Universidad Nacional de Río Cuarto

Facultad de ingeniería

Carrera: Ingeniería en Telecomunicaciones.

Materia: Sistemas y Señales II (Cód. 21).

Tema: Resolución Guía N°1 – GNU Radio.

Integrantes:

- Belotti, Malena. DNI: 44.433.255.
- Coser Lautaro. DNI: 43.230.317.
- Ratto Ruiz, Florencia. DNI: 44.644.091.

Ejercicio N°1

Dos señales analógicas $x_1(t) = cos2\pi(10)t$ y $x_2(t) = cos2\pi(50)t$. Las dos señales son muestreadas a una tasa de muestreo $f_s = 200\ [Hz]$.

a) Encuentre las secuencias $x_1[n]$ y $x_2[n]$.

b) Implemente usando el Companion, y visualice las señales.

Ahora cambie la tasa de muestreo f_s, a 40 muestras por segundo ($40\ [Hz]$). Vuelva a graficar cada señal.

c) Explique el fenómeno de Aliasing que se observa en las gráficas del inciso b.

d) ¿Qué otras señales producen un Aliasing de la señal $x_1[n]$ cuando $f_s = 40\ [Hz]$? Muestre un ejemplo en GNURADIO Companion.

- **Resolución:**

Para el caso a), identificamos las frecuencias de cada señal ($10\ [Hz]$ en la señal 1 y $50\ [Hz]$ en la señal 2), además, la frecuencia de muestreo es $f_s = 200\ [Hz]$. La secuencia $x_1[n]$ será:

$$x_1[n] = cos\left(2\pi(10) \cdot \frac{n}{200}\right)$$

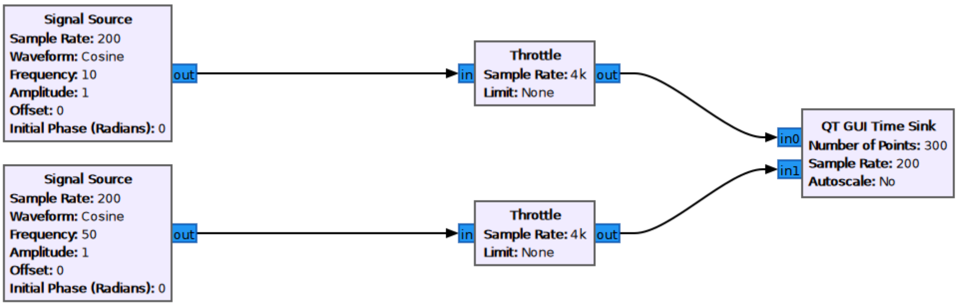
$$x_1[n] = cos\left(\frac{\pi n}{10}\right)$$

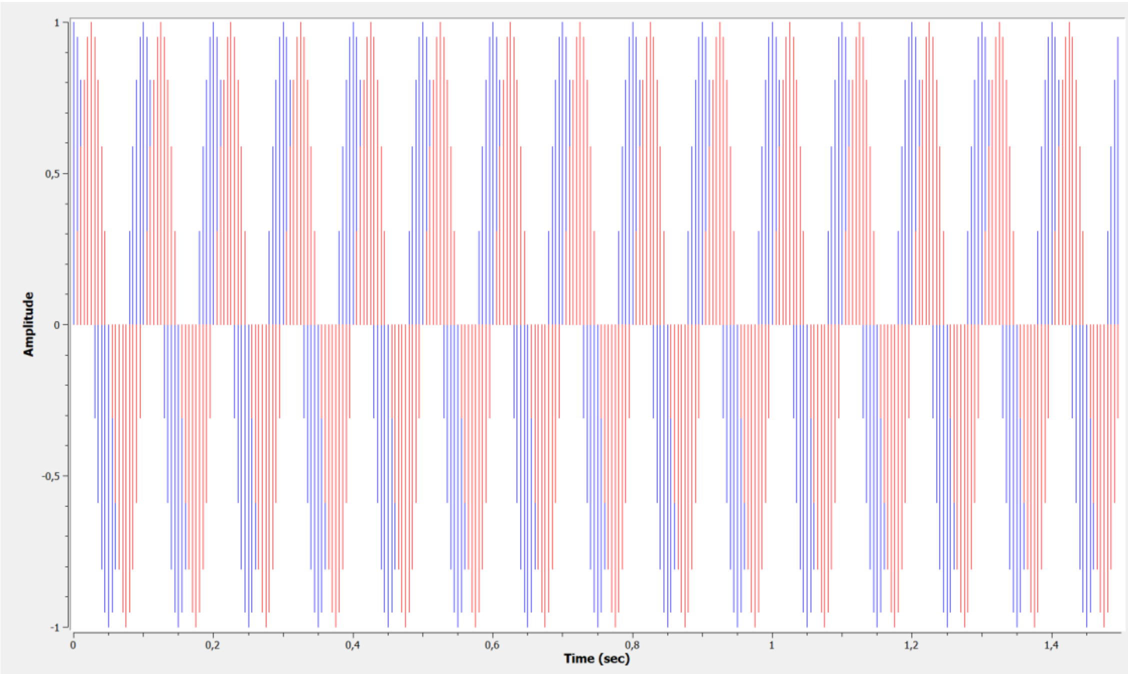
Para la secuencia $x_2[n]$:

