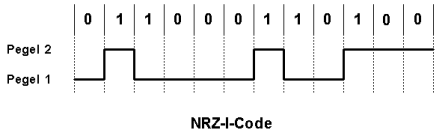


Imagen 2: Ejemplo de codificación NRZI

1.0.2 Decodificación

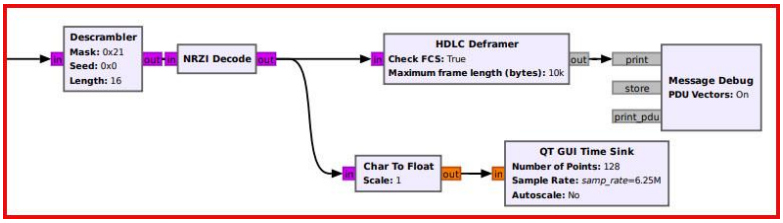


Imagen 3: Decodificación

Ya que el mensaje llega codificado, se tiene que hacer el proceso inverso para ver su contenido, primero se decodifica el **NRZI decoder**, cuando exista un flanco en la señal será un 1, cuando no exista será un 0 (detector de flancos ascendentes descendentes). Hay que mapear la trama de bits para obtener la original con **Descrambler**, luego el **HDLC Deframer** le quita los header y tails además de detectar errores, obteniendo a la salida solo los datos originales.

1.1 Simulación.

1.1.1 QPSK

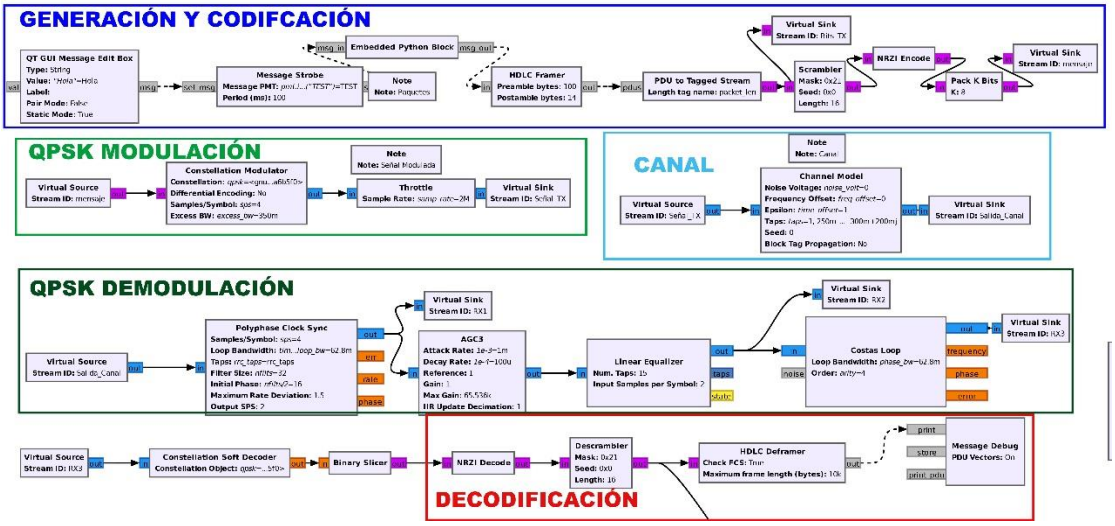


Imagen 4: QPSK Simulación

La simulación QPSk está compuesto por lazos de bloques distribuidos así:

1. GENERACIÓN Y CODIFICACIÓN: El primer bloque que se encuentra es el del editor de mensaje a enviar, seguido por el bloque que envía el mensaje en intervalos definidos. El bloque personalizado creado en Python encargado de tomar el mensaje y convertirlo en tipo unidad de datos de protocolo, por último, se realiza la codificación por protocolo HDLC (control de enlaces de alto nivel).[i]
2. MODULACIÓN QPSK: lazo compuesto por un bloque modulador de constelación que define tanto los puntos del diagrama de constelación como los símbolos que estos representan, seguido de un bloque regulador de flujo. [ii]
3. CANAL: simulación de canal inalámbrico, bloque de channel model permite evaluar, diseñar y testear el sistema en presencia de ruido blanco gaussiano, desviaciones de fase y frecuencia y no linealidades de canal.
4. DEMODULACIÓN QPSK: el primer bloque encontrado en este bloque es el polyphase clock sync el cual realiza sincronización de reloj, seguido del bloque AGC3 encargado del control automático, analizando la velocidad de actualización de muestreo dentro de un rango de tasa de ataque y una tasa de decaimiento. Este bloque tiene como parámetro el diezmado el cual determina el avance por número de muestras antes de calcular la actualización de ganancia de IIR. Utilizan los bloques de linear equalizer y costas loop, utilizados para muestrear la señal en su punto máximo y tomar así muestras donde se encuentra la mayor energía del símbolo.
5. DECODIFICACIÓN: primer bloque sin retorno a cero, utilizada para conversión de la señal binaria en señal digital el bloque descrambler, decodifica la señal de entrada utilizando un filtro de desplazamiento con retroalimentación lineal, seguido por un HDLC desframer que se encarga de tomar los bits desempaquetados y almacenarlos en texto y datos binarios.

