

# Modulación Digital con SDR

Creación Colectiva

IPA2020

Riaño Walter, Barragan Wilmar, Montana Camilo, Giraldo Martin, Alfonso Andres, Hellcenk Murillo,  
Cifuentes Jorge, Lizcano Gonzalo, Marín Laura, Torres Omar, Guayara Camila, Firigua Lorena

Tutor

Rodríguez Mújica Leonardo

Facultad de Ingeniería  
Universidad de Cundinamarca

2020



# Contenidos generales

LABORATORIOS MODULACIONES DIGITALES

BIBLIOGRAFÍA



# LABORATORIOS MODULACIONES DIGITALES

# Parte IV - Tabla de contenidos

Lab17: BPSK	5
Lab18: Esquema de la Modulación y Demodulación DPSK en GNU Radio	12
Lab19: Modulación QPSK en GRC	24
Lab20: Modulacion digital QAM	34
Lab21: Filter taps	43
Lab22: Python block	52
Lab23: BER	59
Lab24: Constelación y FFT, Modulación BPSK	75
GENERADOR ALEATORIO DE PDU	86

# Lab17

## BPSK

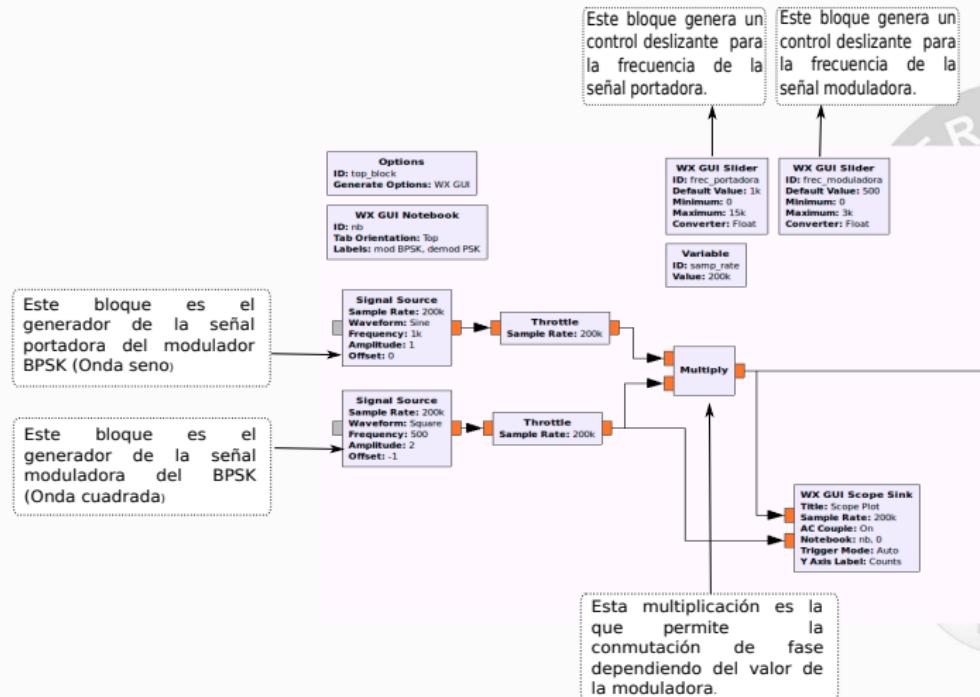
# Modulación BPSK

Para obtener una señal modulada se utiliza un modulador balanceado (Modulador de producto) que funciona como un conmutador, ya que a medida que la señal de información muestre un “1” o un “0” la portadora comuta en dos fases diferentes. En un modulador BPSK se asigna +1 V al 1 lógico y -1 V al 0 lógico, la portadora de entrada,  $\sin \omega_c t$  se multiplica por +1 o por -1. En consecuencia, la señal de salida puede ser  $+1 \sin \omega_c t$  o  $-1 \sin \omega_c t$ ; el primer producto representa una señal que está en fase con el oscilador de referencia, y el último producto, una señal que está desfasada 180 grados [11]. La ecuación de salida de un modulador BPSK es proporcional a:

$$y(t) = [\sin(2\pi f_a t)]X[\sin(2\pi f_c t)]$$

- ①  $f_a$ : frecuencia fundamental máxima de la entrada binaria (señal moduladora) en hertz.
- ②  $f_c$ : frecuencia de portadora de referencia en hertz.

# Modulador BPSK en GRC



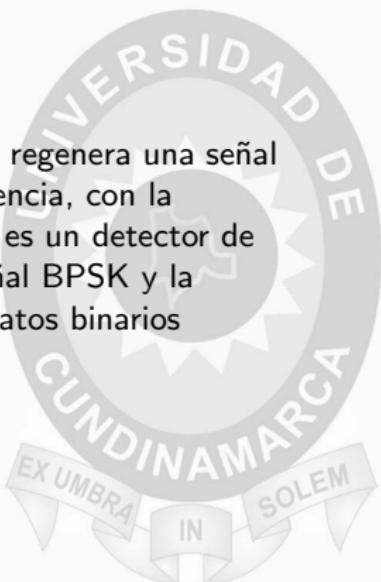
# Modulación BPSK



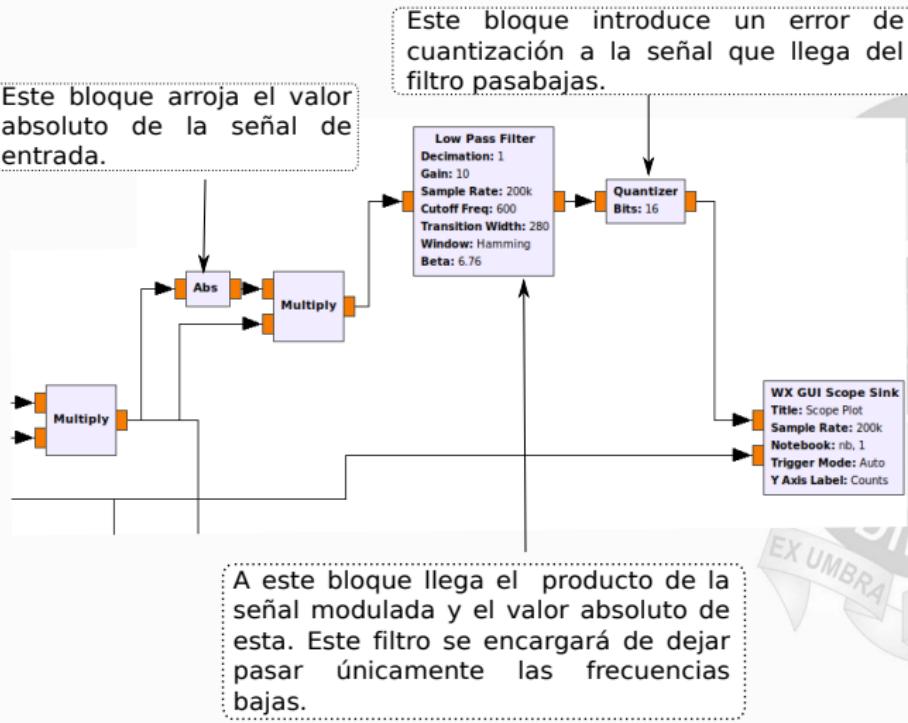
Cuando la señal de entrada digital cambia de estado, la fase de la portadora de salida varía entre dos ángulos que están desfasados 180°

# Demodulación BPSK

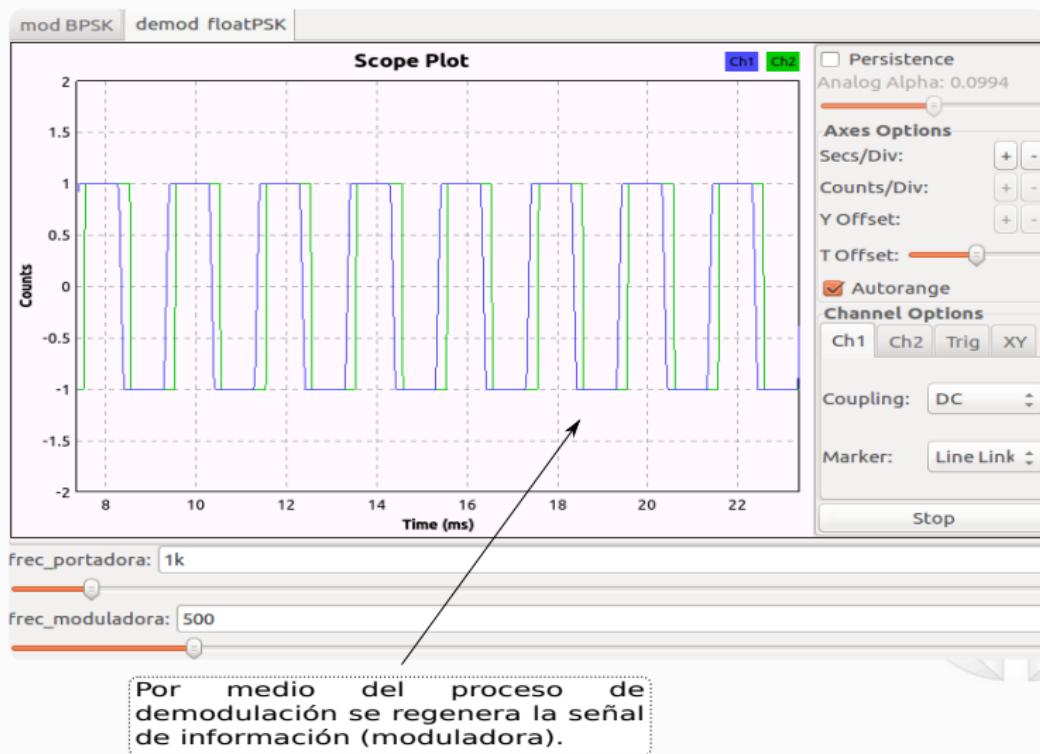
El circuito de recuperación coherente de portadora detecta y regenera una señal de portadora que es coherente, tanto en fase como en frecuencia, con la portadora original de transmisión. El modulador balanceado es un detector de producto; la salida es el producto de las dos entradas (la señal BPSK y la portadora recuperada). El filtro pasabajas (LPF) separa los datos binarios recuperados de la señal demodulada.[11]



# Demodulador BPSK en GRC



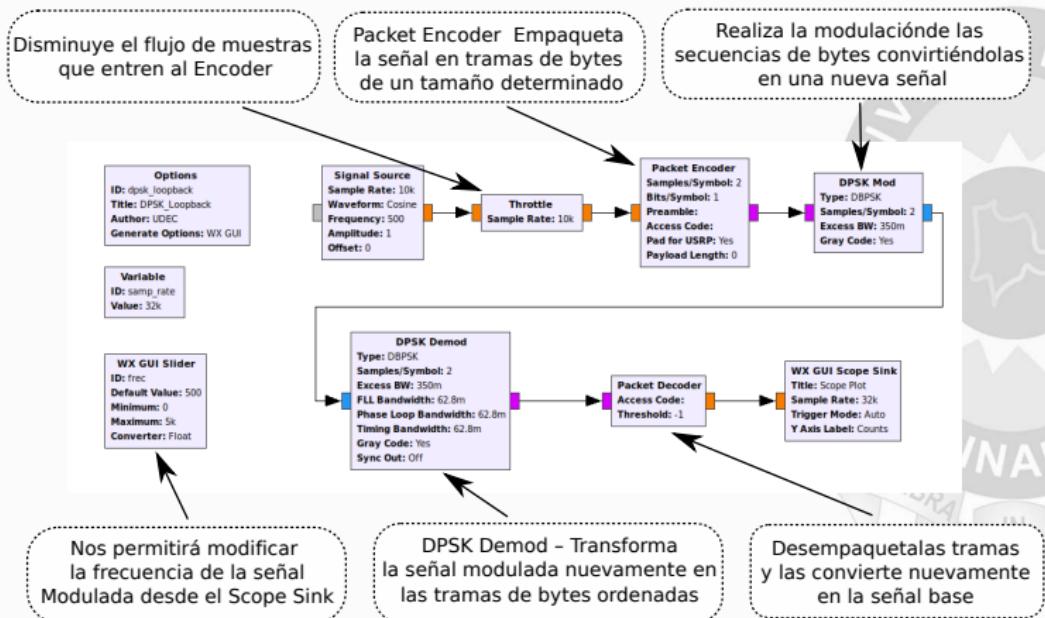
# Demodulación BPSK



# Lab 18:

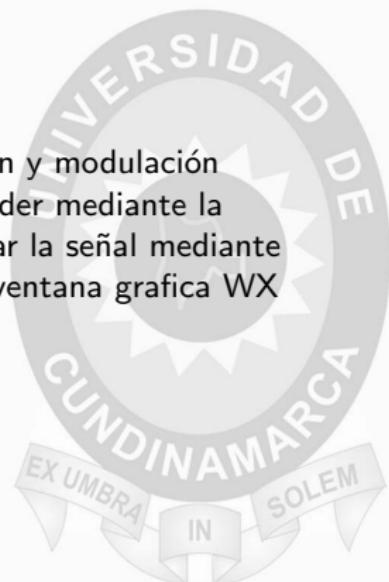
## Esquema de la Modulación y Demodulación DPSK en GNU Radio

# Esquema de la Modulación y Demodulación DPSK en GNU Radio



# Objetivo de la guía del ejemplo Modulación DPSK

Comprender el funcionamiento de los bloques de modulación y modulación DPSK, así como los bloques Packet Encoder y Packet decoder mediante la codificación y modulación de una señal, para luego recuperar la señal mediante la decodificación y modulación y finalmente ser vista en el ventana grafica WX GUI Scope Sink..

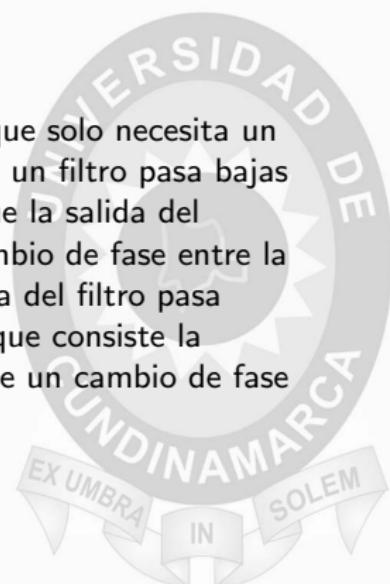


# Guía Ejemplo Modulación DPSK

1. Modulador DPSK: En un modulador Dpsk, se realiza una operación XNOR entre el bit actual de la señal digital de información y bit transmitido con anterioridad. La salida de esta operación entra en un modulador PSK. En el modulador PSK, siempre que la salida de la puerta XNOR aparezca un 1 lógico, el modulador producirá una salida analógica  $+ \cos(Wct)$ . Si en lugar de esto la salida de la XNOR es un 0 lógico, en el modulador aparecerá la señal de salida  $-\cos(Wct)$ . Es decir, la salida de XNOR va a cambiar de valor cada vez que aparezca un 0 lógico en la señal digital de entrada al modulador, y por consiguiente producirá un cambio de fase analógica de salida.

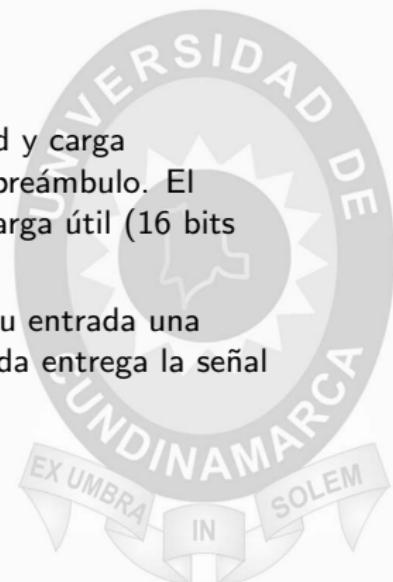
# Guía Ejemplo Modulación DPSK

1. Demodulador DPSK: Es un circuito bastante simple que solo necesita un multiplicador analógico, un latch de retraso de 1 bit y un filtro pasa bajas con frecuencia de corte menor que  $2Wc$ . Cada vez que la salida del multiplicador es  $Wct$ , es decir se ha producido un cambio de fase entre la señal de entrada y señal anterior retrasada , a la salida del filtro pasa bajas aparece un 0 lógico, que es precisamente en lo que consiste la modulación DPSK. En caso contrario, si no se produce un cambio de fase la salida del filtro es un 1 lógico.

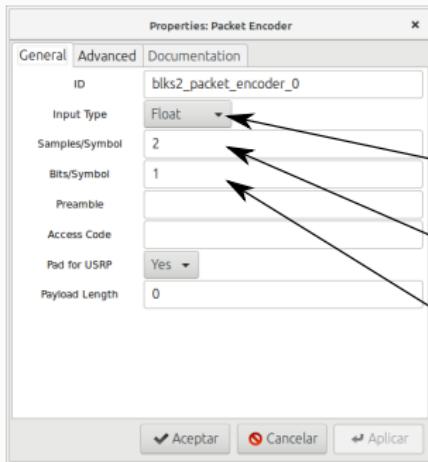


# Codificación y Modulación DPSK

1. Packet Encoder: Empaque los bytes de una longitud y carga determinada con un encabezado, código de acceso y preámbulo. El encabezado es una repetición 2x de la longitud de la carga útil (16 bits para cada campo).
2. DPSK Mod este bloque modulador DPSK recibe en su entrada una secuencia de bytes (carácter sin signo) y en su salida entrega la señal modulada en banda base.



# Packet Encoder y DPSK Mod



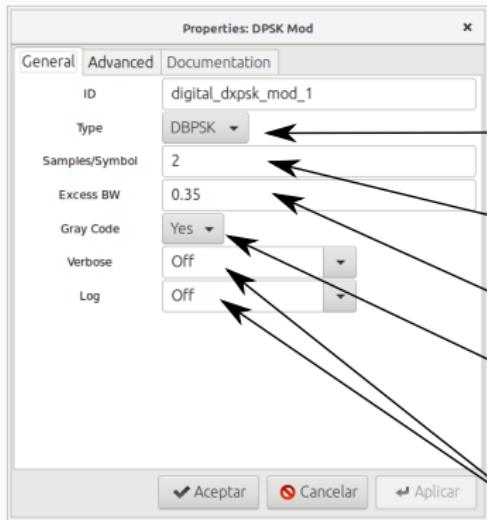
El Bloque Packet encoder se encarga de crear los paquetes de bytes de una longitud y carga determinada, se emplea junto con los bloques gmsk, dpsk, qam de GNU Radio

Tipo de entrada Flotante

El Período de reloj nominal especificado por el usuario en muestras por símbolo debe ser un valor entero superior a 2

Los bits / símbolo se deben configurar dbpsk -> 1 dqpsk -> 2

# Packet Encoder y DPSK Mod



DPSK Mod recibe en su entrada una secuencia de bytes (Carácter sin signo) en su salida entrega una señal compleja modulada en banda base.