$$x_2[n] = cos\left(2\pi(50) \cdot \frac{n}{200}\right)$$

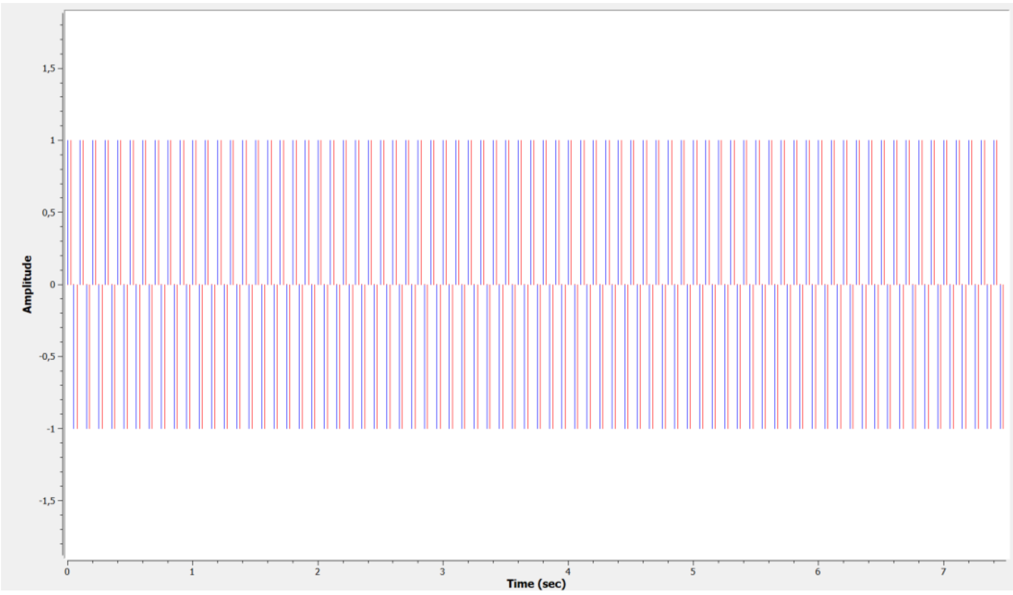
$$x_1[n] = cos\left(\frac{\pi n}{2}\right)$$

Para el caso b), graficamos las señales $x_1(t)$ y $x_2(t)$ en GNU:





Para el inciso c), cambiamos la frecuencia de muestre de $200\ [Hz]$ a $40\ [Hz]$, obteniendo el siguiente gráfico:



Para el caso c), definimos al fenómeno de Aliasing como a la distorsión que ocurre cuando una señal analógica continua en el tiempo es muestreada a una frecuencia que no es lo suficientemente alta. Cuando reducimos la frecuencia de muestreo, la señal en tojo comienza a parecerse a la azul, perdiendo información.

Si en el caso d), queremos que una señal se vea idéntica a la de $10\ [Hz]$ cuando se utiliza una frecuencia de muestreo de $10\ [Hz]$, se tiene que cumplir con:

$$|f \pm k \cdot 40| = 10$$

Por ejemplo, para una frecuencia $f = 50\ [Hz]$, tendremos:

$$|50 - 1 \cdot 40| = 10$$

De esta manera. Cualquier señal con una frecuencia que cumpla con $|f - k \cdot 40| = 10$, se verá de manera idéntica a la señal de $10\ [Hz]$.