1.1.2 GMSK

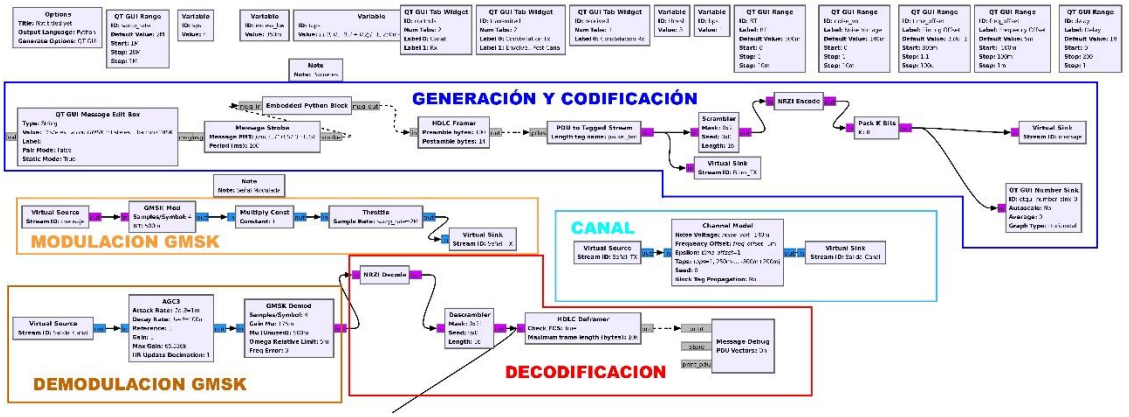


Imagen 5: GMSK Simulación

La simulación GMSK se divide en los mismos lazos de bloque de la simulación QPSK. El único bloque que presenta variaciones es el de modulación, compuesto así: Primero se tiene el bloque Mod GMSK el cual determina el bloque jerárquico para la modulación gaussiana, seguido por el bloque multiply const, el cual cumple con multiplicar la señal de entrada por un escalar y el throttle para regular el flujo de la señal para que no se exceda el sampt rate. el del GMSK demod, toma la envolvente compleja e implementa un muestreo cuando es necesario para la corrección y detección de errores en las tramas de datos transmitidos. [iii]

1.2 Implementación transmisión

1.2.1 Transmisión QPSK

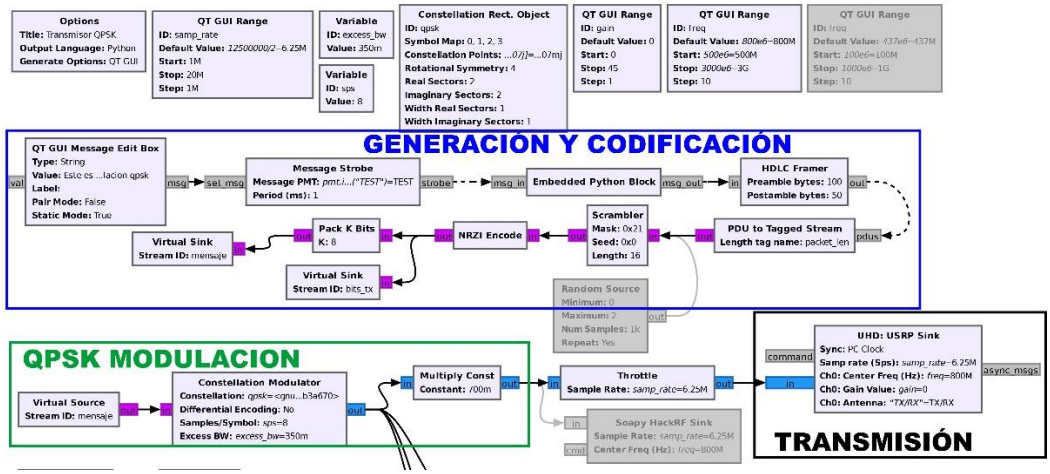


Imagen 6: QPSK Transmisión

Se genera y codifica el mensaje como se mostró en 1.0.1, El **Constellation Modulator** tiene la ubicación de cada uno de los cuatro puntos de la constelación, e internamente modula la señal por lo que pasa de ser un flujo bytes a número flotante que representa la señal. Se le multiplica una ganancia de 0.1 lo cual reduce su magnitud. Finalmente, la señal es transmitida de la

frecuencia de 800MHz que está en una frecuencia soportada por el Radio, por la antena, el osciloscopio y el analizador de espectros.

toca mirar hasta que frecuencia soporta el hackRFOne

1.2.2 Transmisión GMSK

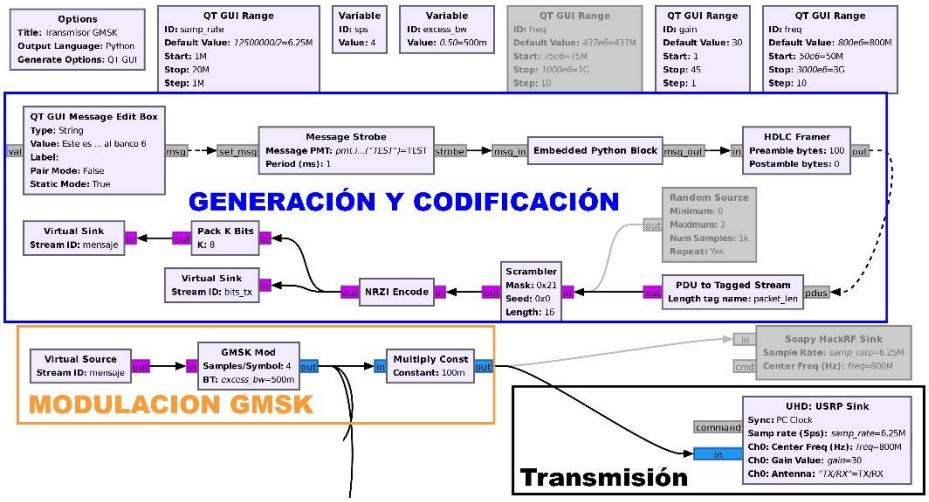


Imagen 7: GMSK Transmisión

Se genera y codifica el mensaje como se muestra en 1.0.1 . Para lo modulación se usó un bloque que ya viene con GNU radio que hace la nodulación **GMSK mod.** Se aplica la ganancia de 0.1 y finalmente se transmite al radio a la frecuencia de 800M, la cual es soportada por el Radio, la antena, el analizador de espectros y el osciloscopio.