Indicamos el tipo de modulación DPSK Mod opera para modulaciones del tipo DBPSK O DQPSK

las muestras por símbolo son las muestras por baudio que serán mayores a o iguales a 2

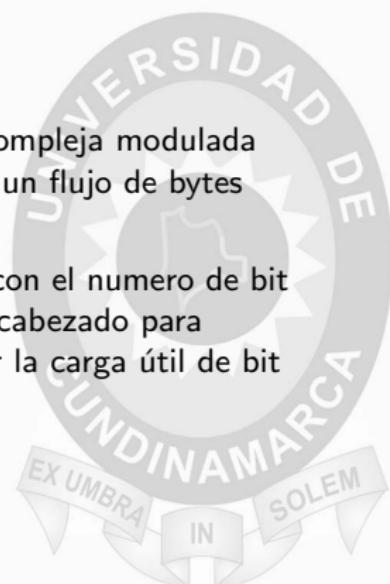
ancho de banda de exceso de filtro de coseno en raíz (flotante)

Se puede usar codificación o no cuando se habilita Gray Code

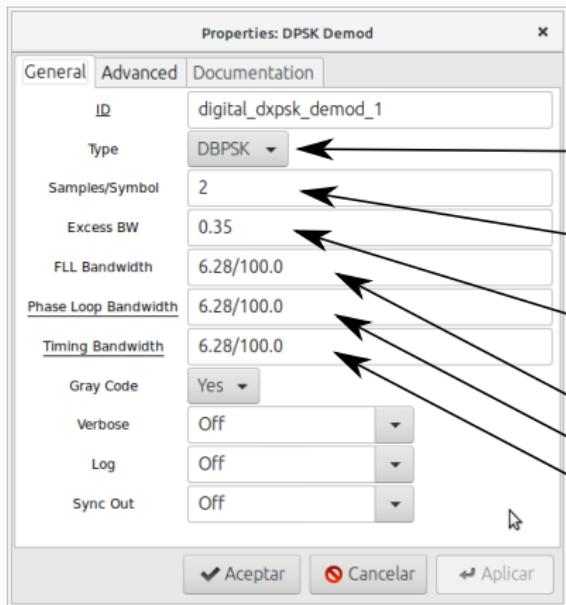
En caso de querer Registrar datos de modulación en archivos o Imprimir información sobre el modulador podemos habitar la opción las dos opciones

# Demodulación y Decodificación DPSK

1. DPSK Demod: Realiza la demodulación de la señal compleja modulada en la frecuencia de banda base y entrega en su salida un flujo de bytes empaquetados
2. Packet Decoder: El decodificador busca el código de con el numero de bit disponibles incorrecto, cuando lo encuentra , lee el encabezado para obtener la longitud de la carga útil extraerla y generar la carga útil de bit original.



# DPSK Demod y Packet Decoder



DPSK Mod recibe una señal compleja modulada y la convierte en un flujo de bit empaquetados

Indicamos el tipo de modulación DBPSK O DQPSK

las muestras por baudio serán mayores a o iguales a 2

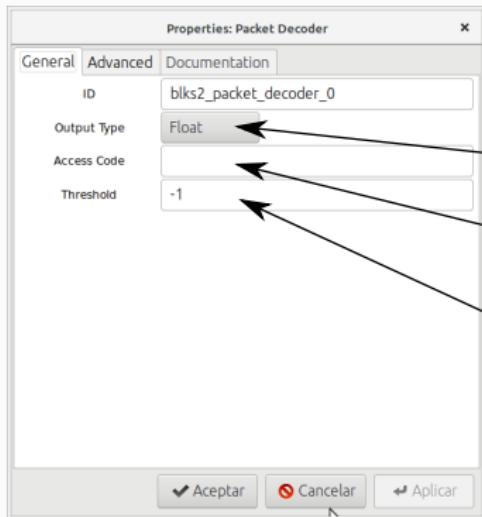
Exceso de ancho de banda

Filtro de bloqueo del ancho de banda

Ancho de banda de recuperación

Temporizador del Ancho de banda de bloqueo

# DPSK Demod y Packet Decoder



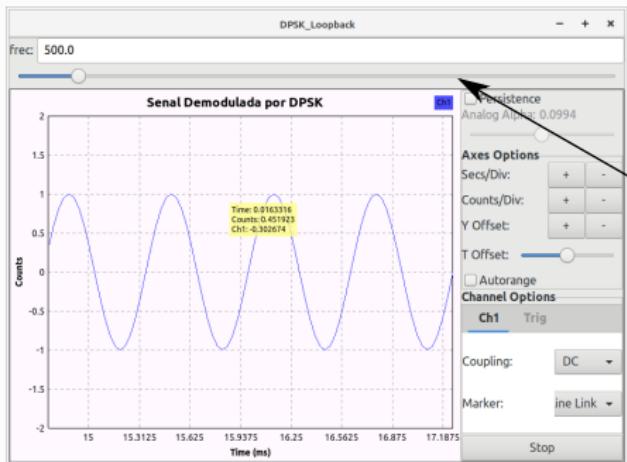
El Packet Decoder se encarga leer la longitud de la carga del paquete y generar la carga útil

Tipo de salida flotante

Sin un valor el código de acceso funciona como automático

para que el límite sea automático el valor debe ser -1

# Señal Demodulada DPSK



Este toolbox simula un osciloscopio, los parámetro de la frecuencia gracias al WX Gui Slider añadido, ademas podemos cambiar el rango de visión de la grafica desde los ejes.

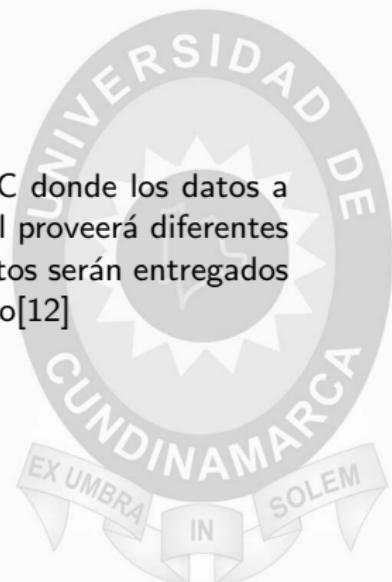
Cambio de la frecuencia Modulada

Tras el proceso de modulación y demodulación la señal se reconstruye teniendo sus características originales posteriores al ingreso del bloque Packet Encoder

# Lab 19: Modulación QPSK en GRC

# Modulación QPSK en GRC

Este ejercicio mostrará la modulación digital QPSK en GRC donde los datos a enviar serán generados por el bloque Random Source el cual proveerá diferentes valores de manera aleatoria en el rango de 0 a 100, estos datos serán entregados en forma de bytes, para poderlos visualizar en un osciloscopio[12]



# Modulación QPSK en GRC

**Options**  
ID: top\_block  
Generate Options: WX GUI

**Variable**  
ID: samp\_rate  
Value: 64k

**Variable**  
ID: sps  
Value: 4

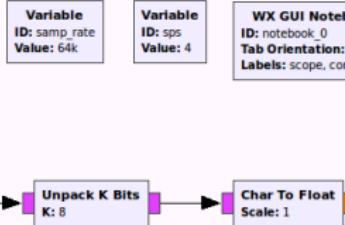
**WX GUI Notebook**  
ID: notebook\_0  
Tab Orientation: Top  
Labels: scope, const, freq

**WX GUI Slider**  
ID: muestreo  
Default Value: 64k  
Minimum: 64k  
Maximum: 1M  
Converter: Float

**Random Source**  
Minimum: 0  
Maximum: 100  
Num Samples: 100  
Repeat: Yes

**Constellation Object**  
ID: qpsk  
Symbol Map: 0, 1, 3, 2  
Constellation Points: ...1-1]  
Rotational Symmetry: 4  
Dimensionality: 1

**Constellation Modulator**  
Constellation: <>con... (m=4)>  
Differential Encoding: Yes  
Samples/Symbol: 4  
Excess BW: 1

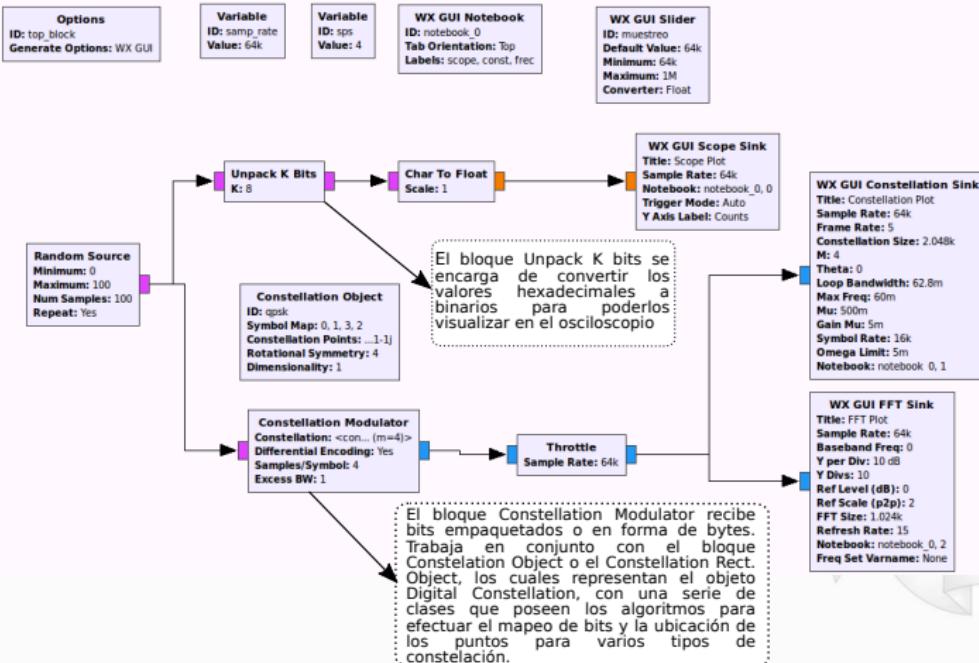


**WX GUI Scope Sink**  
Title: Scope Plot  
Sample Rate: 64k  
Notebook: notebook\_0, 0  
Trigger Mode: Auto  
Y Axis Label: Counts

**WX GUI Constellation Sink**  
Title: Constellation Plot  
Sample Rate: 64k  
Frame Rate: 5  
Constellation Size: 2.048k  
M: 4  
Theta: 0  
Loop Bandwidth: 62.8m  
Max Freq: 60m  
Mu: 500m  
Gain Mu: 5m  
Symbol Rate: 16k  
Omega Limit: 5m  
Notebook: notebook\_0, 1

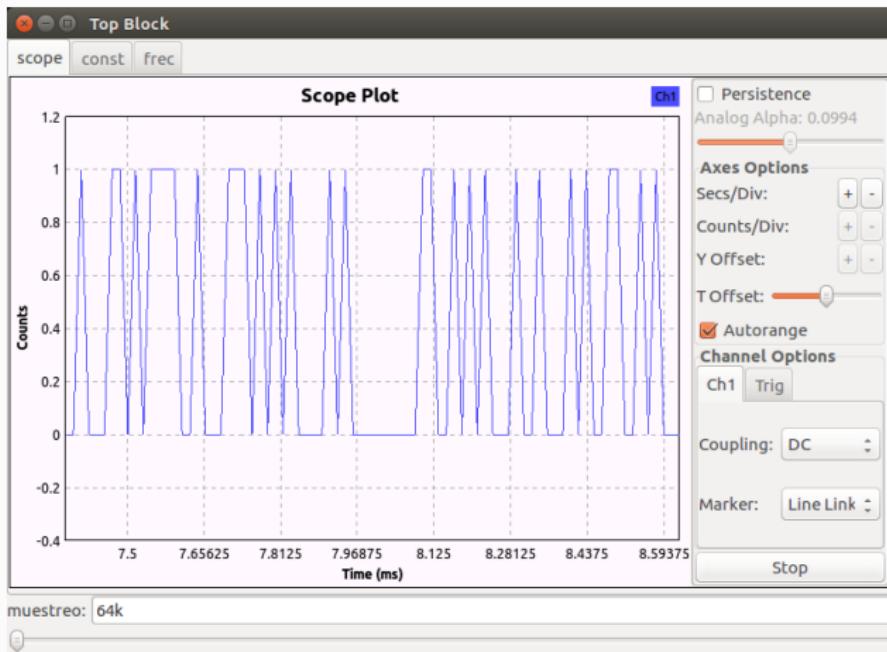
**WX GUI FFT Sink**  
Title: FFT Plot  
Sample Rate: 64k  
Baseband Freq: 0  
Y per Div: 10 dB  
Y Divs: 10  
Ref Level (dB): 0  
Ref Scale (p2p): 2  
FFT Size: 1.024k  
Refresh Rate: 15  
Notebook: notebook\_0, 2  
Freq Set Varname: None

# Modulación QPSK en GRC



# Modulación QPSK en GRC

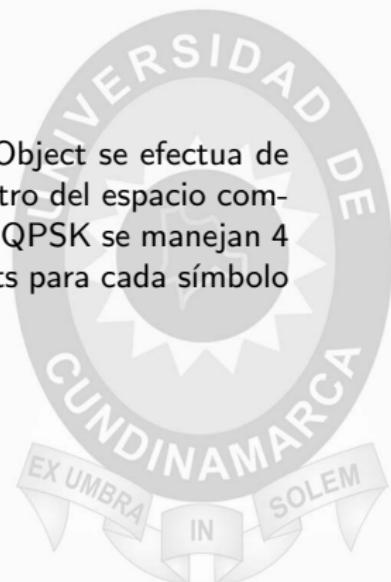
Pulsos de señal generados por Random Source



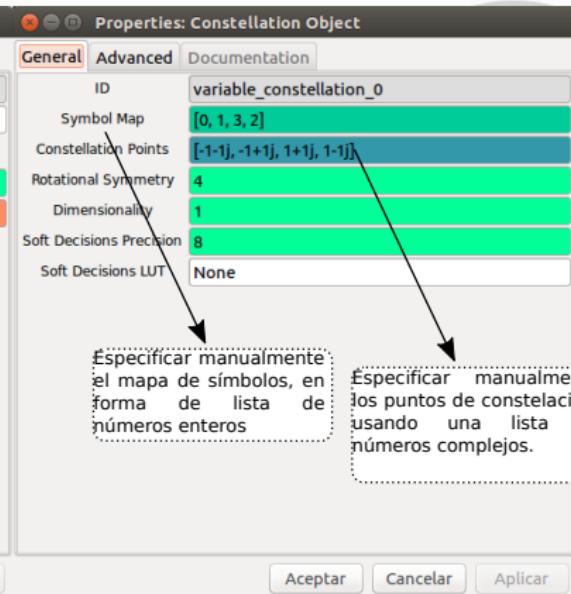
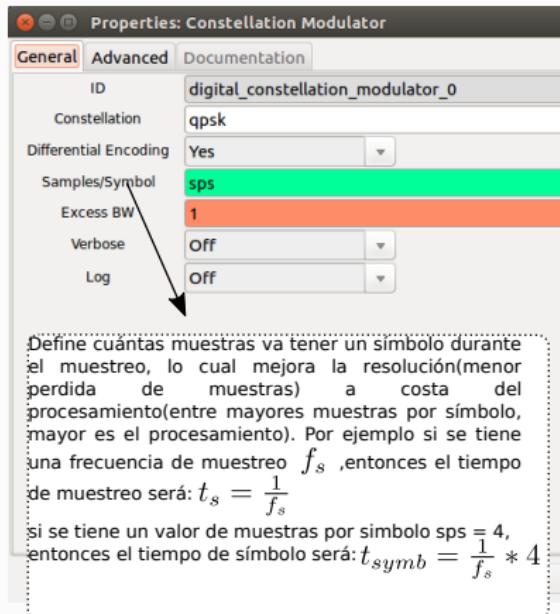
# Modulación QPSK en GRC

La definición de la constelación en el bloque Constellation Object se efectua de acuerdo a la cantidad de símbolos posibles en la misma dentro del espacio complejo o el diagrama de constelación. Para la constelación de QPSK se manejan 4 símbolos, entonces de acuerdo a la siguiente ecuación los bits para cada símbolo serían 2.

$$\log_2(4) = 2 \text{ bits/símbolo}$$



# Modulación QPSK en GRC



# Modulación QPSK en GRC

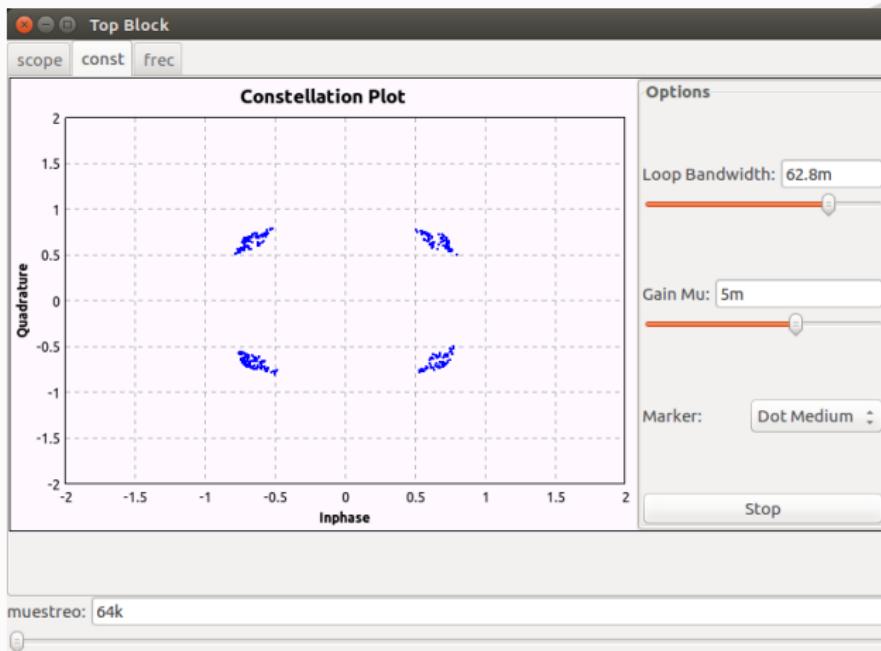
En el bloque Constellation modulator es posible implementar codificación diferencial con tan sólo habilitar este parámetro dentro de la configuración. La codificación diferencial o modulación diferencial en fase, no siempre asigna una fase determinada a cada símbolo (0 o 1), sino que de acuerdo a cada transición de bit, el modulador efectúa un cambio de fase de la siguiente manera:

- Si el siguiente símbolo es un 1 el cambio de fase es de  $270^\circ$
- Si el siguiente símbolo es un 1 el cambio de fase es de  $90^\circ$

La codificación diferencial contribuye a la reducción de errores y hace la modulación QPSK más robusta frente a la sensibilidad de ruido debido a los saltos de fase, puesto que no es necesario una demodulación síncrona con la portadora.[12]

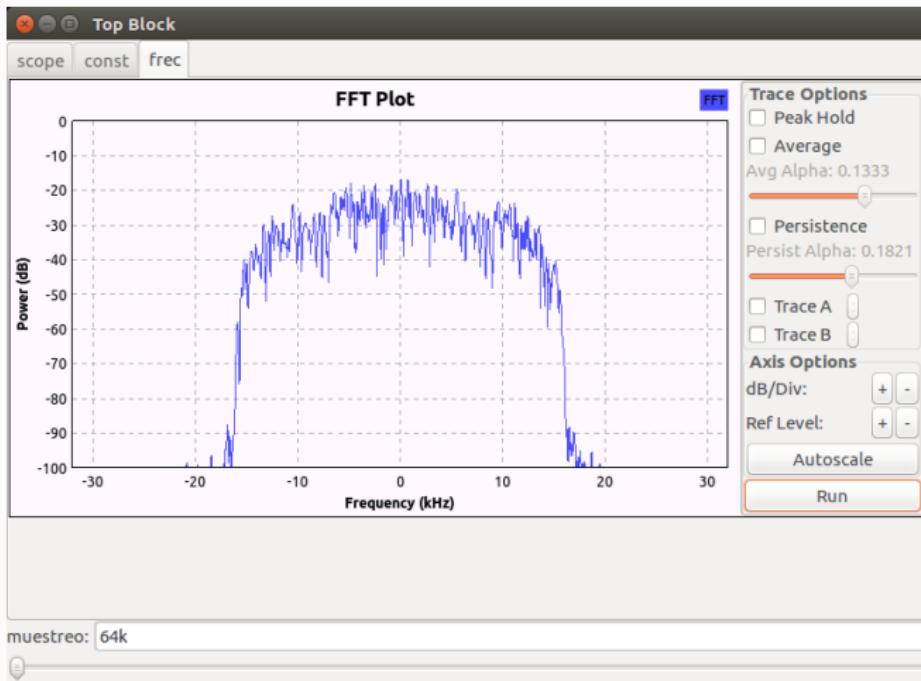
# Modulación QPSK en GRC

Constelación QPSK con codificación diferencial habilitada y 4 Muestras por símbolo



# Modulación QPSK en GRC

Espectro de señal QPSK modulada a una frecuencia de muestreo igual a 64 kHz



# Lab20

## Modulacion QAM

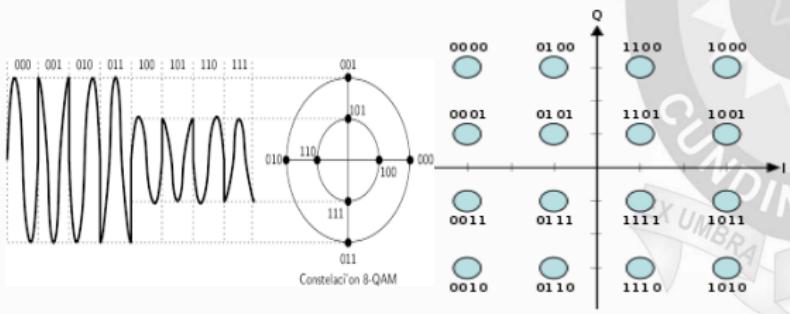
# Modulacion de amplitud en cuadratura QAM

La modulación de amplitud en cuadratura, en inglés Quadrature Amplitude Modulation (QAM), es una técnica de modulación digital avanzada que transporta datos, técnica en la cual la información va a ser modulada tanto en amplitud como en fase (la señal de portadora va a ser modificada en amplitud y fase) o sea que la información digital está contenida, tanto en la amplitud como en la fase de la portadora transmitida. Esto se consigue modulando una misma portadora, desfasando  $90^\circ$  la fase y la amplitud.

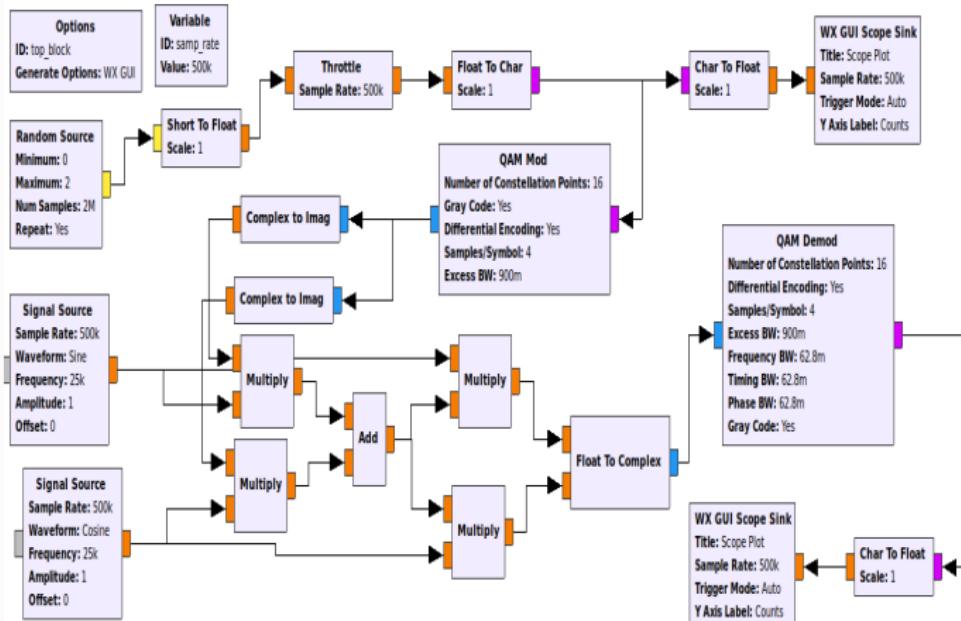
La modulación QAM, es una forma de modulación digital en donde la información digital está contenida, tanto en la amplitud como en la fase de la portadora transmitida.

# Modulacion de amplitud en cuadratura QAM

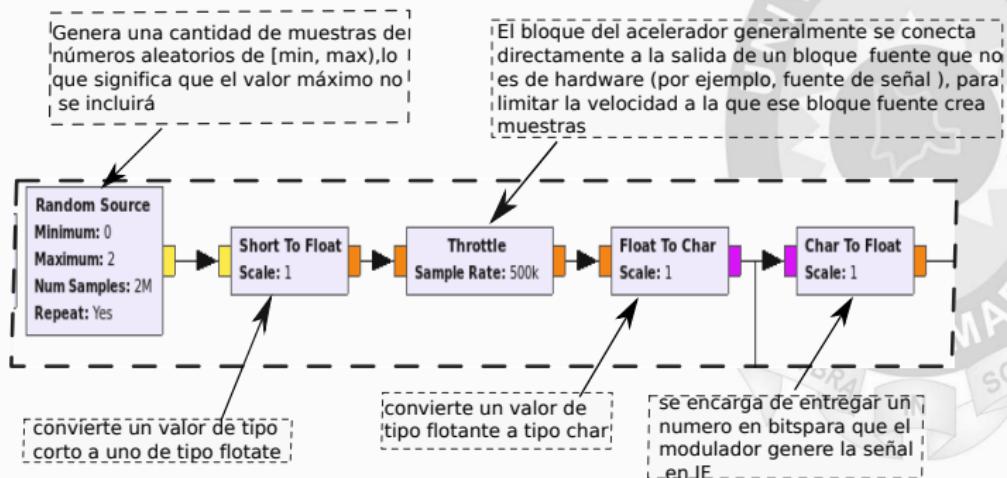
La modulación QAM consiste en modular por desplazamiento en amplitud (ASK) de forma independiente, dos señales portadoras que tienen la misma frecuencia pero que están desfasadas entre sí  $90^\circ$ . La señal modulada QAM es el resultado de sumar ambas señales ASK. Estas pueden operar por el mismo canal sin interferencia mutua porque sus portadoras están en cuadratura. La ecuación matemática de una señal modulada en QAM es:  
se encarga de entregar un numero en bits para que el modulador genere la señal en if  $a_n \cos(wt) + b_n \sin wt$



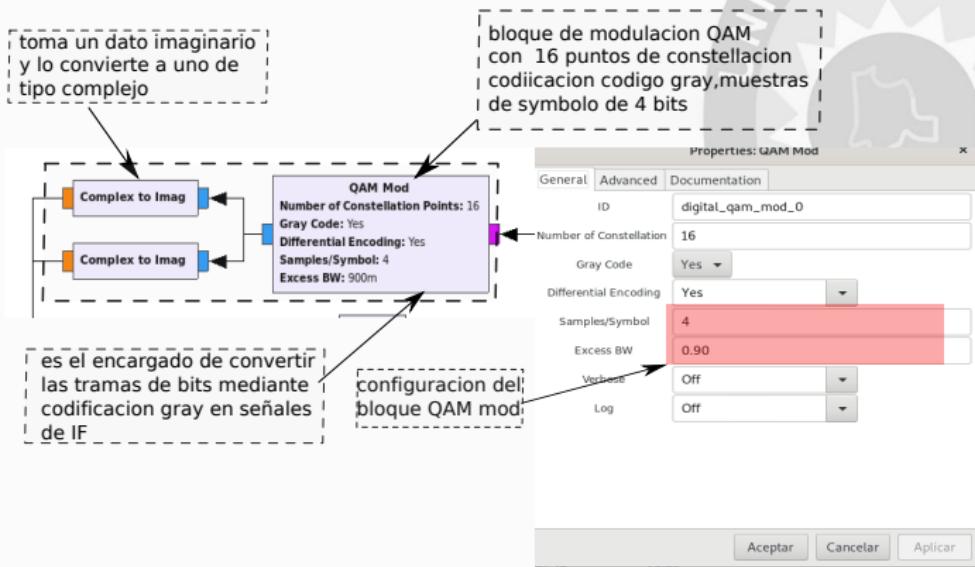
# Modulacion de amplitud en cuadratura QAM



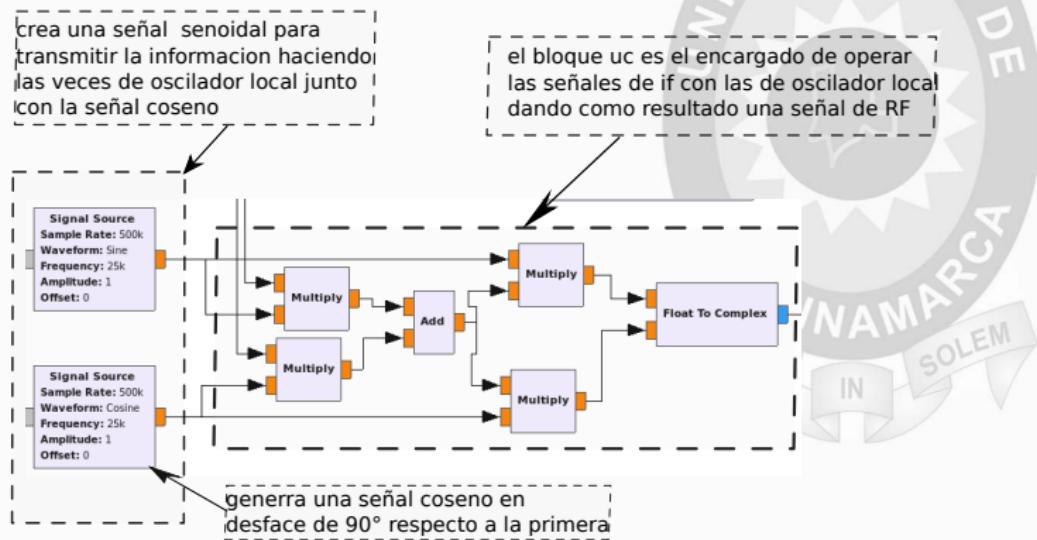
# Modulacion de amplitud en cuadratura QAM



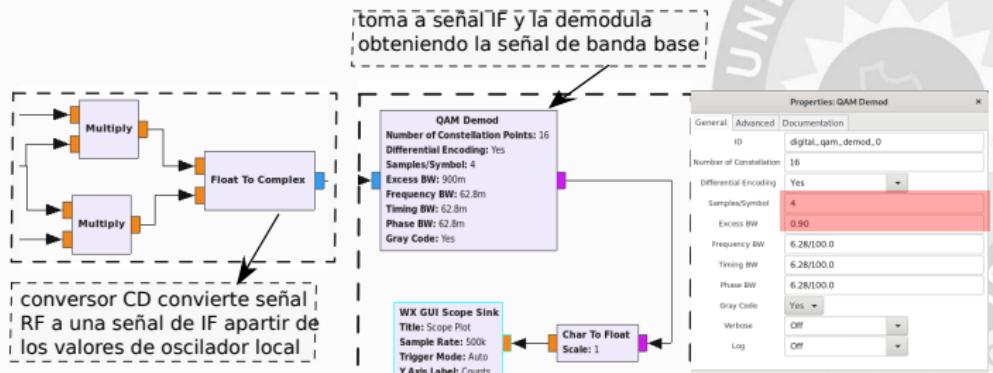
# Modulacion de amplitud en cuadratura QAM



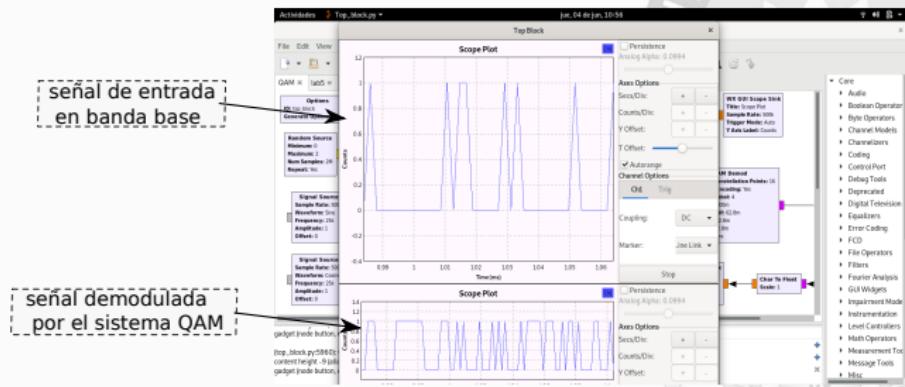
# Modulacion de amplitud en cuadratura QAM



# Modulacion de ampliacion en cuadratura QAM



# Modulacion de amplitud en cuadratura QAM



# Lab21

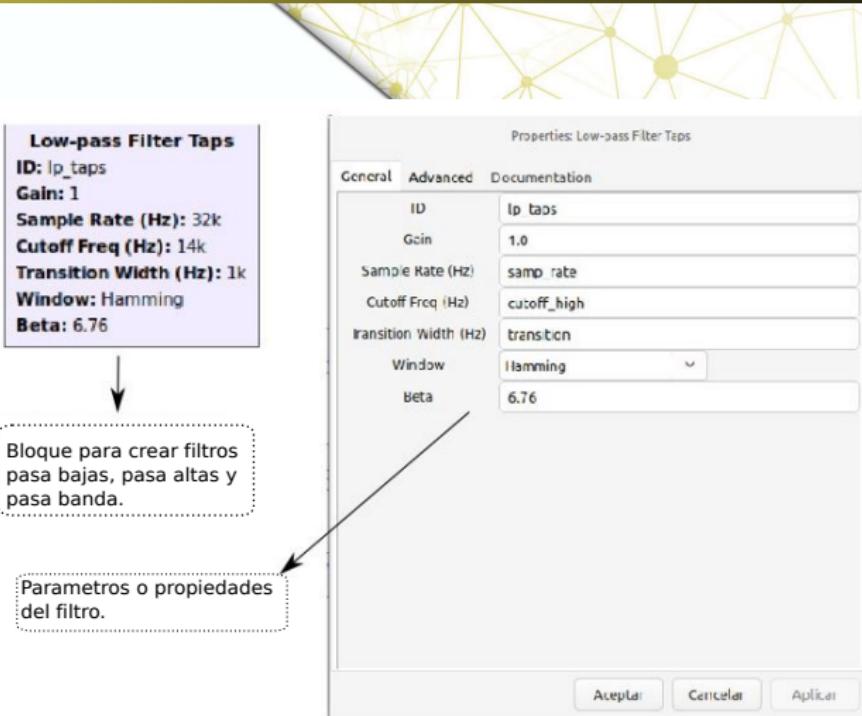
## Filter taps

# Filter taps

## Taps

El parámetro Taps indica las secciones de un filtro FIR que emula un retardo con múltiples trayectos. Uno de los parámetros más importantes para este bloque son los coeficientes (taps) del filtro. Uno de los beneficios de implementación del bloque es que usted puede definir el filtro acoplado directamente en este bloque, lo que permite tener el filtro acoplado y la corrección de tiempo de muestreo de manera simultánea.

# Filter taps



# Filter taps

## FIR Decimating

Este es un bloque que permite cargar toques de filtro desde un archivo (desde la herramienta de diseño de filtro). El nombre FIR viene de la manera en que el filtro afecta una señal. Una función impulso es una entrada bastante interesante, donde la señal es 0 siempre, excepto en un lugar donde tiene valor de 1.

# Filter taps

The screenshot shows a software interface for digital signal processing. On the left, there is a block labeled "Decimating FIR Filter" with the following parameters:

- Decimation:** 1
- Taps:** lp\_taps

Below this block is a label "Low-pass filter". A vertical arrow points downwards from the "lp\_taps" parameter towards a dashed-line box containing the text:

Bloque para llamar el filtro  
y decimar la señal.