En el caso e), se va a cumplir con cualquier señal que tenga una frecuencia que cumpla con $|f \pm k \cdot 40| = 10$.

Ejercicio N°3

Una señal de tiempo continuo $x(t)$ tiene la forma de:

$$x(t) = 5\cos\left(2\pi 5000 t\right)$$

Considere inicialmente una frecuencia de muestreo tal que permita una correcta reconstrucción de la señal. Utilice el bloque QT GUI Range del Companion para modificar la misma.

a) ¿Qué pasa cuando usamos una fs (fs: frequency sample, o frecuencia de muestreo, o tasa de muestreo) menor a la mínima requerida por el criterio de Nyquist)?

b) ¿Una fs igual a la de Nyquist que produce?

c) ¿Y una superior a la de Nyquist?, ¿Qué mejora aún más al aumentarla?

- **Resolución:**

Para empezar a resolver este ejercicio, se necesita encontrar nuestro valor de frecuencia de Nyquist. El Teorema de Nyquist nos dice que para evitar aliasing, nuestra frecuencia de muestreo debe cumplir con:

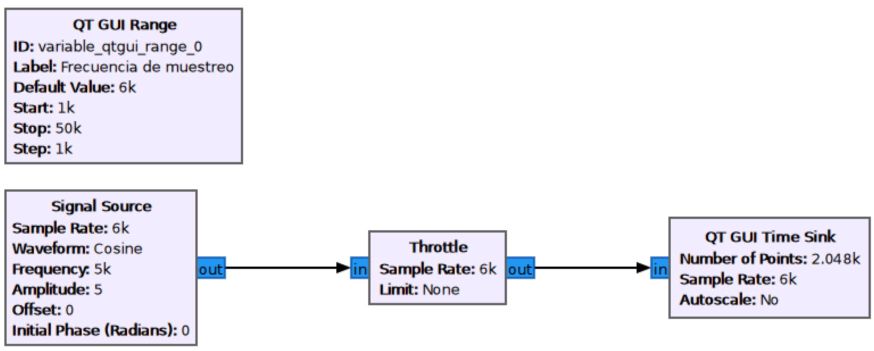
$$f_s \geq 2f_{max}$$

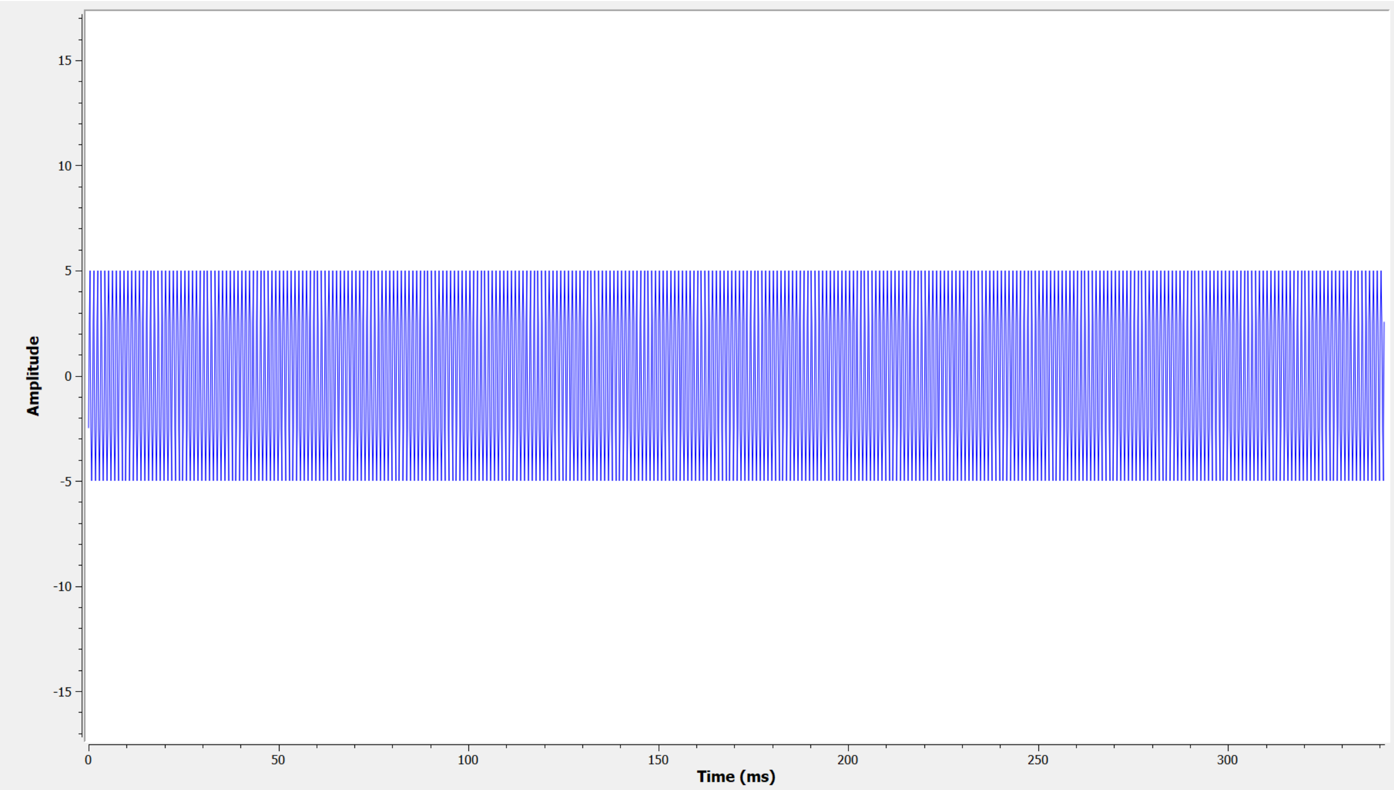
Reemplazando nuestros valores:

$$f_s = 2 \cdot 5000 = 10000\ [Hz]$$

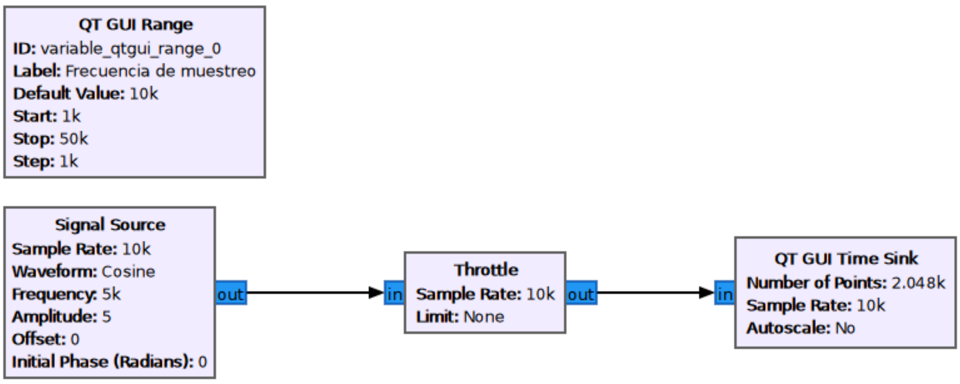
Una vez encontrada esta frecuencia, podemos resolver la consigna.