1.3 Implementación recepción

1.3.1 Recepción QPSK

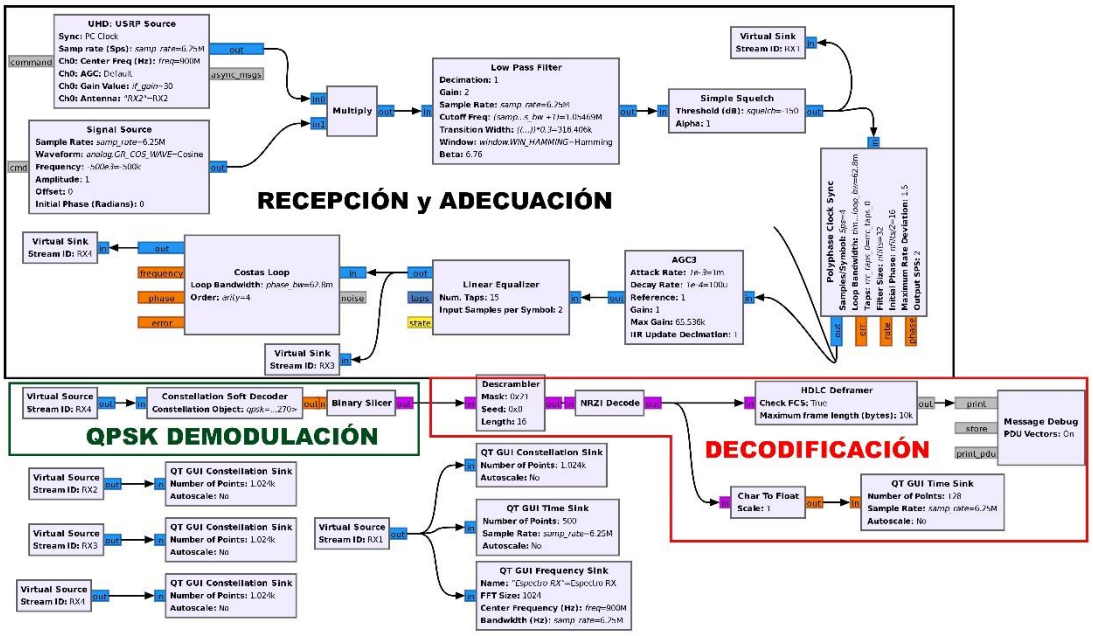


Imagen 8: QPSK Transmisión

En la recepción tenemos la señal captada por el radio, con el bloque ***UHD: USRP Source*** recibimos la señal, luego haciendo uso de ***Multiply***, multiplicamos la señal de entrada con un coseno dado por el bloque ***Signal Source***, de este modo se heterodina la señal y posteriormente pasa por ***Low Pass Filter*** donde nos quedamos con la señal de interés y quitando la parte DC, para realizar la demodulación, primero debemos sincronizar en fase nuestra señal, esto lo hacemos mediante el bloque ***Polyphase Clock Sync,*** procedemos con ***AGC3*** y ***Linear Equalizer*** se evitan los efectos de canal, por esto mismo fijamos la constelación y aproximamos los puntos haciendo uso del bloque ***Costas Loop***, en este momento si demodulamos la señal y lo siguiente a realizar es decodificar nuestro mensaje, tomamos los bits desempaquetados con ***Descrambler*** y decodificamos con ***NRZI Decode***, generamos el mensaje con ***HDLC Deframer*** y lo visualizamos en la terminal con ***Message Debug***, adicionalmente durante todo el proceso podemos visualizar las constelaciones con ***QT GUI Constellation Sink*** y analizar cómo cambian los puntos, así como también observamos el espectro en tiempo y frecuencia de la señal recibida con los bloques ***QT GUI Time Sink*** y ***QT GUI Frequency Sink*** respectivamente.

1.3.2 Recepción GMSK

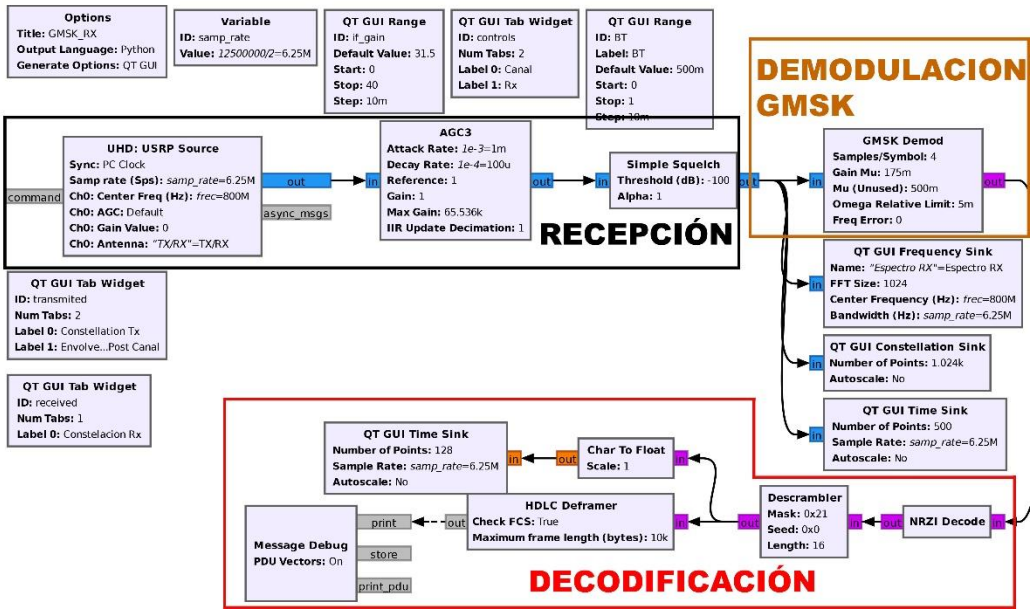


Imagen 9: Recepción GMSK

Se realiza la recepción de la señal con el radio, en el diagrama de bloques los encargados de realizar dicha recepción son ***USRP SOURCE***, ***AGC3*** y ***Simple Squelch***, demodulamos la señal usando el ***GMSK Demod*** y así recuperamos los bits transmitidos, visualizamos el espectro en frecuencia, en tiempo y también observamos el diagrama de constelación con los bloques ***QT GUI Frequency Sink***, ***QT GUI Time Sink*** y ***QT GUI Constelation Sink*** respectivamente, luego sigue la etapa de decodificación explicada en **1.0.2**, esta etapa nos entrega el mensaje que podemos visualizar a través del bloque ***Message Debug***.