To the right of the block is a "Properties" dialog box titled "Properties: Decimating FIR Filter". It has three tabs: General, Advanced, and Documentation. The General tab displays the following settings:

ID	lp_filter
Type	Float>Float (Real Taps)
Decimation	1
Taps	lp_taps
Sample Delay	0

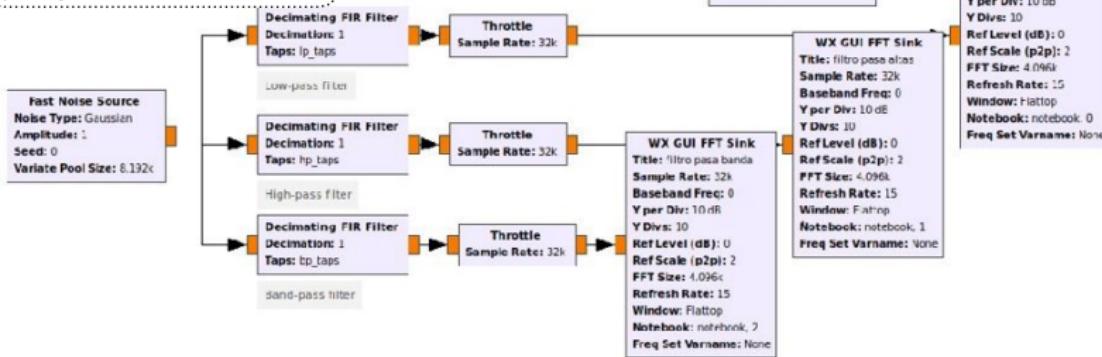
At the bottom of the dialog box are three buttons: "Aceptar" (Accept), "Cancelar" (Cancel), and "Aplicar" (Apply).

# Filter taps

Options		Variable	Variable	Variable	
ID:	filter_taps	ID:	samo_rate	ID:	bp_low
Generate Options:	WX GUI	Value:	32k	Value:	6k
Variable	ID: cutoff_high	Variable	ID: bp_high	Variable	ID: transition
Value:	14k	Value:	2k	Value:	10k
Variable	ID: cutoff_low	Variable	ID: bp_low	Variable	ID: transition
Value:	2k	Value:	6k	Value:	1k

<b>Low-pass Filter Taps</b> ID: lp_taps Gain: 1 Sample Rate (Hz): 32k Cutoff Freq (Hz): 14k Transition Width (Hz): 1k Window: Hamming Beta: 6./b	<b>High-pass Filter Taps</b> ID: hp_taps Gain: 1 Sample Rate (Hz): 32k Cutoff Freq (Hz): 2k Transition Width (Hz): 1k Window: Hamming Beta: 6./b	<b>Band-pass Filter Taps</b> ID: bp_taps Tap Type: Real Gain: 1 Sample Rate (Hz): 32k Low Cutoff Freq (Hz): 6k High Cutoff Freq (Hz): 10k Transition Width (Hz): 1k Window: Hamming Beta: 6.76	<b>WX GUI Notebook</b> ID: notebook Tab Orientation: Top Labels: lp, hp, bp
---	---	---	--

Se modifica de QT GUI a WX GUI, cambiando el visualizador de señal a FFT Sink.



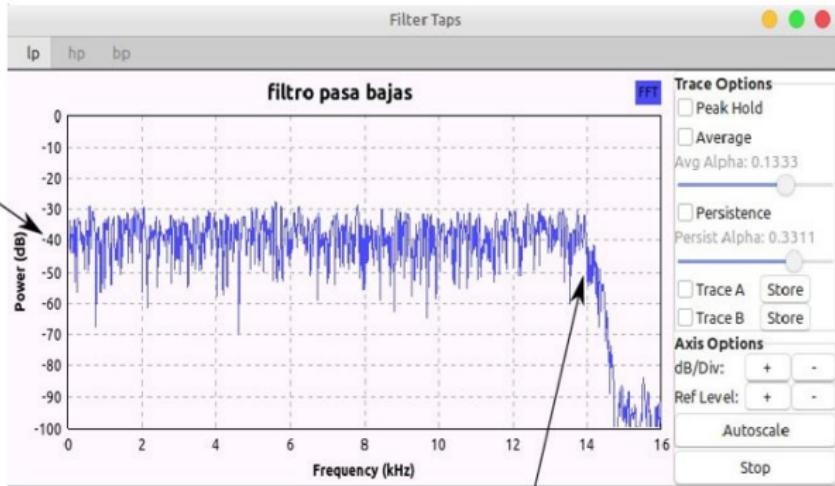
# Filter taps

Options		Variable	Variable
<b>ID:</b> filter_taps		<b>ID:</b> samp_rate	<b>ID:</b> bp_low
<b>Generate Options:</b> WX GUI		<b>Value:</b> 32k	<b>Value:</b> 6k
Variable	Variable	Variable	Variable
<b>ID:</b> cutoff_high	<b>ID:</b> cutoff_low	<b>ID:</b> bp_high	<b>ID:</b> transition
<b>Value:</b> 14k	<b>Value:</b> 2k	<b>Value:</b> 10k	<b>Value:</b> 1k

Se crean variables, " transición, frecuencias de corte, las cuales utilizaremos en el ajuste de los filtros.

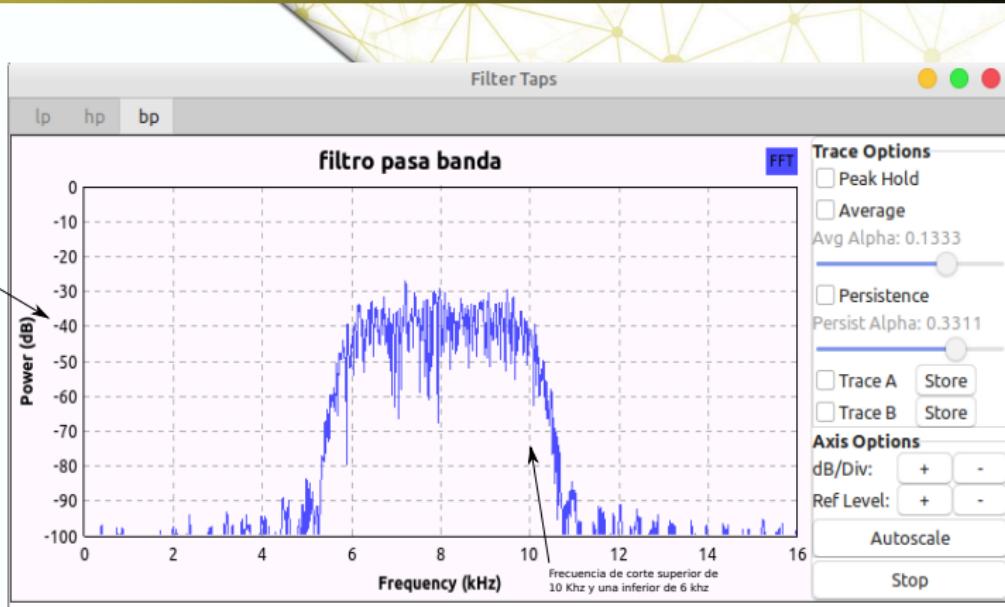
# Filter taps

Filtros pasa bajas a una amplitud de -40 dB.



Frecuencia de corte  
de 14 Khz.

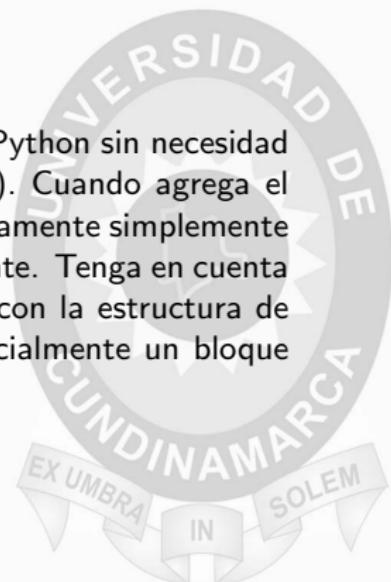
# Filter taps



# Actividad Embedded Python Block

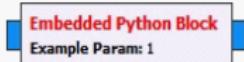
# Python block

- Le permite crear un nuevo bloque (personalizado) en Python sin necesidad de crear e instalar un Módulo fuera del árbol (OOT). Cuando agrega el bloque a su diagrama de flujo, el código rellenado previamente simplemente toma el flujo de entrada y lo multiplica por una constante. Tenga en cuenta que la estructura de este bloque de Python coincide con la estructura de un bloque de Python fuera del árbol (OOT). Es esencialmente un bloque Python OOT integrado en un diagrama de flujo grc.



# Python block

permite crear un nuevo bloque (personalizado)



al abrir el bloque no aparece la siguiente ventana

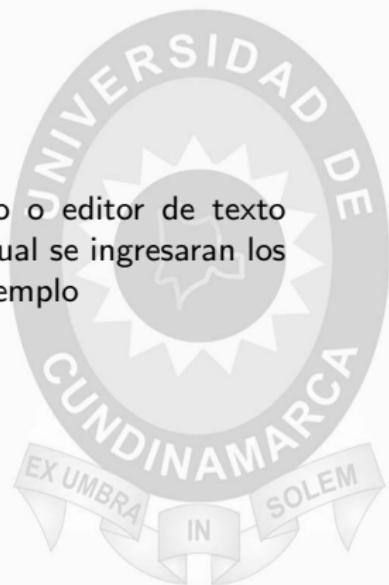


se mostrara el id (nombre de bloque)  
code donde al darle click se nos abria  
una ventana nos dara unas opciones a tomar  
example parametros se utiliza para  
multiplicar los valores ingresados

al dar click en open editor  
nos mostrara las opciones de  
escoger un editor de texto  
por defecto o alguno de preferencia

# Python block

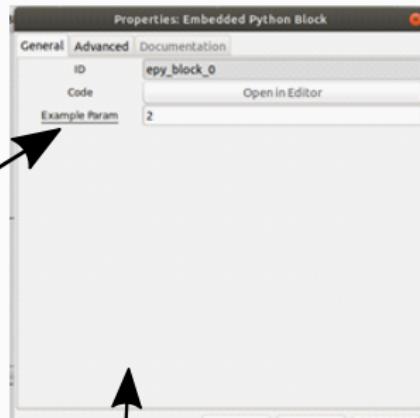
- Después de haber elegido nuestro editor por defecto o editor de texto preferente se nos genera un código del bloque. en el cual se ingresaran los parametros como se observara a continuacion en el ejemplo



# Python block

```
1  """
2  Embedded Python Blocks:
3
4  Each time this file is saved, GRC will instantiate the first class it finds
5  to get ports and parameters of your block. The arguments to __init__ will
6  be the parameters. All of them are required to have default values!
7  """
8  # SE IMPORTAN LAS LIBRERIAS PARA LA REALIZACION DEL PROCESO.
9  import numpy as np
10 from gnuradio import gr
11
12
13 class blk(gr.sync_block): # other base classes are basic_block, decim_block, interp_block
14     """Embedded Python Block example - a simple multiply const"""
15
16     def __init__(self, example_param=1.0): # only default arguments here
17         """arguments to this function show up as parameters in GRC"""
18         gr.sync_block.__init__(
19             self,
20             # EN ESTE APARTADO SE TIENEN LOS PARAMETROS DE LA CAJA PARA TRABAJAR
21             name='Embedded Python Block', # EN ESTA LINEA SE LA ASIGNA NOMBRE A LA CAJA
22             in_sig=[np.complex64], # EN ESTA LINEA SE OBTIENE LA SENAL DE ENTRADA
23             out_sig=[np.complex64] # EN ESTA LINEA SE OBTIENE LA SENAL DE SALIDA
24         )
25         # if an attribute with the same name as a parameter is found,
26         # a callback is registered (properties work, too).
27         self.example_param = example_param # EN ESTA LINEA SE OBTIENE EL PARAMETRO DE GANANCIA
28                                         # O CONSTANTE QUE SE MULTIPLICA POR LA SENAL DE ENTRADA
29
30     def work(self, input_items, output_items):
31         """AQUI SE REALIZA LA ACCION LA CAJA ES MULTIPLICAR LA SENAL DE ENTRADA POR LA CONSTANTE DADA POR LA CAJA"""
32         output_items[0][:] = input_items[0] * self.example_param
33         # SE RETORNA LA SENAL DE SALIDA.
34         return len(output_items[0])
35
```

# Python block

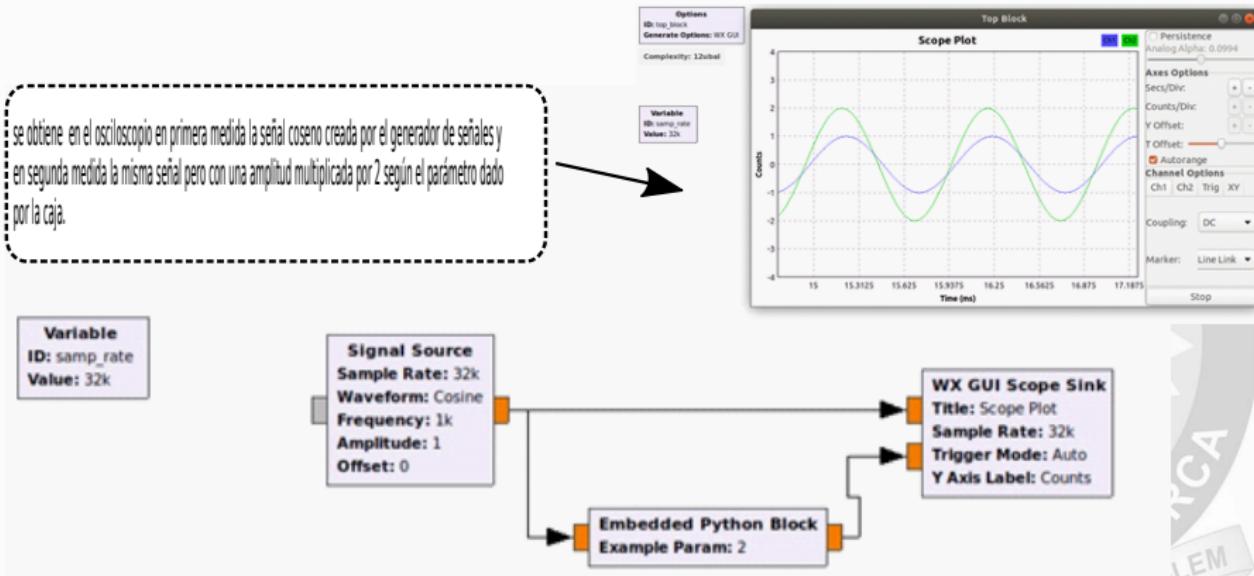


se cambia el parámetro de ganancia por 2  
como se ve en el diagrama

En el diagrama de bloques se abre la caja de Python Block y se modifica el **Example param** ya sea a con un numero entero o decimal, esto con el fin de multiplicar la señal de entrada por este parámetro para obtener una señal con mayor ganancia o menor dependiendo del usuario.

# Python block

se obtiene en el osciloscopio en primera medida la señal coseno creada por el generador de señales y en segunda medida la misma señal pero con una amplitud multiplicada por 2 según el parámetro dado por la caja.

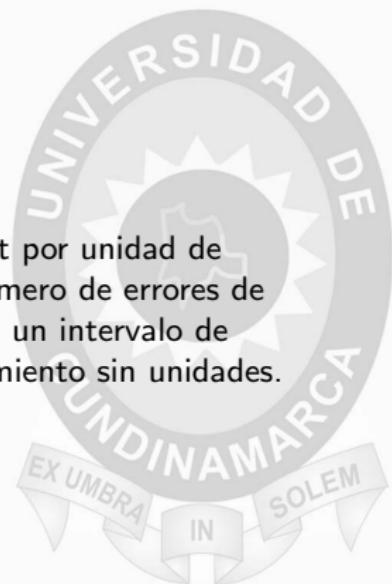


# Lab23 BER

# BER

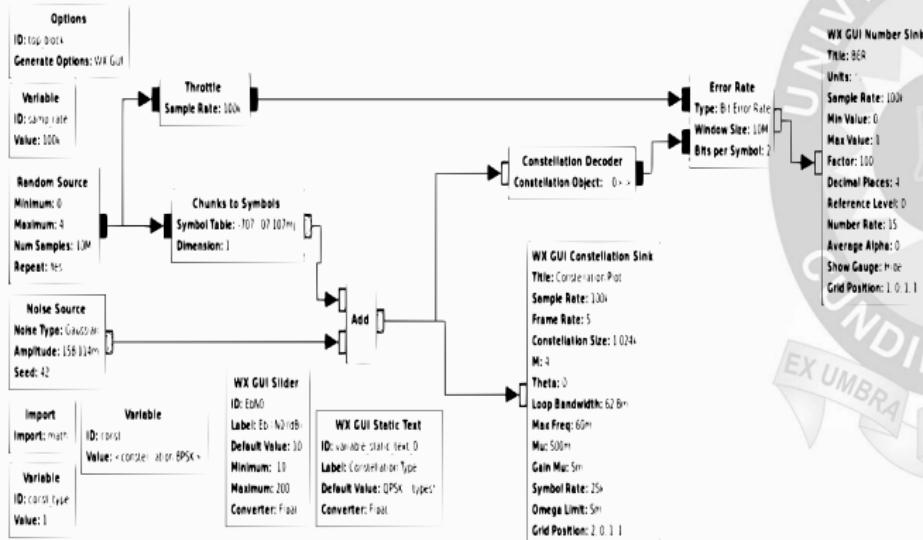
## BER

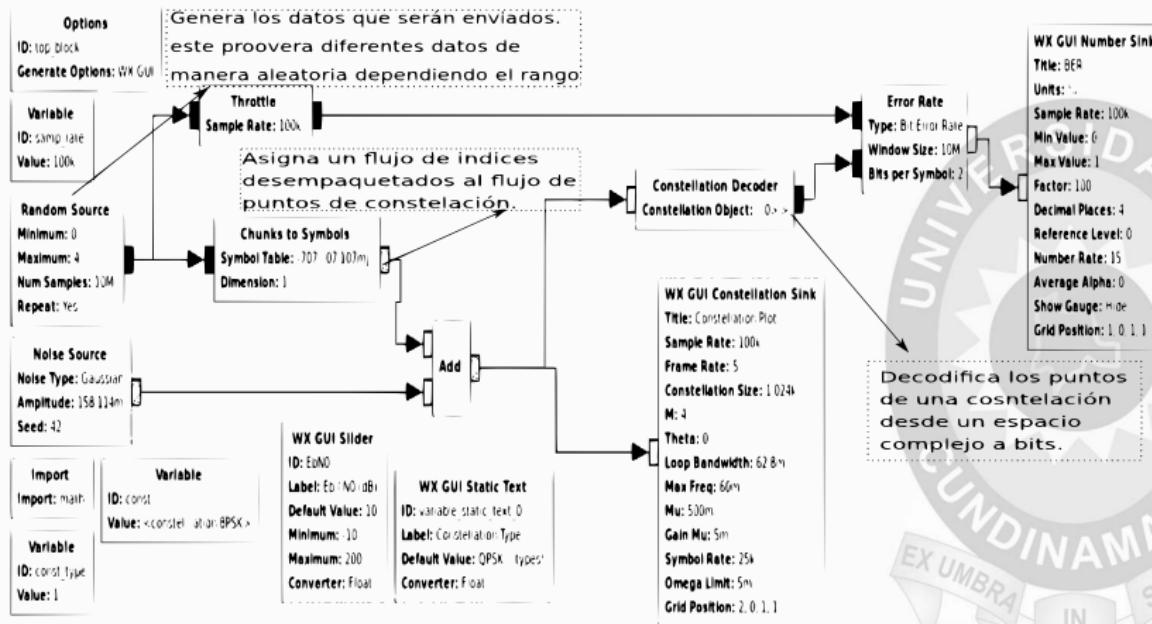
La tasa de error de bit (BER) es el número de errores de bit por unidad de tiempo. La relación de error de bit (también BER) es el número de errores de bit dividido por el número total de bits transferidos durante un intervalo de tiempo. La relación de error de bit es una medida de rendimiento sin unidades.