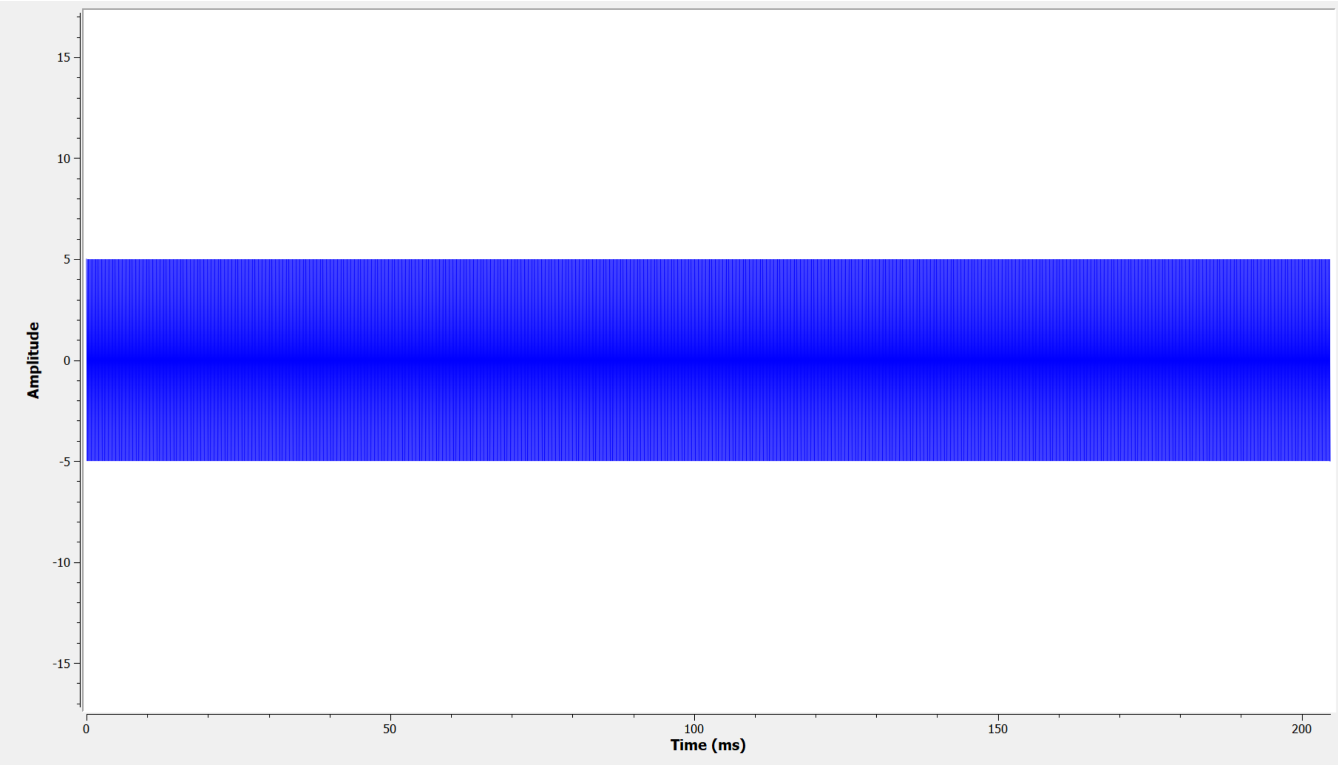
Para el caso a), nos pide utilizar una frecuencia menor a la mínima requerida por el criterio de Nyquist, es decir, $f_s < 10\ [kHz]$. Nosotros elegimos una frecuencia $f_s = 6\ [kHz]$, logrando como respuesta que aparezca aliasing, donde la señal muestreada se distorsiona.