2. Validación de sistemas

Para hacer la validación de las modulaciones se hizo de dos formas, la primera enviando un mensaje y la segunda enviando una señal aleatoria. Se observo la señal en el analizador de espectros y en el osciloscopio.

2.1 Enlace cableado

2.1.1 QPSK con mensaje

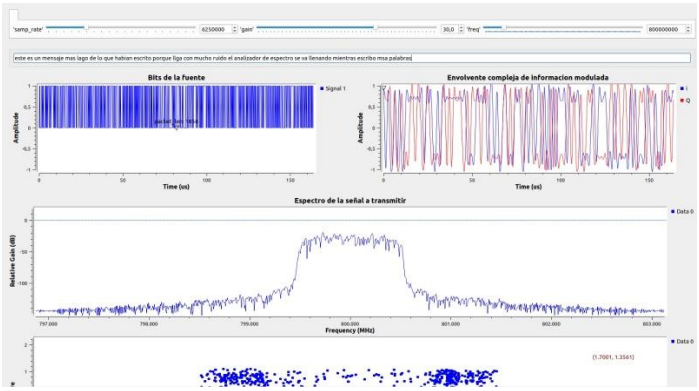


Imagen 10

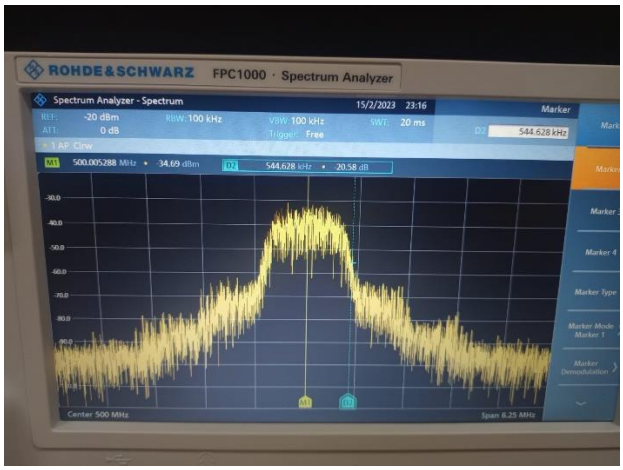


Imagen 11

Al comparar la señal de salida Imagen 11, vemos que se corresponde con la que estamos transmitiendo Imagen 10, con una buena señal a ruido. Teniendo en cuenta que el ancho de banda para la modulación QPSK [3]es:

$$BW_{3dB} = \frac{0.44}{T_B}$$

Y el ancho de de banda a 3dB nos dio $BW \approx 480kHz$
El tiempo del bit estaría dado por:

$$T_B = \frac{0.44}{480kHz} = 916.66nS$$

Al compararlo con el tiempo del Bit teórico:

$$T_B = \frac{sps}{f_{samp}} = \frac{6}{6.25MHz} = 960nS$$

Tenemos un error de:

$$Error = \frac{|960 - 916|}{960} = 4\%$$

2.1.2 QPSK Aleatorio

Si bien el scrambling hace aleatoria la trama de bits, ahora se usa ahora una señal de naturaleza completamente aleatoria.

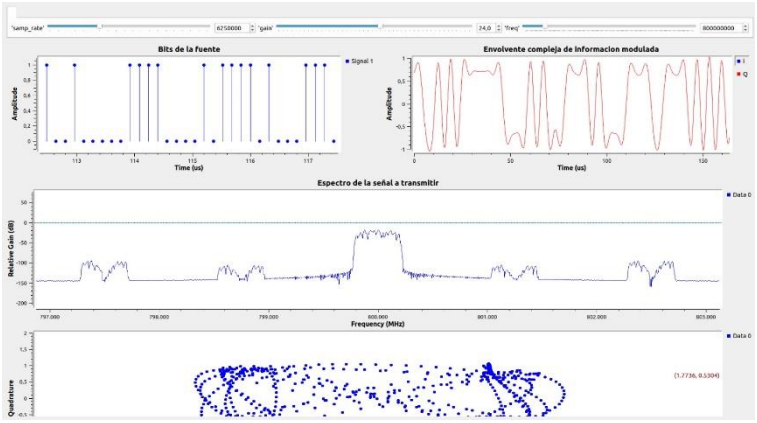


Imagen 12

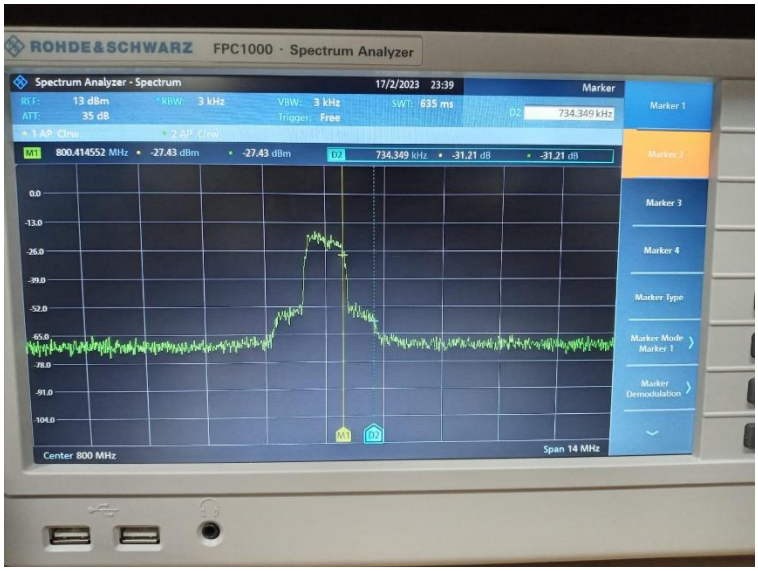


Imagen 13

Es una señal similar a la que se obtuvo anteriormente con el mensaje, al aplicar la misma ecuación para hallar tiempo de bit:

$$T_B = \frac{0.44}{414kHz} = 1062.8nS$$

Es decir que tiene un Error de:

$$Error = \frac{|960 - 1062|}{960} = 10\%$$

2.1.3 GMSK con mensaje

Se envió un mensaje con la modulación GMSK y se conectó al analizador de espectros por cable para ver su comportamiento. Lo que más evidente es que la señal no es un flujo constante de bits, ya que la señal aparece y desaparece de manera constante y repetitiva, por lo que se hace necesario usar el maxhold como modo de trazado.