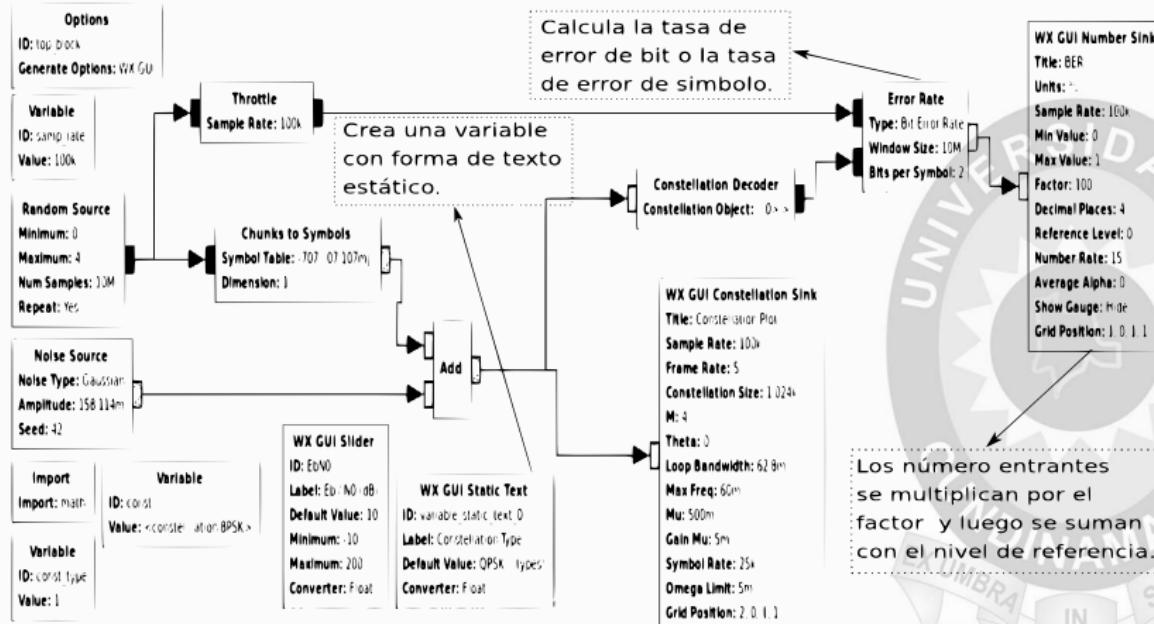


# BER

En GNU Radio para calcular la tasa de error en un modulación utilizamos el siguiente diagrama

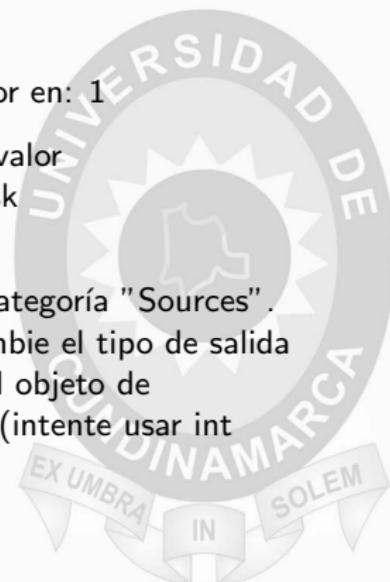






# BER

- Agregue una variable "consttype" y establezca su valor en: 1
- Agregue una nueva variable "const", y establezca su valor en:(digital.constellationbpsk (),digital.constellationqpsk (),digital.constellation8psk ())
- En el lado derecho, busque "Random Source" en la categoría "Sources". Haga doble clic en el bloque "Fuente aleatoria" y cambie el tipo de salida a "Byte", el máximo a. Constelación es el nombre del objeto de constelación GnuRadio y número de muestras a 10M (intente usar int (10e6) en lugar de 1000000) .



Properties: Random Source

General Advanced Documentation

ID: analog\_random\_source\_x\_0

Output Type: Byte

Minimum: 0

Maximum: const[const\_type].arity()

Num Samples: int(10e6)

Repeat: Yes

**Cambia el tipo de salida a byte.**

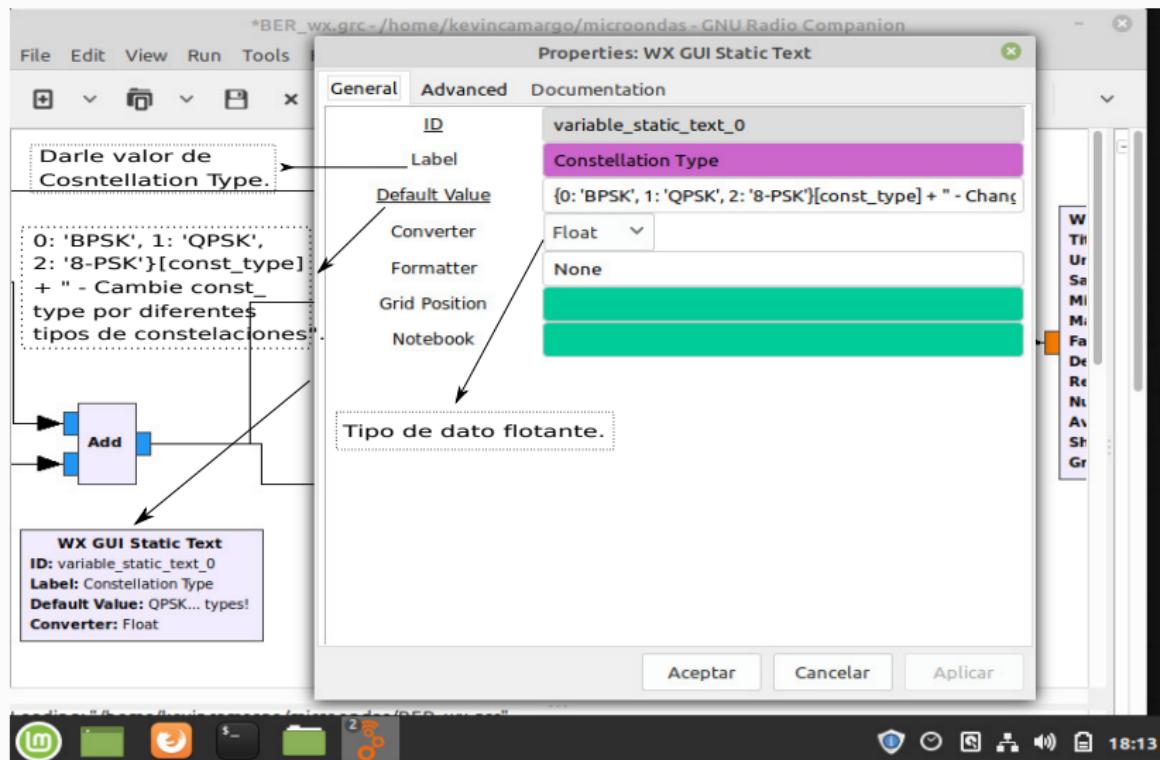
**Asigne en maximo: const[const\_type].arity()**

Aceptar Cancelar



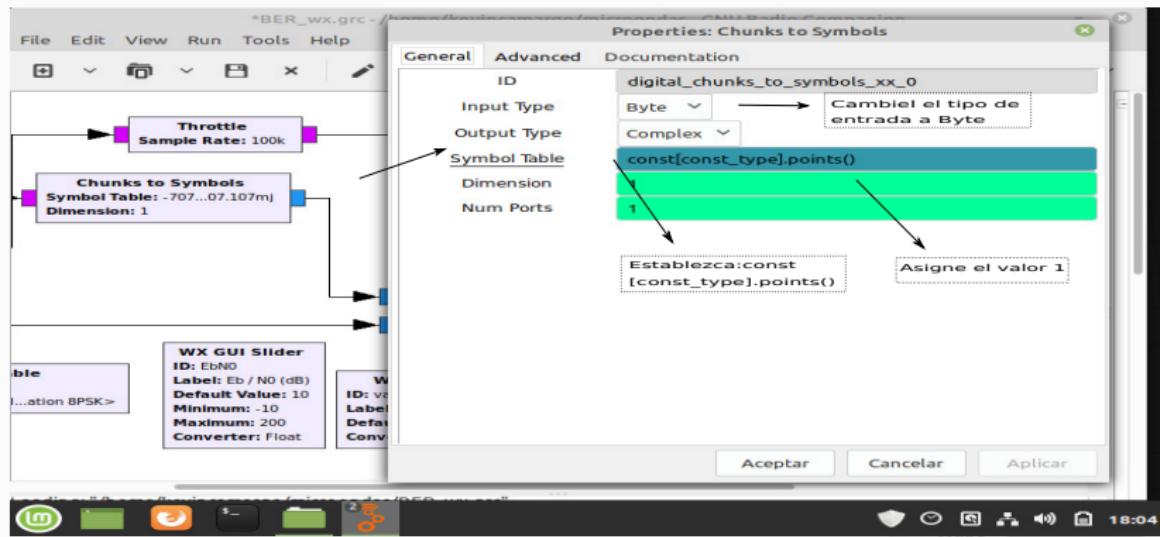
# BER

- Agrege un "WX Static Text"

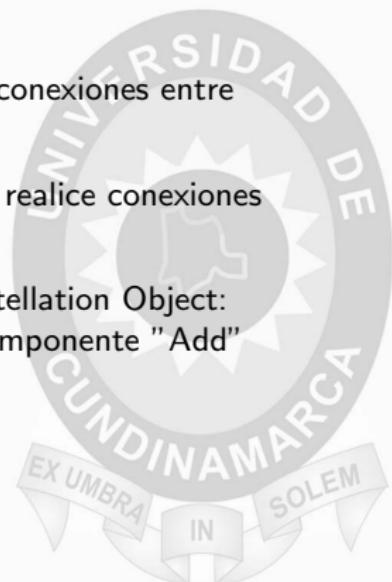


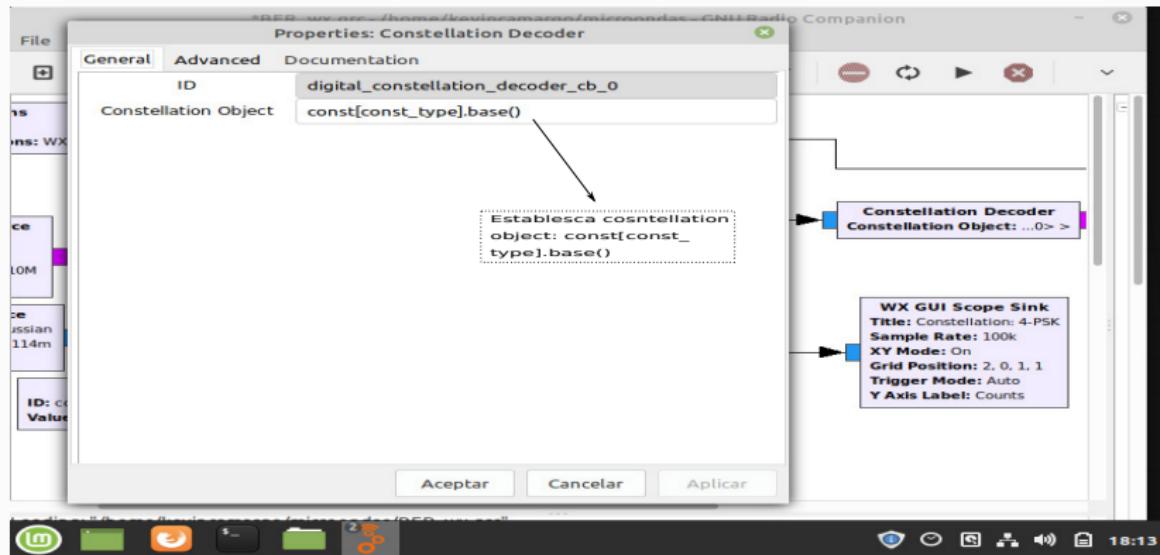
# BER

- Agregue WX GUI Slider, valores mínimos establecidos en -100, valor máximo establecido en 100, valores predeterminados establecidos en 10. Nombre "EbN0" este control deslizante como "EbN0" (ID de objeto).
- Importar una biblioteca "math" (componente "Import", valor: "import math")
- Agregue "Noise source" de la categoría "Sources" y cambie la amplitud como  $1.0 / \text{math.sqrt}(2.0 * \text{const}[\text{consttype}].\text{bitspersymbol}()) * 10 ** (\text{EbN0} / 10)$ .
- Agregue "Chunks to symbols" de la categoría "Misc Conversion" y cambie el tipo de entrada a "Byte", tabla de símbolos para "const [consttype].points()", la dimensión a 1.

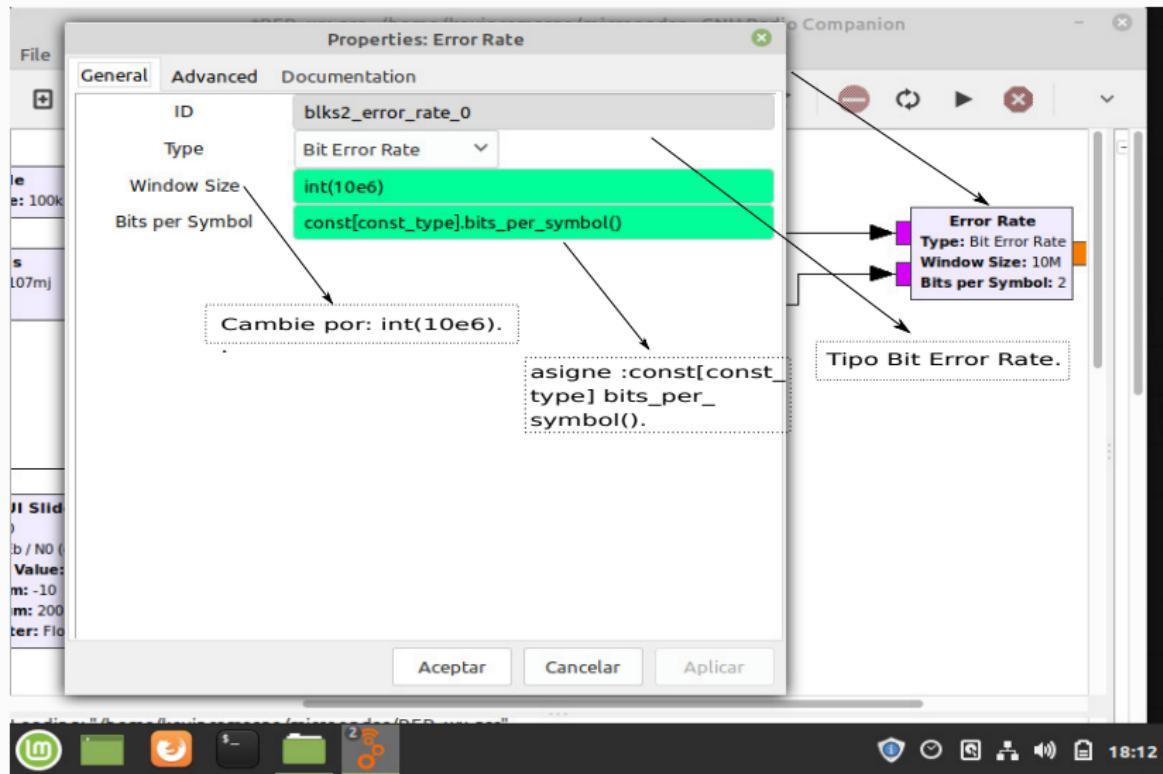


- Agregue "Add" de la categoría "Operadores" y haga conexiones entre "Chunks to Symbols" y " Noise Source ".
- Agregue "WX Constellation Sink" de la categoría " " y realice conexiones entre este componente y el componente "Add".
- Agregue " Constellation Decoder" Establezca en Constellation Object: const [tipoconst] .base (). Conecte a su entrada el componente "Add"