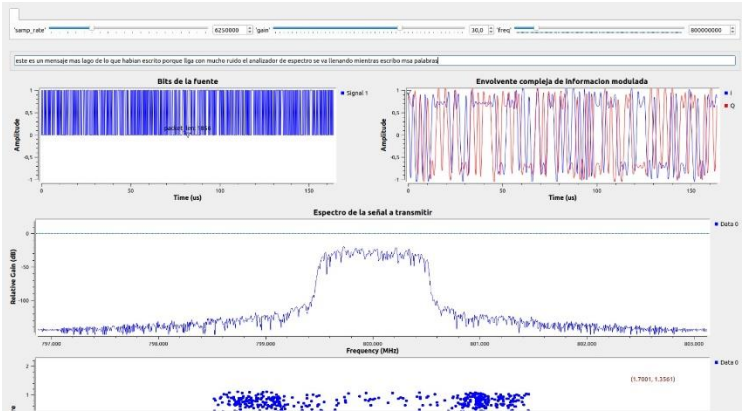


Imagen 14

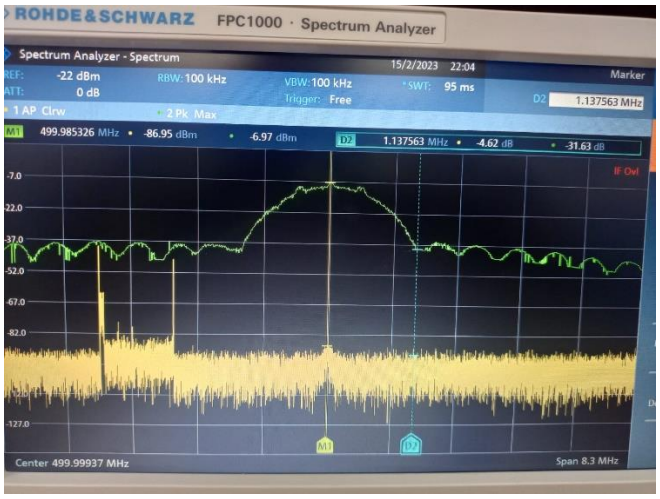


Imagen 15

Para esta modulación se usó un sps=4 y dio un ancho de banda de $Bw \approx 1.137\ \text{MHz}$
de esta modulación, comparada con la de 480Khz de la QPSK, se ve una reducción considerable en ancho de banda.

2.1.4 GMSK Aleatorio

Como la trama aleatoria se envía todo el tiempo, en este caso no fue necesario usar el maxhold para visualizar el espectro, ya que todo el tiempo aparece.

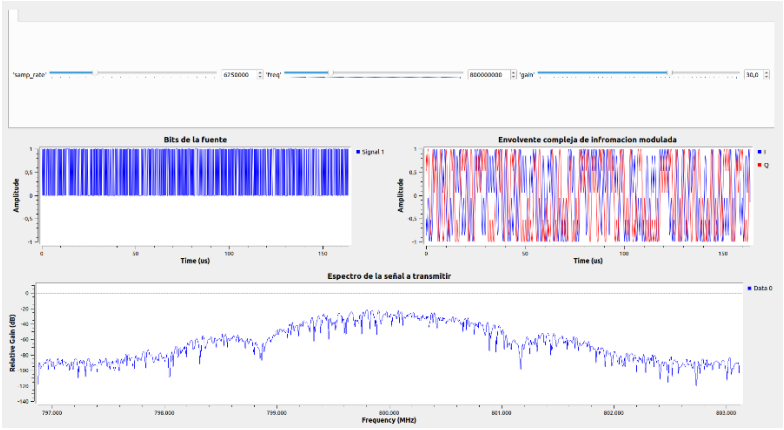


Imagen 16: Espectro enviado con modulación GMSK con bits aleatorios

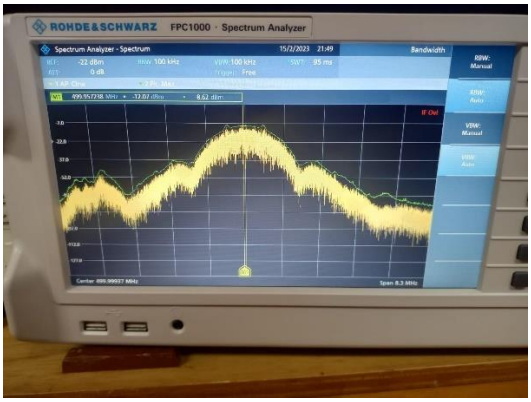


Imagen 17: Espectro recibido con modulación GMSK con bits aleatorios

En este caso el ancho de banda dio $\mathrm{BW} \approx 1.097\ \mathrm{MHz}$ que es similar al que se obtuvo con el mensaje, con lo que podemos verificar que medir el ancho de banda usando maxhold no afecto la medida.

2.2 Enlace inalámbrico

Para validar el enlace inalámbrico se uso la antena **Ref.:** ANTENA-900MHZ-5DBI *. La cual opera entre 824MHz y 900MHz. Se conecto el sistema de transmisión al sistema de recepción de otro computador del laboratorio de manera inalámbrica. La modulación GMSK se envió de forma satisfactoria pero la QPSK no se logro transmitir.*

2.2.1 GMSK

Para esta modulación se tuvo en cuenta la frecuencia de transmisión y los valores de ganancia son importantes, ya que, al ser una transmisión inalámbrica, se tiene un alto ruido y alta atenuación.

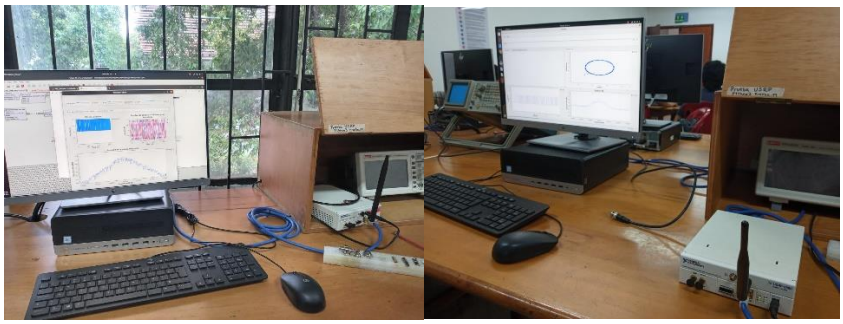


Imagen 18: Radio de transmisión (Izquierda) y Radio de recepción (Derecha)

Se hicieron dos pruebas, primero se envió un mensaje corto que dice "GSMK" y no se logró transmitir correctamente el mensaje ya que llegaba una trama de bits diferente, al cambiar el mensaje por uno más largo se logró transmitir como se muestra en la imagen.

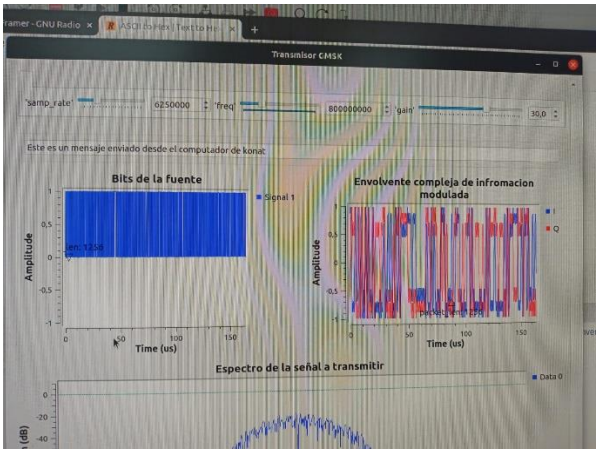


Imagen 19: Mensaje transmitido

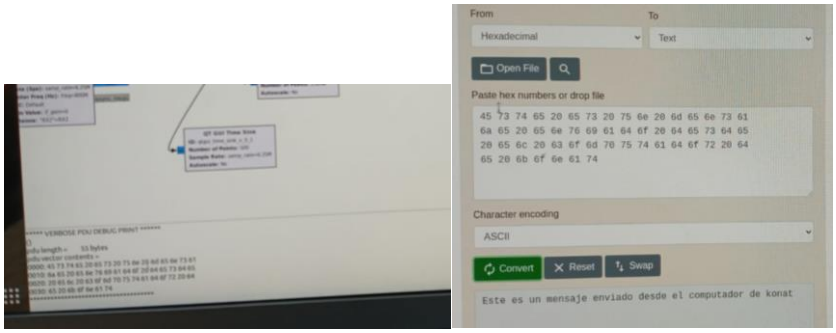


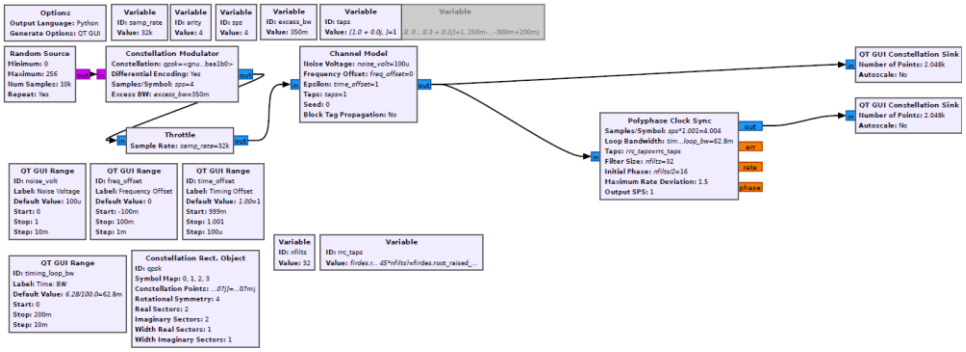
Imagen 20: Mensaje recibido por GMSK

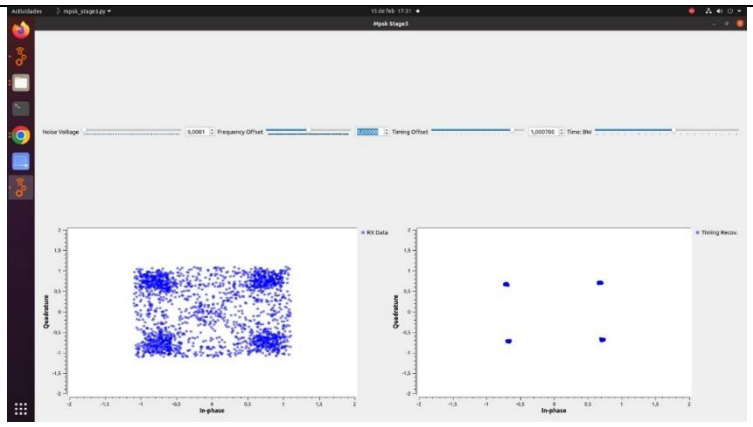
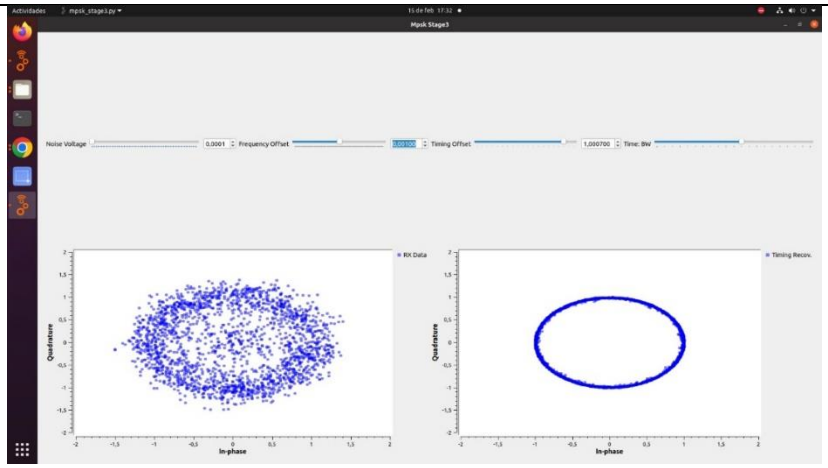
Es importante aclarar que no todos los mensajes llegaban bien, pero si la mayoría de las veces llega de manera correcta. Esto podría deberse a errores en la transmisión, se propone estudiar esto y

*compararlo con la BER asociada a la transmisión. Hay que tener en cuenta que el **HDLC Deframer** ya arregla varios de estos errores.*

3. Sincronización del reloj

Polyphase clock sync.