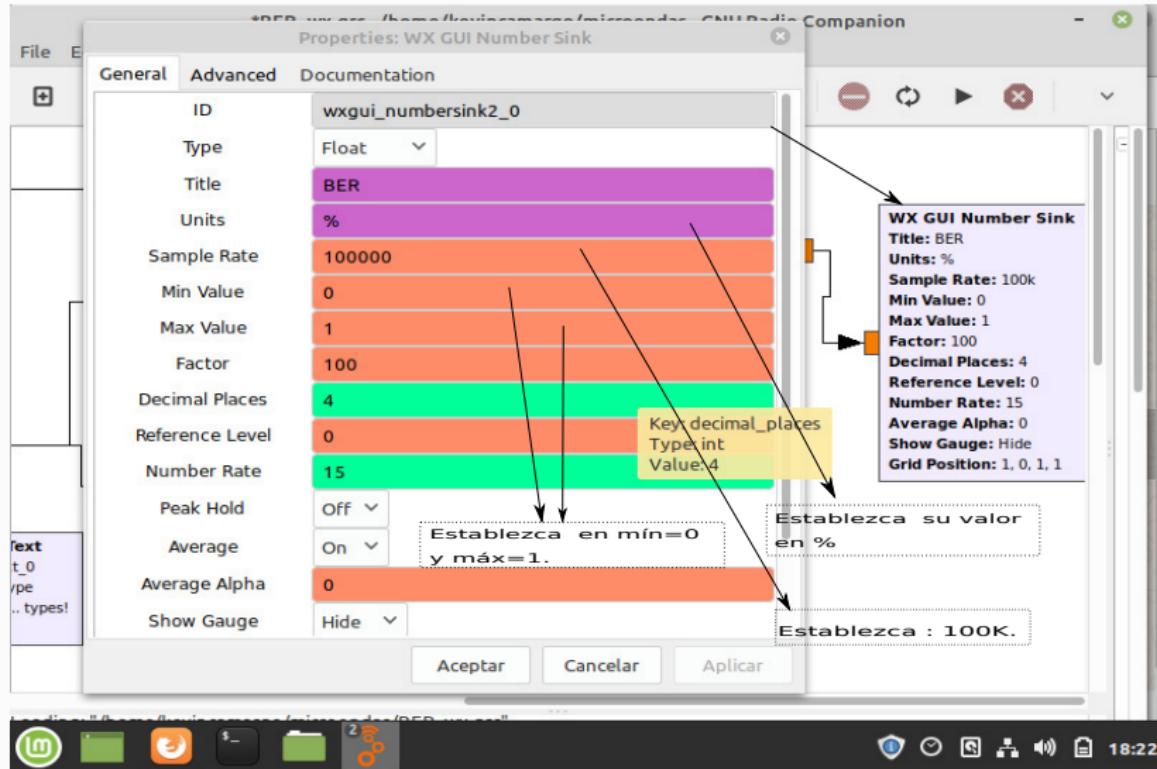


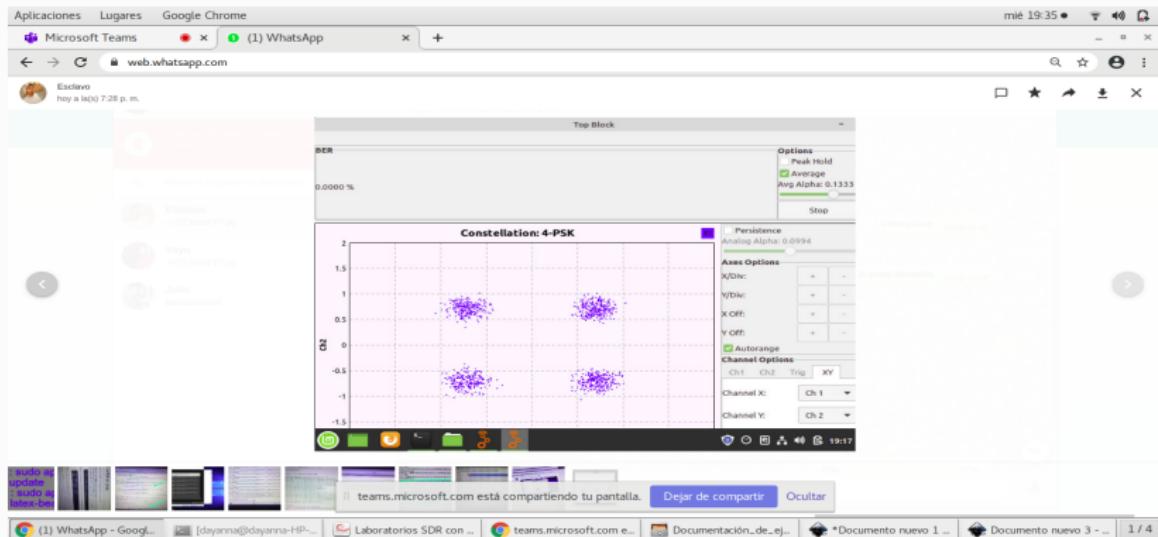


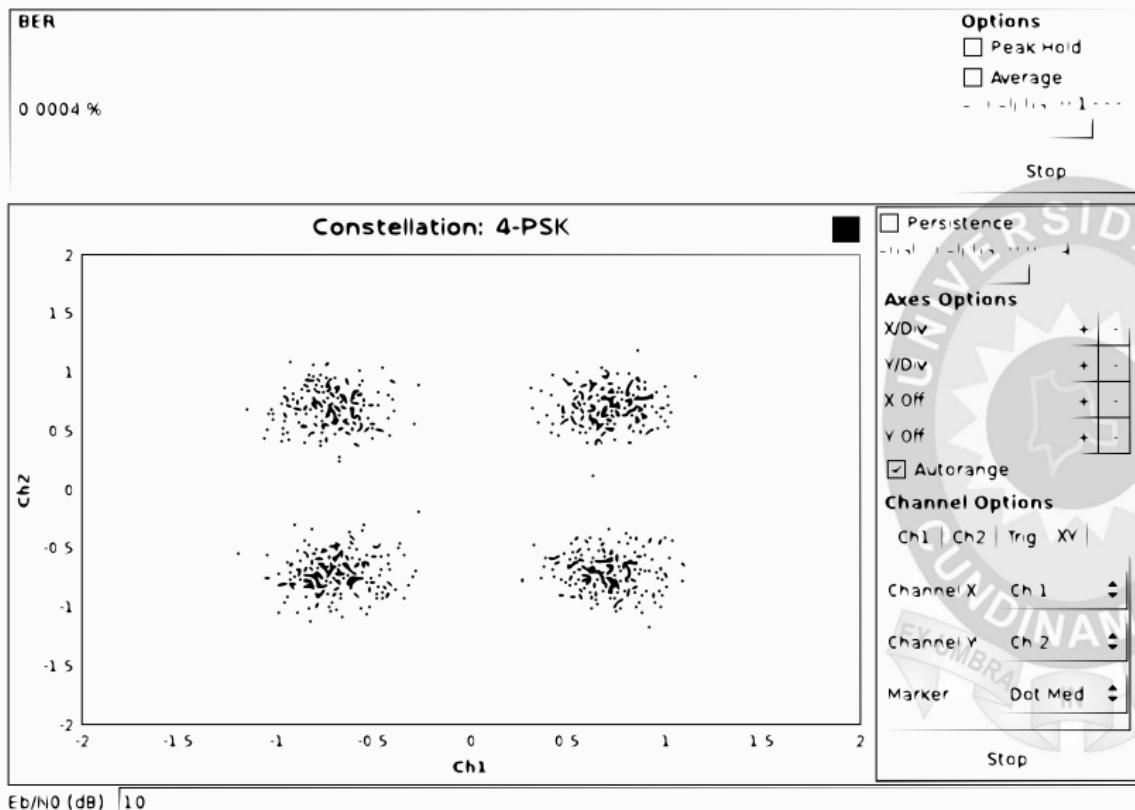
- Agregue "Error Rate" y cambie el tamaño de la ventana a 10M (int (1e7)), el Bits per símbolo a "const[consttype].bitspersymbol ()").



- Agregue "WX Number Sink". Establezca la entrada en flotante, min a 0 y max a 1.







# Lab24

## Constelación y FFT, Modulación BPSK

# Modulación BPSK

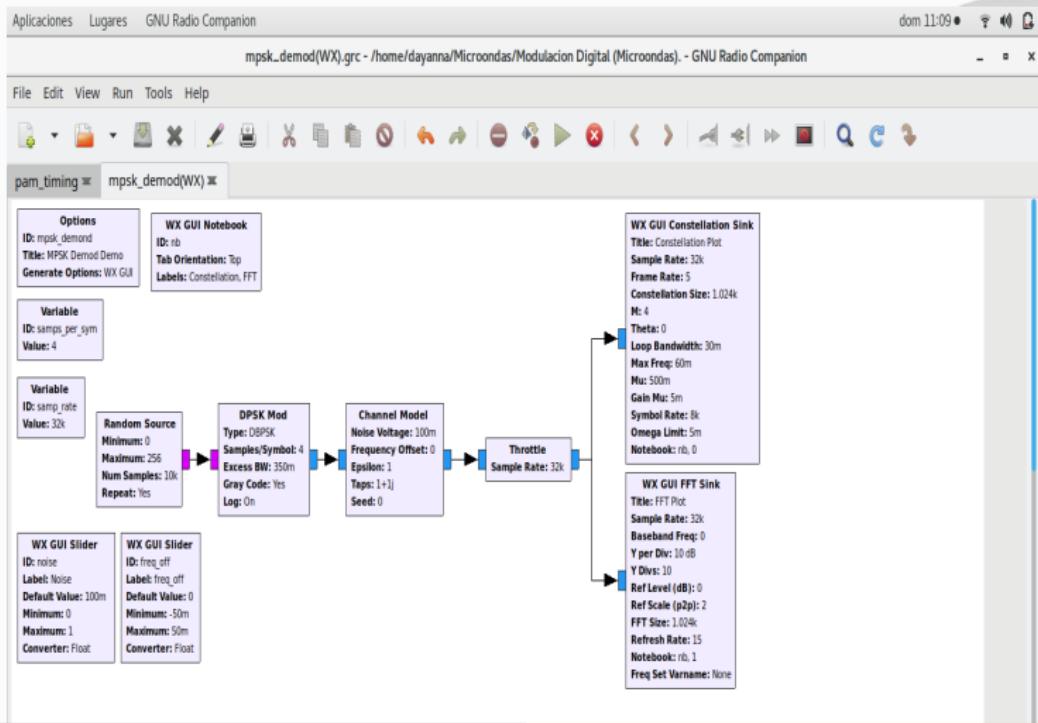
La modulación BPSK o Modulación por desplazamiento de fase binaria, es una modulación digital que se representa mediante un diagrama de constelación conformado por dos puntos equidistantes del origen de coordenadas.

Representa la modulación por desplazamiento de dos símbolos, con un bit de información cada uno, cada una de estas fases separadas a  $180^\circ$ , estos símbolos tienen un valor de salto de fase de  $0^\circ$  para (1) y de  $180^\circ$  para (0).

Este tipo de modulación digital (BPSK), es equivalente a la modulación 2-QAM, y constelación de esta modulación se representa en el diagrama del plano I-Q.

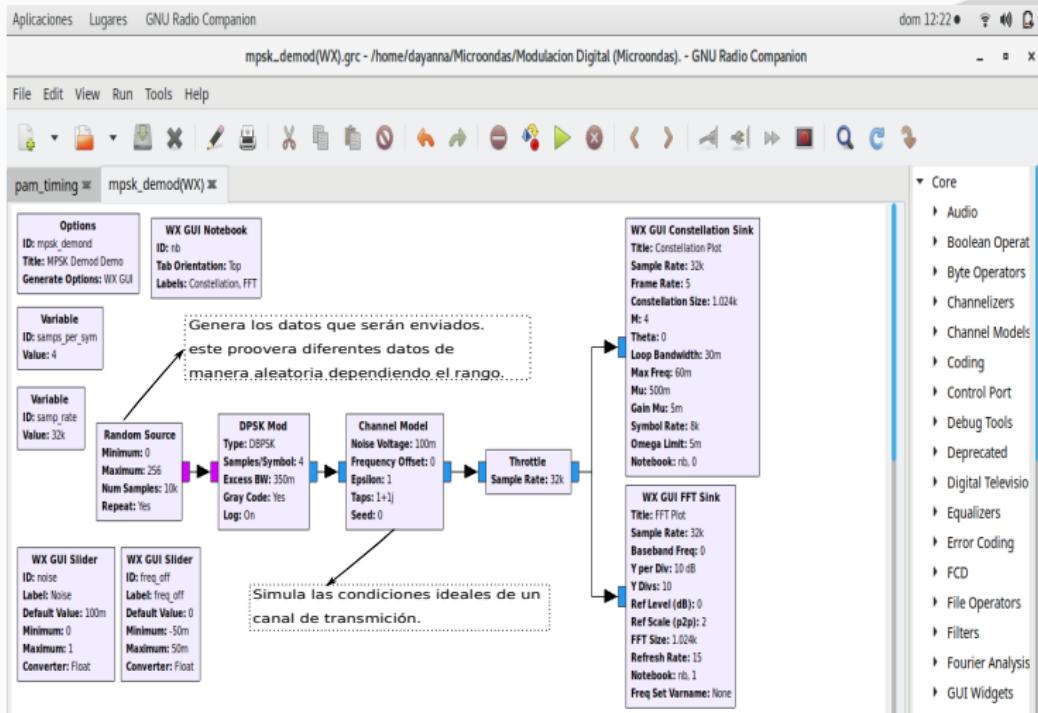
# Modulación BPSK

Como obtener la constelación en el plano de diagrama I-Q de una modulación BPSK, y su FFT.



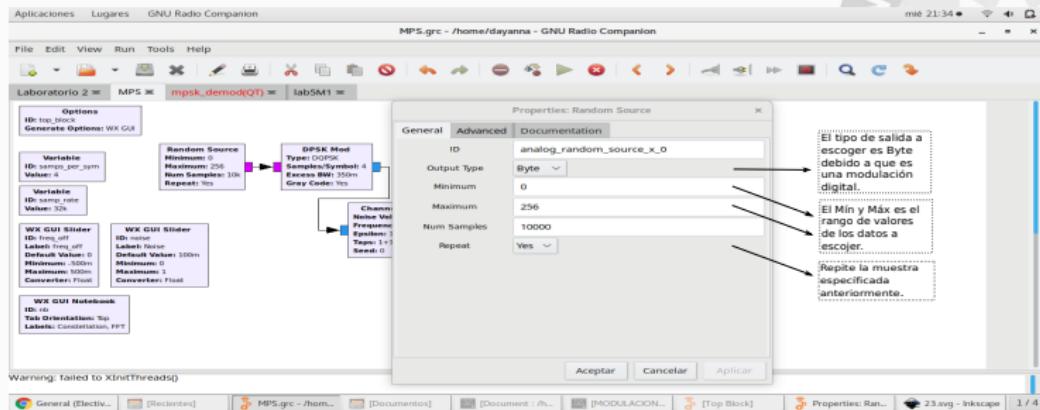
# Constelación y FFT; BPSK

Para observar la constelación y la señal FFT de una modulación BPSK, se tiene en cuenta la utilización del Channel Model y Random Source.



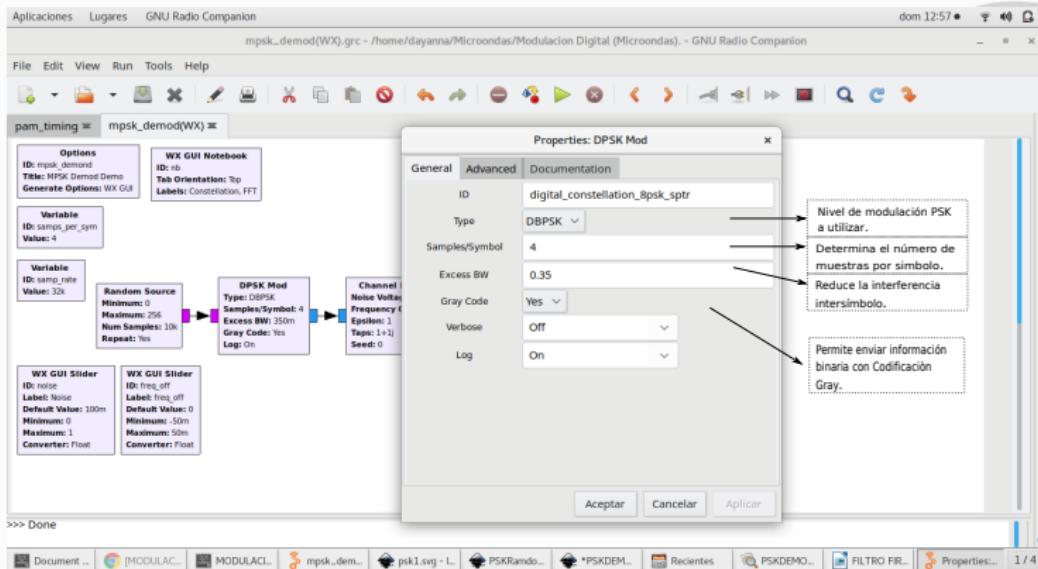
# Configuración de Bloque

Para generar los datos que serán enviados mediante el canal de transmisión, se debe configurar el bloque Random Source.



# Configuración de Modulador

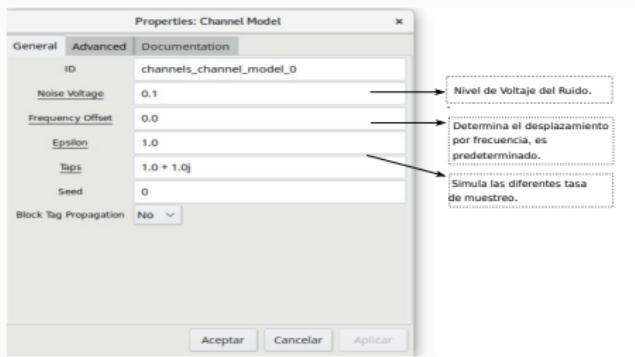
Realizamos la configuración previa en el bloque para realizar la modulación BPSK



En la modulación BPSK, la información que se transmite a través de un canal de comunicación se envía durante la fase de la portadora, una fase particular de  $180^\circ$  se usa para representar la información discreta.

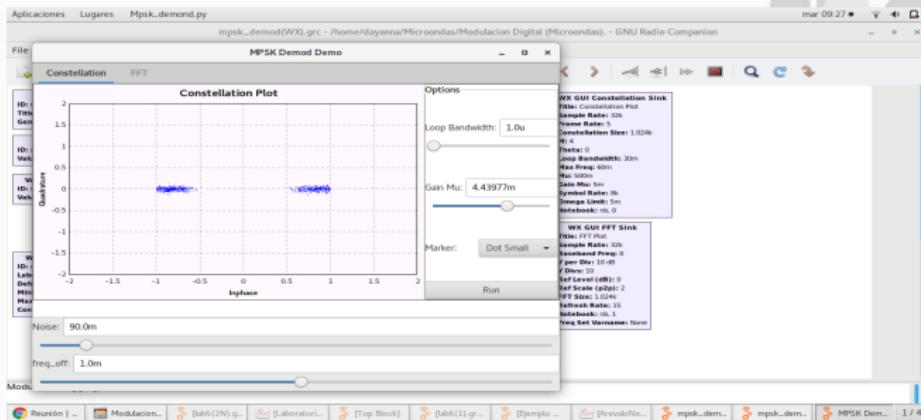
# Configuración del Canal

La simulación del canal se hace mediante el bloque Channel Model, este simula un canal AWGN.



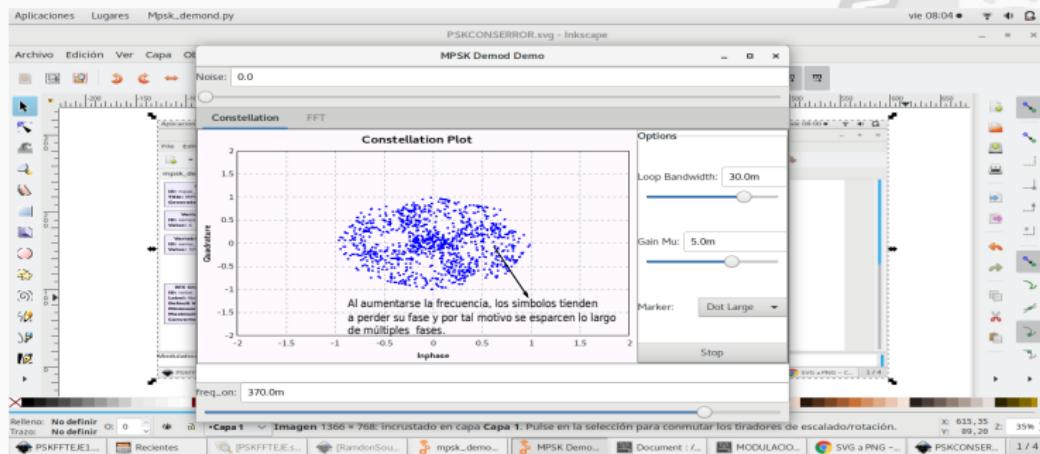
# Constelación BPSK

En la modulación se observa el desfase de  $180^\circ$  entre cada símbolo de la constelación del diagrama I-Q, donde I representa la fase y Q la cuadratura, pero en este caso la cuadratura no es utilizada ya que no es una modulación de tipo cuadrifásica.