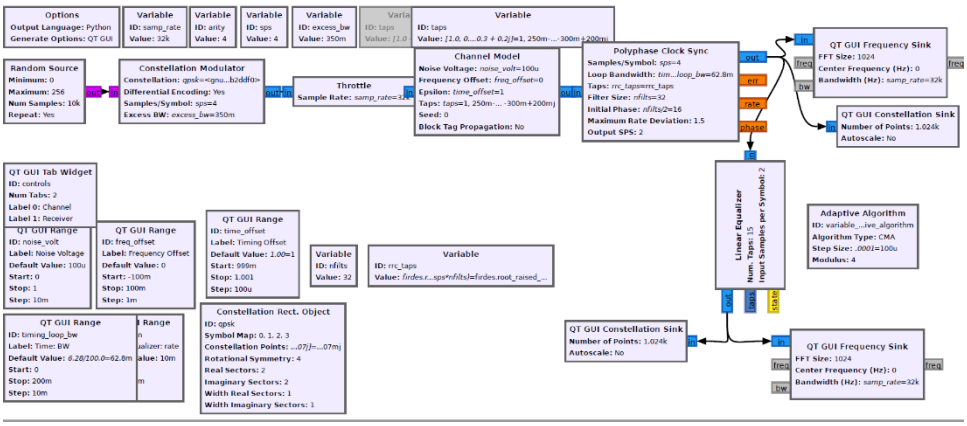
Sin desviación de frecuencia.	
Con desviación de frecuencia	

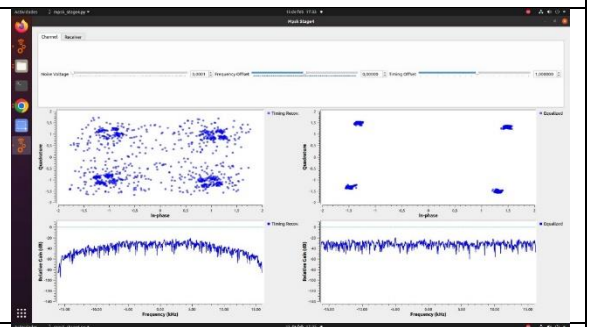
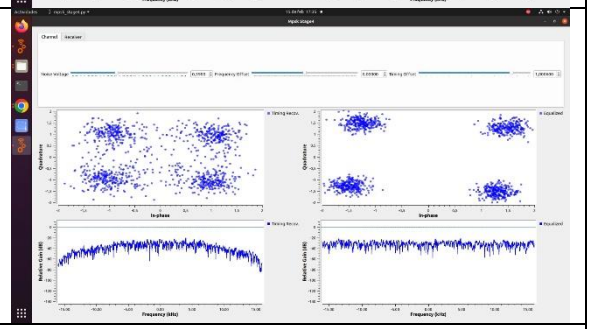
Realiza la sincronización de temporización de señales PAM minimizando la derivada de la señal filtrada, y a su vez maximiza la SNR, también minimiza interferencias. Los parámetros estipulados para que el bloque funcione correctamente son: ***muestra por símbolo*** (debe ser real positivo),

ancho de banda de lazo: representa el tamaño de paso del lazo de control, establece ganancia del lazo de control interno. (debe ser un número pequeño), ***taps***: valores de cada muestra correspondiente al filtro***, filter size***: número de filtros que tendrá el banco de filtros. ***Initial pase***: se sugiere que sea la mitad del número de filtros. ***Max rate deviation***: tasa de oscilación por defecto 1.5. Corrige los efectos debido al aumento de ruido en el canal, pero no las desviaciones a las que se ve afectado el espectro.

4. Distorsión de la señal

4.1 Linear Equalizer

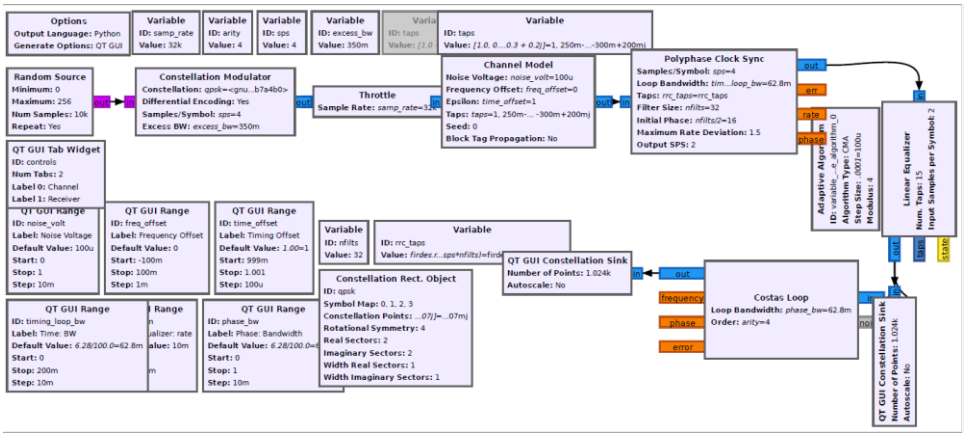


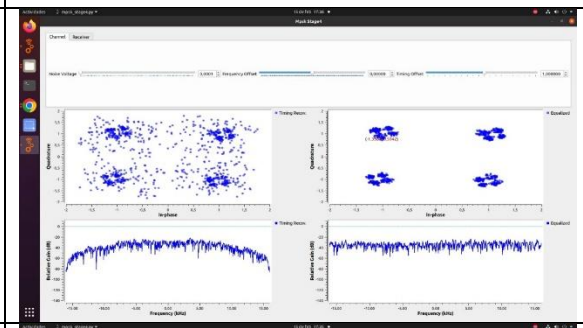
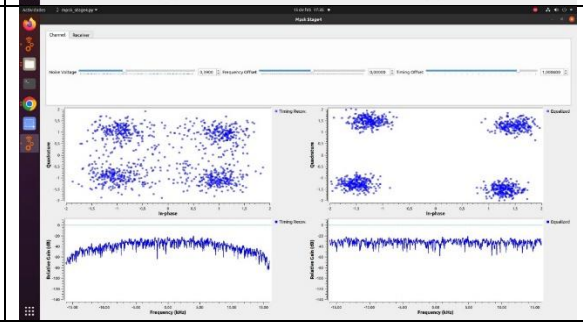
Variación de parámetros	*simulación*
Ruido=0	
Ruido=0.39	

El bloque linear equalizer realiza la ecualización lineal en un flujo de muestras compuestas, entre sus parámetros destacados para el filtro es el ***número de taps*** (separador de entrada de datos con los datos de la salida); también ***Sps***: se deben definir el número de muestras por símbolo del flujo de datos de entrada. En la imagen vemos como agrupa datos dispersos en el diagrama de constelación en puntos más específicos. Al agregar ruido es capaz de mantener los

datos más juntos. Es más robusto ante el ruido, pues esto hace que el espectro de la señal muestreada no se distorcione tan fácil.

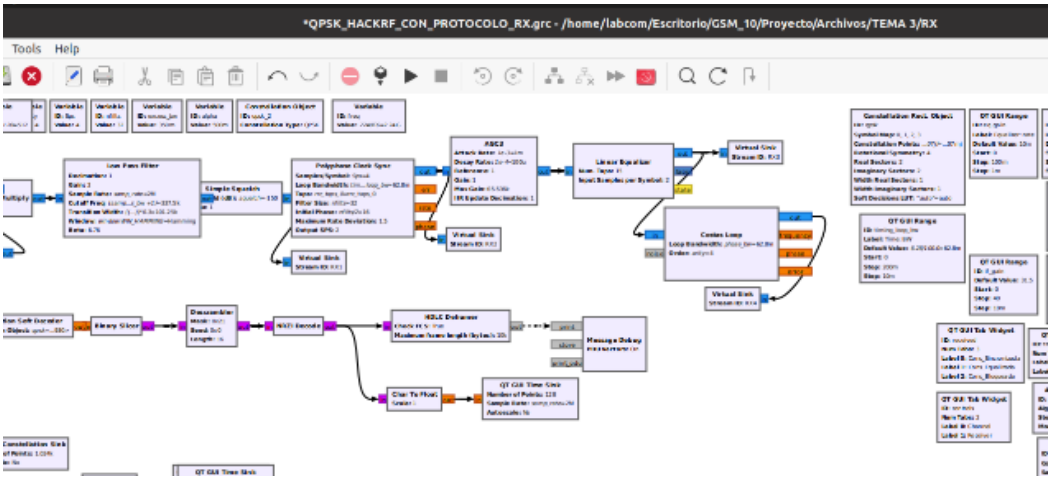
4.2 Adaptive Algoritmo

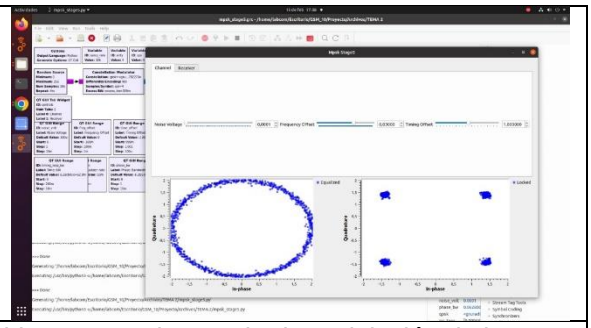


Variación de parámetros	
Ruido=0	
Ruido= 0.39	

Algoritmo de igualación ciega, este, no necesita una señal de referencia o conocer la señal transmitida para poder operar, su funcionalidad es fijar coeficientes de un filtro digital de FIR, generando control automático de ganancia. Se deben definir parámetros como: ***tipo de algoritmo*** en donde se especifica que algoritmo adaptativo se utilizará; ***Objeto de constelación digital***: modulación utilizada para adaptar la señal; ***Tamaño de paso***: velocidad donde convergerá el algoritmo. EL valor óptimo de este parámetro depende de la señal de entrada***;*** ***Modulo***: número de puntos que tiene el diagrama de constelación. En la imagen vemos como a pesar de tener ruido puede mantener el diagrama de constelaciones más definido.

5. Desplazamiento en fase y frecuencia: bloque costas loop





El bloque costas loop se encuentra en el lazo de bloques usado para la demodulación de la señal, realiza variaciones de fase y bloquea puntos de la constelación (para mantener posición estable) para recuperar la señal portadora.
Es utilizado para eliminar las desviaciones de frecuencia y fase en el diagrama de constelación. En la simulación se puede observar que el filtro permite observar los puntos correspondientes a la modulación ejecutada. Cuando no se ha puesto el costas loop, el sistema es sensible a desviaciones de frecuencia (izquierda).

Referencias

[1] A. F. P. Rueda, Diseño e implementación del prototipo de un sistema de comunicaciones satelital, Bucaramanga, 2022.

[2] D. Kozel, «GNU Radio,» [En línea]. Available: https://www.gnuradio.org/about/. [Último acceso: 18 Febrero 2023].

[3] E. Hernandez, «Sistemas de Telecomunicación. 5. Modulación digital.,» p. 8.

[i] tomado de “Diseño e implementación del prototipo de un sistema de comunicaciones satelital”. (Pág. 80).

[ii] tomado de “Diseño e implementación del prototipo de un sistema de comunicaciones satelital”. (Pág. 30).

[iii] tomado de “Diseño e implementación del prototipo de un sistema de comunicaciones satelital”. (Pág. 165).