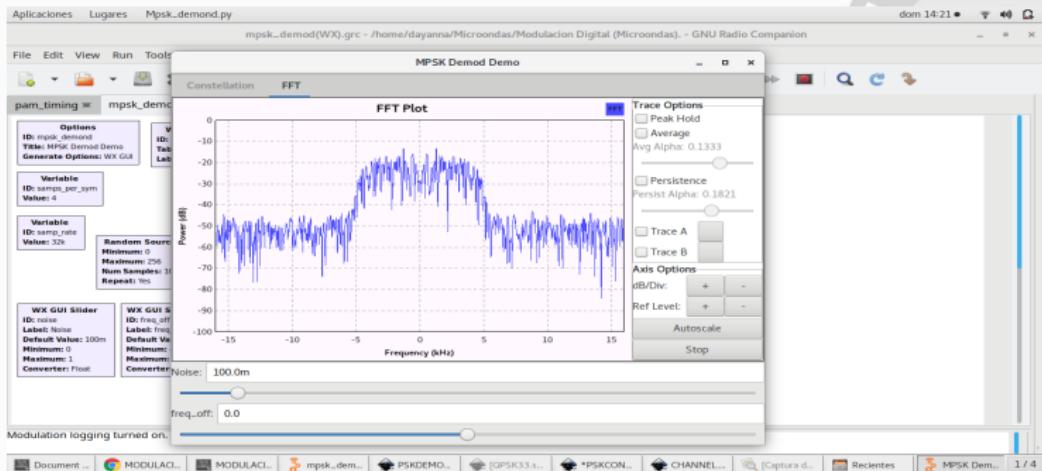
# Constelación BPSK

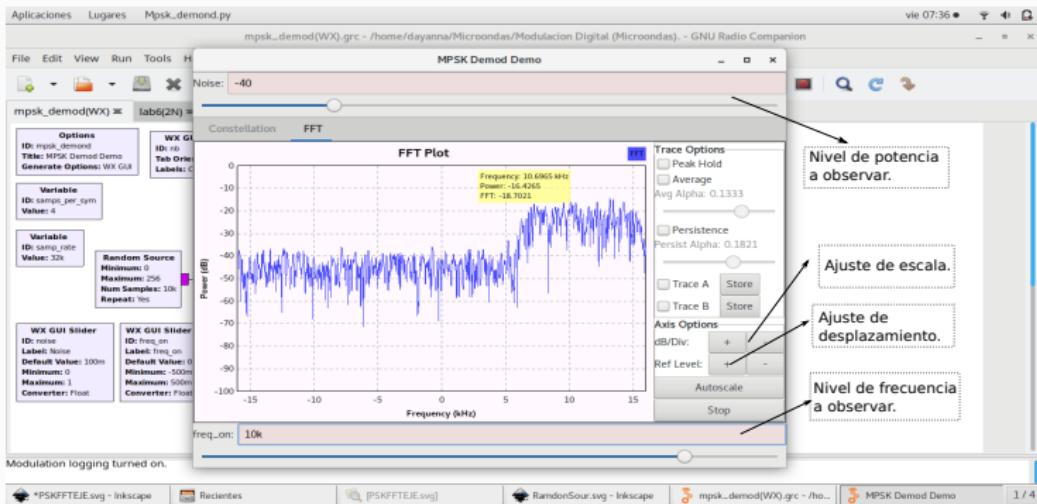
Cuando existe dispersión en los símbolos durante diferentes fases, se considera como un error, debido que a que los símbolos que contienen información se perderán, es decir la información no será transmitida.



# FFT, Modulación BPSK

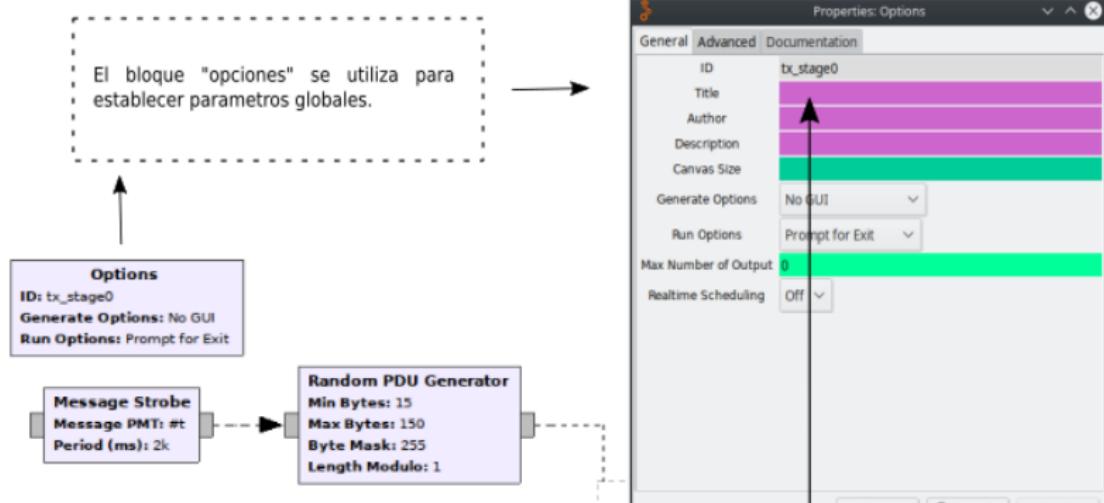
La FFT en la modulación BPSK, nos permite analizar y transformar una señal en el dominio de la frecuencia.





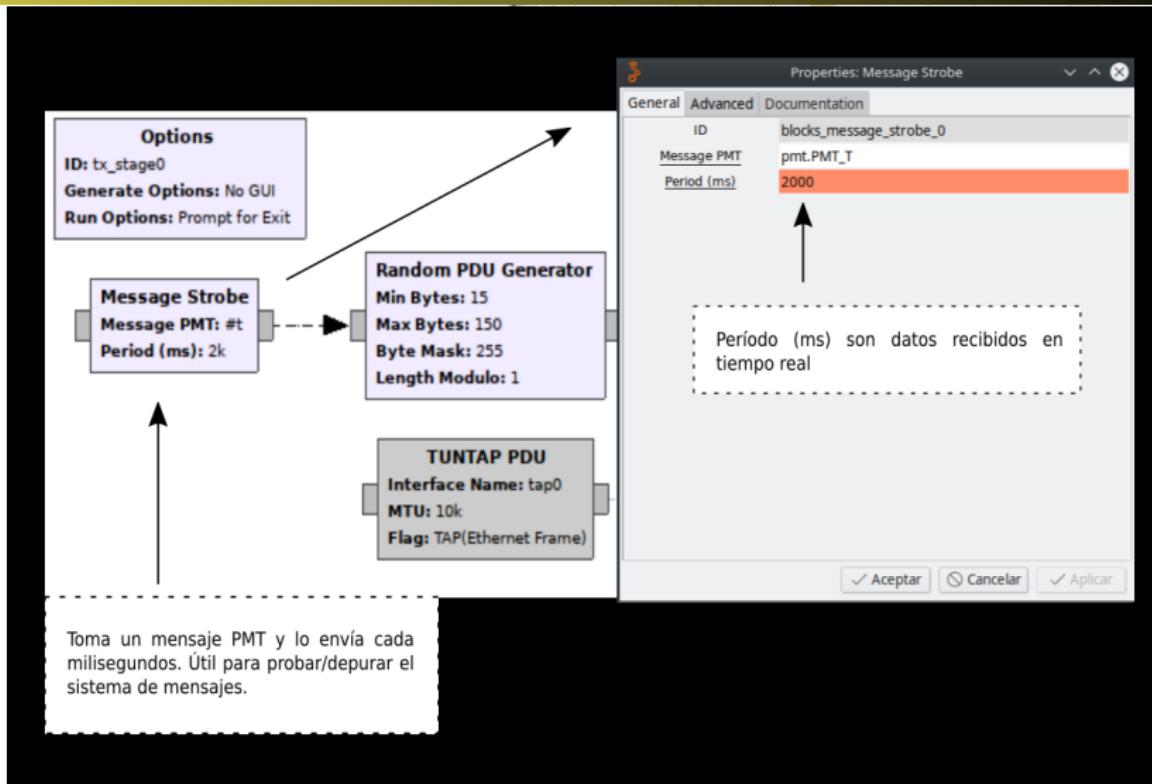
# Lab17 GENERADOR ALEATORIO DE PDU

# PASO 1: UNIDADES DE DATOS DE PROTOCOLOS (PDU)

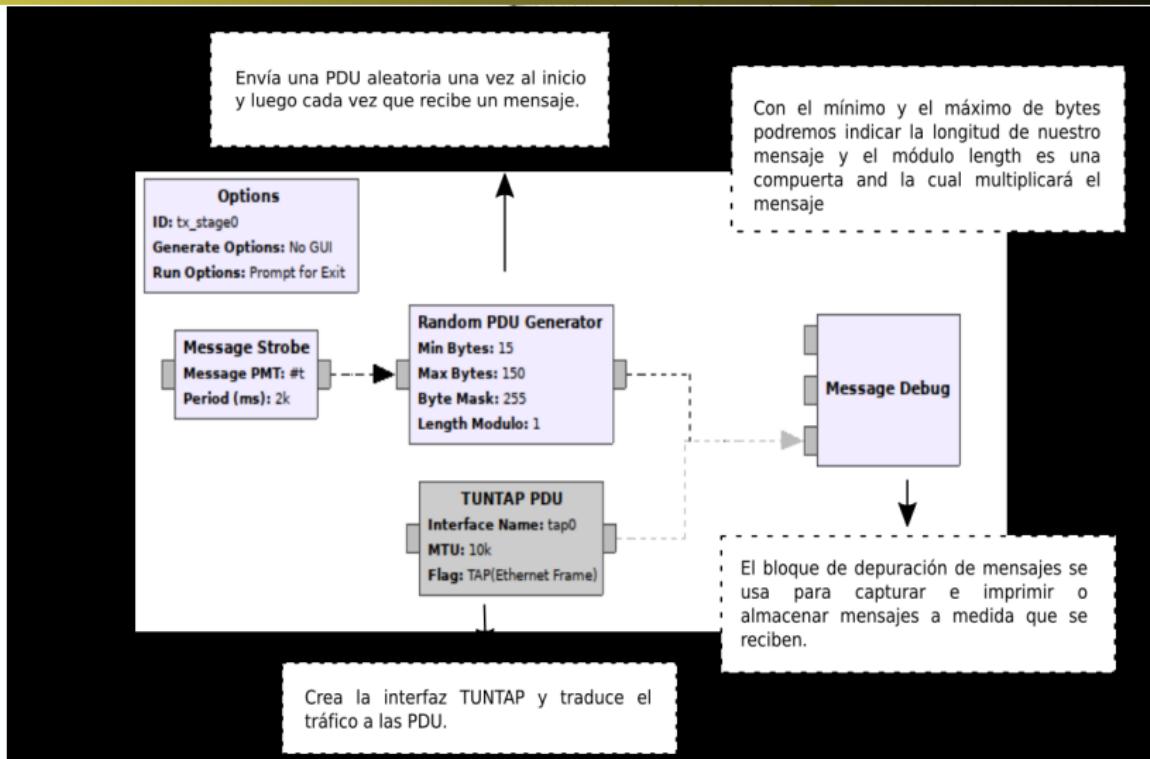


El ID es como quedará guardado nuestro archivo .py para poder verlo por consola. Prompt for Exit es una forma segura de cerrar el programa por consola.

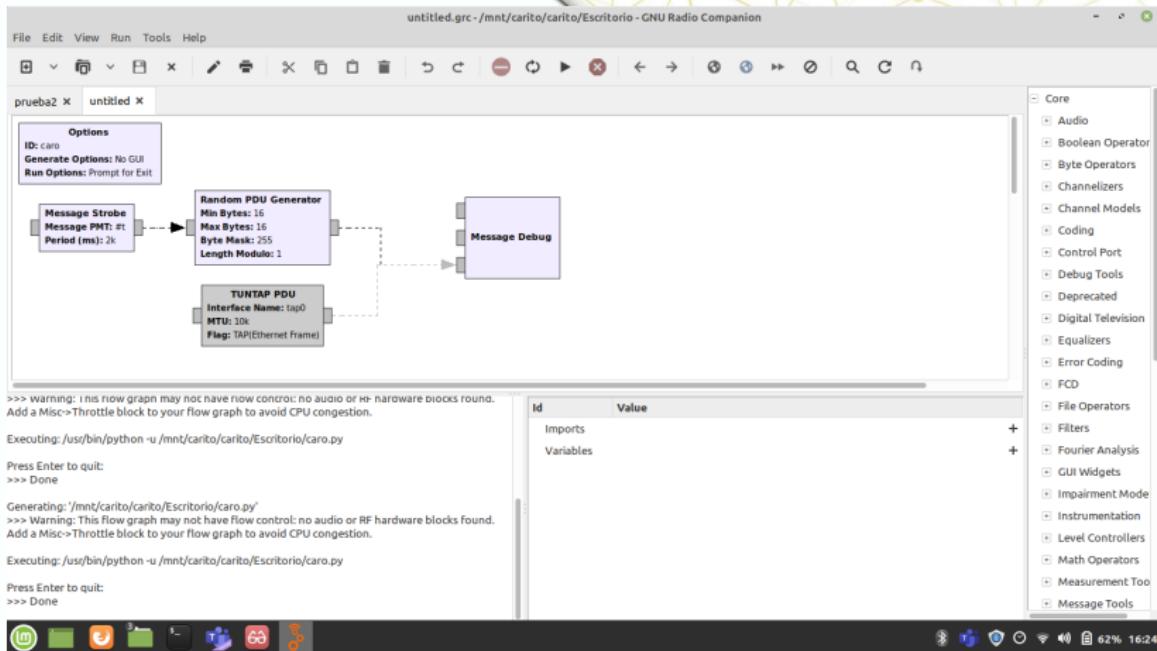
# PASO 1: UNIDADES DE DATOS DE PROTOCOLOS (PDU)



# PASO 1: UNIDADES DE DATOS DE PROTOCOLOS (PDU)



# PASO 1: UNIDADES DE DATOS DE PROTOCOLOS (PDU)



# RESPUESTAS POR MEDIOS DE INTERFAZ GRÁFICA

```
>>> Done
```

```
Generating: '/mnt/carito/carito/Escritorio/caro.py'
```

```
>>> Warning: This flow graph may not have flow control: no audio or RF hardware blocks found.  
Add a Misc->Throttle block to your flow graph to avoid CPU congestion.
```

```
Executing: /usr/bin/python -u /mnt/carito/carito/Escritorio/caro.py
```

Press Enter to quit:

```
>>> Done
```

## REPUESTA POR CONSOLA

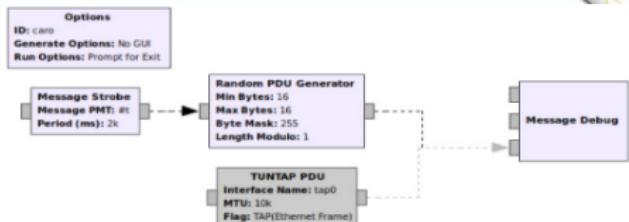
```
* MESSAGE DEBUG PRINT PDU VERBOSE *
() pdu length = 125
contents =
0000: d0 22 e7 d5 20 f8 e9 38 a1 4e 18 8c 47 30 8c fe
0010: f5 ff f7 f7 28 b9 f8 fb f5 1c 7c cc cc 4c 24 01
0020: 6b 1c ea a3 ca e0 f5 80 a7 cc 09 5c d9 36 ef ae
0030: ad 66 c1 bd be 79 64 6c a7 2c 2b 4d b4 cc 08 51
0040: 46 df 0b 26 18 fe d2 d2 b1 20 51 c3 f3 7d 08 a9
0050: 70 20 61 35 c3 0d cb 09 2f 68 7d 75 72 7c a5 cb
0060: b5 eb c1 ce 46 b4 ae 00 a7 b5 29 a4 1e 74 7f c6
0070: f5 92 57 e0 95 ce 39 04 c0 d2 41 d2 81
*****
Press Enter to quit:
```

# DEFINICIONES

**PTM** Los tipos polimórficos se utilizan como portadores de datos de un bloque / subprocesso a otro.

**PDU** Una PDU (unidad de datos de protocolo) en GNU Radio tiene un tipo PMT especiales un par de diccionario y un tipo de vector uniforme.

## PASO 2: DETECCION DE ERRORES CRC

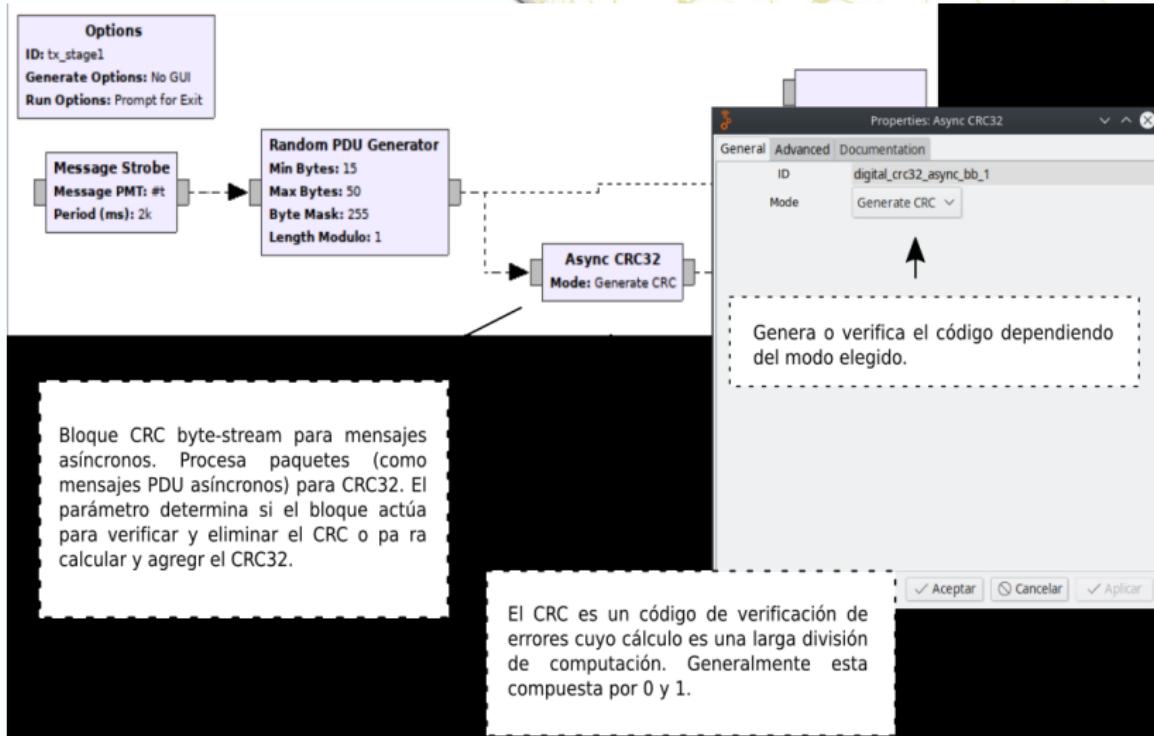


```
>> warning: This flow graph may not have flow control: no audio or RF hardware blocks found.  
Add a Misc->Throttle block to your flow graph to avoid CPU congestion.  
xecuting: /usr/bin/python -u /mnt/carito/carito/Escritorio/caro.py  
ress Enter to quit:  
>> Done  
enerating: '/mnt/carito/carito/Escritorio/caro.py'  
>> Warning: This flow graph may not have flow control: no audio or RF hardware blocks found.  
Add a Misc->Throttle block to your flow graph to avoid CPU congestion.  
xecuting: /usr/bin/python -u /mnt/carito/carito/Escritorio/caro.py  
ress Enter to quit:  
>> Done
```

Id	Value
Imports	
Variables	

Console Panel  
Aquí se puede ver el resultado de cada simulación, el mensaje que se genera.

# ANEXO DE CÓDIGO DE COMPROBACIÓN DE ERRORES (PDU CON CRC)



# CALCULADORA DE CRC32

a mecánica de la informática con su lenguaje binario produce unas CRC simples. Los bits representados de entrada son alineados en una fila, y el  $(n + 1)$  representa el patrón de bits del divisor CRC (llamado polinomio) se coloca debajo de la parte izquierda del final de la fila. Aquí está la primera de ellas para el cálculo de 3 bits de CRC:

## CALCULADORA DE CRC32



```
11010011101100 <--- entrada
1011           <--- divisor (4 bits)
-----
01100011101100 <--- resultado
```

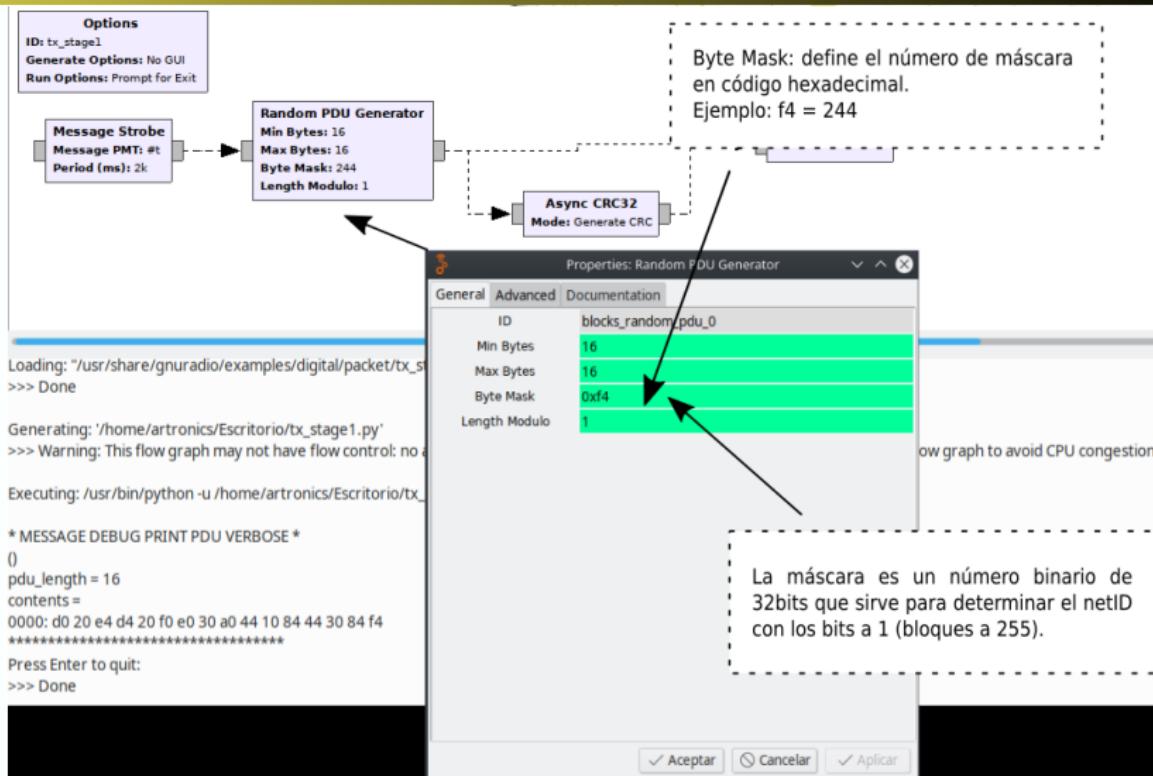
# CALCULADORA DE CRC32

Si la entrada que está por encima del extremo izquierdo del divisor es 0, no se hace nada y se pasa el divisor a la derecha de uno en uno. Si la entrada que está por encima de la izquierda del divisor es 1, el divisor es (Or) exclusiva en la entrada (en otras palabras, por encima de la entrada de cada bit el primer bit conmuta con el divisor). El divisor es entonces desplazado hacia la derecha, y el proceso se repite hasta que el divisor llega a la derecha, en la parte final de la fila de entrada. Aquí está el último cálculo.

# CALCULADORA DE CRC32

```
00000000001110 <--- resultado de la multiplicación de cálculo  
1011 <--- divisor  
-----  
0000000000101 <--- resto (3 bits)
```

# ANEXO DE CÓDIGO DE COMPROBACIÓN DE ERRORES (PDU CON CRC)



## REPUESTA POR MEDIO DE CONSOLE PANEL

```
* MESSAGE DEBUG PRINT PDU VERBOSE *
()
Press Enter to quit: pdu_length = 16
contents =
0000: d0 20 e4 d4 20 f0 e0 30 a0 44 10 84 44 30 84 f4
*****
* MESSAGE DEBUG PRINT PDU VERBOSE *
()
pdu_length = 20
contents =
0000: d0 20 e4 d4 20 f0 e0 30 a0 44 10 84 44 30 84 f4
0010: d3 55 3b 8f
*****
>>> Done
```

## REPUESTA POR CONSOLA

```
* MESSAGE DEBUG PRINT PDU VERBOSE *
```

```
() pdu length = 20
```

```
contents =
```

```
0000: d0 20 e4 d4 20 f0 e0 30 a0 44 10 84 44 30 84 f4
```

```
0010: d3 55 3b 8f
```

```
*****
```

```
Press Enter to quit:
```

## EJEMPLO CON MASCARA PAR

The screenshot shows the GNU Radio Companion (GRC) interface with a flowgraph titled "prueba2.grc".

**Block Diagram:**

- A "Message Strobe" block is connected to a "Random PDU Generator" block.
- The "Random PDU Generator" block has the following properties:
  - ID: blocks\_random\_pdu\_0
  - Min Bytes: 16
  - Max Bytes: 16
  - Byte Mask: 0xf4
  - Length Modulo: 1

**Properties Dialog:**

The "Properties: Random PDU Generator" dialog is open, showing the "General" tab with the same configuration values.

**Output Terminal:**

```
* MESSAGE DEBUG PRINT PDU VERBOSE *
()
pdu_length = 16
contents =
0000: d0 20 e4 d4 20 f0 e0 30 a0 44 10 84 44 30 84 f4
*****
* MESSAGE DEBUG PRINT PDU VERBOSE *
()
pdu_length = 20
contents =
0000: d0 20 e4 d4 20 f0 e0 30 a0 44 10 84 44 30 84 f4
0010: d3 55 3b 8f
*****
Press Enter to quit:
>>> Done
```

**System Tray:**

Icons for network, battery (91%), and time (11:29) are visible in the system tray.

# CALCULADORA DE CRC32

La calculadora de CRC32 da el resultado de forma invertida.

## Calculadora de CRC en línea

PHP en GitHub Fuentes C # en GitHub Fuentes Java en GitHub

```
d0 20 e4 d4 20 f0 e0 30 a0 44 10 84 44 30 84 f4
```

Tipo de entrada:  ASCII  Maleficio Tipo de salida:  MALEFICIO  DIC  OCT  COMPARTIMIENTO  
 Mostrar datos procesados (HEX)

pdu\_length = 20  
contents =  
0000: d0 20 e4 d4 20 f0 e0 30 a0 44 10 84 44 30 84 f4  
0010: d3 55 3b 8f

Datos procesados

```
0xd0 0x20 0xe4 0xd4 0x20 0xf0 0xe0 0x30 0xa0 0x44 0x10 0x84 0x44 0x30 0x84 0xf4
```

Algoritmo	Resultado	Cheque	escuela politécnica	En eso	RefIn	RefOut	XorOut
CRC-32	0x8F3B55D3	0xCBFB43926	0x04C11DB7	0xFFFFFFFF	cierto	cierto	0xFFFFFFFF

## REPUESTA POR CONSOLA

```
* MESSAGE DEBUG PRINT PDU VERBOSE *
```

```
() pdu length = 20
```

```
contents =
```

```
0000: d0 20 e4 d4 20 f0 e0 30 a0 44 10 84 44 30 84 f4
```

```
0010: d3 55 3b 8f
```

```
*****
```

```
Press Enter to quit:
```

## EJEMPLO CON MASCARA IMPAR

The screenshot shows a GNU Radio Companion (GRC) interface with a flowgraph. On the left, there is an "Options" block with the ID "carolina", "Generate Options: No GUI", and "Run Options: Prompt for Exit". A "Message Strobe" block is connected to a "Random PDU Generator" block. The "Random PDU Generator" block has the following parameters set: Min Bytes: 16, Max Bytes: 16, Byte Mask: 255, and Length Modulo: 1. A properties dialog box titled "Properties: Random PDU Generator" is open, showing the same parameter values. The output of the "Random PDU Generator" block is displayed in the terminal window below, showing two hex dumps of 16 bytes each, both starting with 0000: d0 22 e7 d5 20 f8 e9 38 a1 4e 18 8c 47 30 8c fe. The terminal also prompts the user to press Enter to quit and indicates the process is done.

```
* MESSAGE DEBUG PRINT PDU VERBOSE *
()
pdu_length = 16
contents =
0000: d0 22 e7 d5 20 f8 e9 38 a1 4e 18 8c 47 30 8c fe
*****
* MESSAGE DEBUG PRINT PDU VERBOSE *
()
pdu_length = 20
contents =
0000: d0 22 e7 d5 20 f8 e9 38 a1 4e 18 8c 47 30 8c fe
0010: 0f 2a 04 f5
*****
Press Enter to quit:
>>> Done
```

## REPUESTA POR CONSOLA

```
* MESSAGE DEBUG PRINT PDU VERBOSE *
```

```
() pdu length = 20
```

```
contents =
```

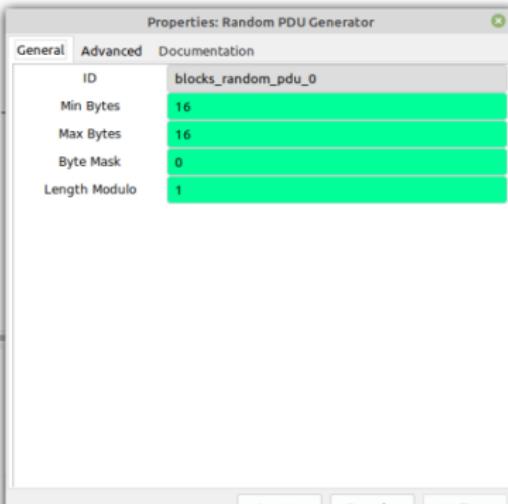
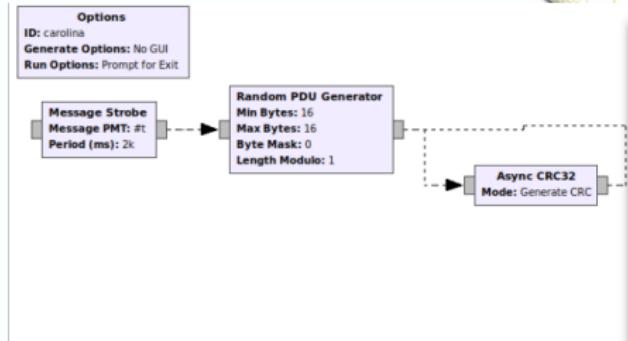
```
0000: d0 22 e7 d5 20 f8 e9 38 a1 4e 18 8c 47 30 8c fe
```

```
0010: 0f 2a 04 f5
```

```
*****
```

```
Press Enter to quit:
```

## EJEMPLO CON MASCARA 0



```
>>> Done

Generating: '/mnt/carito/carito/Escritorio/micro pruebas /carolina.py'
>>> Warning: This flow graph may not have flow control: no audio or RF hardware blocks found.
Add a Misc->Throttle block to your flow graph to avoid CPU congestion.

Executing: /usr/bin/python -u /mnt/carito/carito/Escritorio/micro pruebas /carolina.py

* MESSAGE DEBUG PRINT PDU VERBOSE *
Press Enter to quit: ()
pdu_length = 16
contents =
0000: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

## REPUESTA POR CONSOLA

```
* MESSAGE DEBUG PRINT PDU VERBOSE *
```

```
() pdu length = 20
```

```
contents =
```

```
0000: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
0010: 55 4b bb ec
```

```
*****
```

```
Press Enter to quit:
```

# BIBLIOGRAFÍA

# Referencias |

[1] Seeber, Balint.

"GNU Radio Tutorials Labs 1 – 5". [Online]

[https://files.ettus.com/tutorials/labs/Lab\\_1-5.pdf](https://files.ettus.com/tutorials/labs/Lab_1-5.pdf).

[2] Vachhani, Khyati. Gokhruwala, Kenil; Kumar, Jay

"Design Analysis of Digital Modulation Schemes with GNU Radio".

[Descargado de:] [https://www.researchgate.net/publication/281106848\\_Design\\_Analysis\\_of\\_Digital\\_Modulation\\_Schemes\\_with\\_GNU\\_Radio](https://www.researchgate.net/publication/281106848_Design_Analysis_of_Digital_Modulation_Schemes_with_GNU_Radio).

[3] Wikipedia.

"Frecuencia modulada". [Online]

[https://es.wikipedia.org/wiki/Frecuencia\\_modulada](https://es.wikipedia.org/wiki/Frecuencia_modulada).

[4] Vaught, Andy.

"Introduction to Named Pipes". [Online]

<https://www.linuxjournal.com/article/2156>.

## Referencias II

[5] **Wiki Opendigitalradio.**

"Simple FM transmitter using gnuradio". [Online]

[http://wiki.opendigitalradio.org/Simple\\_FM\\_transmitter\\_using\\_gnuradio](http://wiki.opendigitalradio.org/Simple_FM_transmitter_using_gnuradio).

[6] **Electronisys.**

"Ficha técnica de radio portátil UHF Motorola EP150". [Descargado de:]

[http://www.electronisys.cl/index.php?route=product/product&product\\_id=50](http://www.electronisys.cl/index.php?route=product/product&product_id=50).

[7] **Wikipedia.**

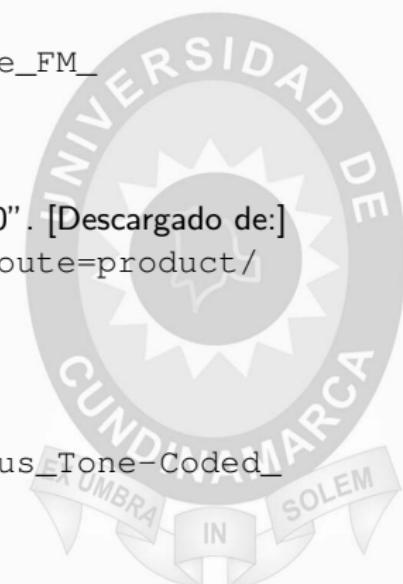
"Continuous Tone-Coded Squelch System". [Online]

[https://en.wikipedia.org/wiki/Continuous\\_Tone-Coded\\_Squelch\\_System](https://en.wikipedia.org/wiki/Continuous_Tone-Coded_Squelch_System).

[8] **Airspy.**

"Airspy Redefining the Radio Experience". [Online]

<https://airspy.com/download/>



# Referencias III

## [9] Motorola Solutions.

"Two-Way Radios". [Online]

<https://www.motorolasolutions.com/content/dam/msi/docs/enxl/products/two-way-radios-business/portable-radios/on-site-smallbusiness/>.

## [10] Documento PDF

"Lecture 9 Analog and Digital I/Q Modulation". [Online]

<http://web.mit.edu/6.02/www/f2006/handouts/Lec9.pdf>.

## [11] TOMASI, WAYNE.

"Sistemas de comunicaciones electrónicas". [Descargado de:]

<http://fernandoarciniega.com/books/sistemas-de-comunicaciones-electronicas-tomasi-4ta-edicion.pdf>.

# Referencias IV

## [12] Arévalo Nelson

"Transmisión y recepción de imagen con modulación QPSK usando radio definido por software". [Online] <https://repository.unimilitar.edu.co/handle/10654/16606>.

## [13] Aaron Scher

"Cómo convertir un flujo de datos digitales en una señal analógica de banda base utilizando un filtro FIR interpolador". [Online] [http://aaronsscher.com/GNU\\_Radio\\_Companion\\_Collection/BPSK\\_modulatorB.html](http://aaronsscher.com/GNU_Radio_Companion_Collection/BPSK_modulatorB.html).

## [14] Agencia Nacional Europea

"Señales de coseno realzado de raíz cuadrada ". [Online] [https://gssc.esa.int/navipedia/index.php/Square-Root\\_Raised\\_Cosine\\_Signals\\_\(SRRC\)/tutorials/labs/Lab\\_1-5.pdf](https://gssc.esa.int/navipedia/index.php/Square-Root_Raised_Cosine_Signals_(SRRC)/tutorials/labs/Lab_1-5.pdf).