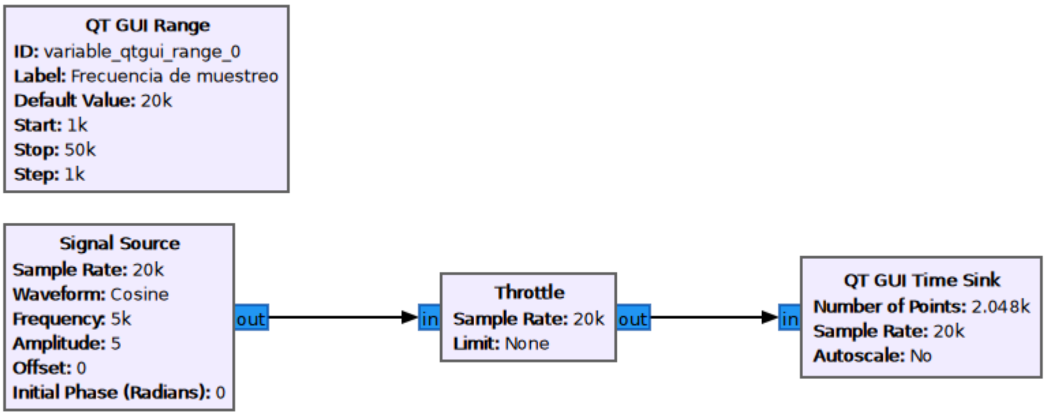


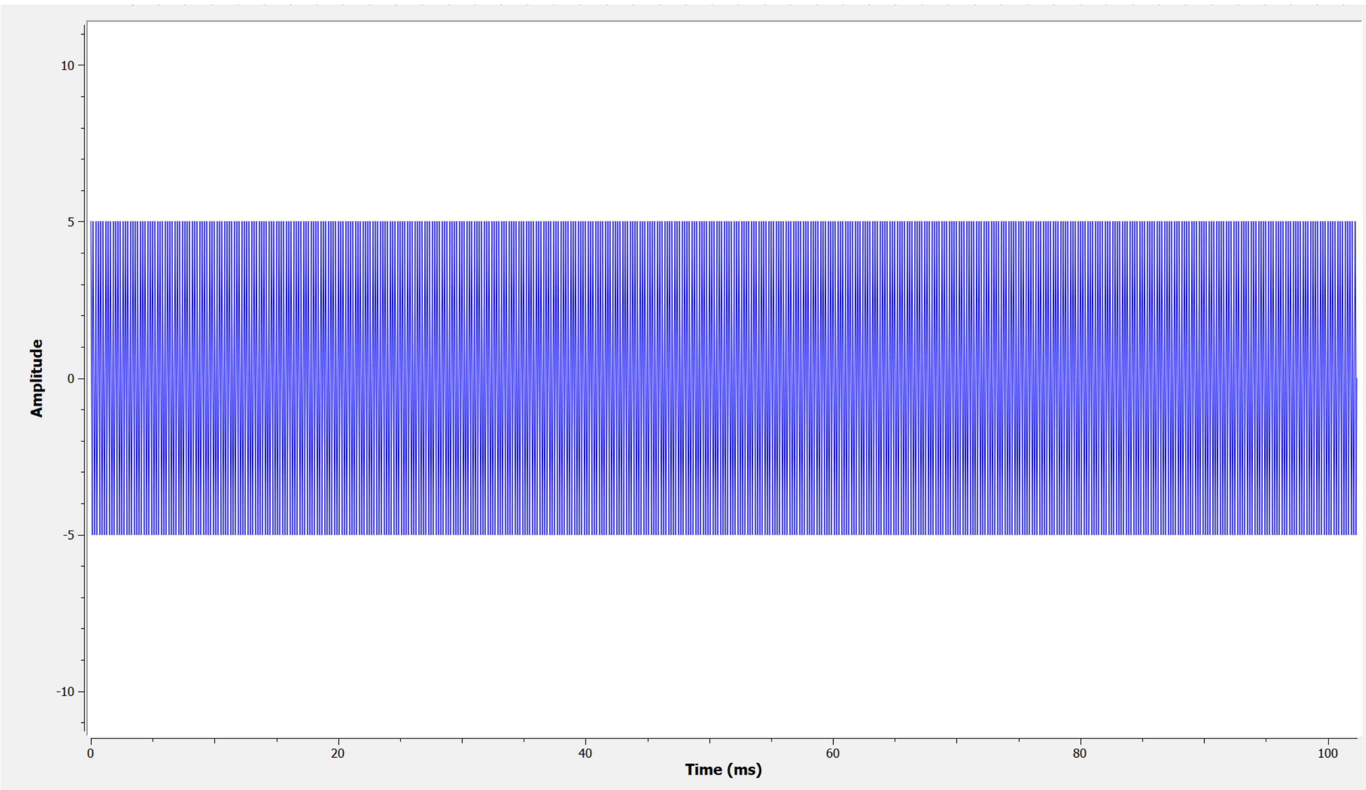
Para el caso b), nos pide utilizar una frecuencia igual a la de Nyquist, esto es, $f_s = 10\ [kHz]$. No se cumple el Teorema de Nyquist porque no es mayor, pero no se puede distinguir la forma de la señal debido a que trabajamos con una frecuencia de muestreo al límite que la de Nyquist





Para el caso c), se pide utilizar una frecuencia mayor a la de Nyquist, en este caso se utilizo $f_s = 20\ [kHz]$. Hay que tener en cuenta que, a mayor tasa de muestreo, se obtiene una mejor definición, obteniendo una reconstrucción.





10
5
0
-5
-10
Amplitude
0
20
40
60
80
100
Time (ms)

Ejercicio N°5

Una señal compleja puede descomponerse y/o representarse en coordenadas ortogonales (IQ, o bien, parte real y parte imaginaria) o también, en coordenadas polares (módulo o magnitud y fase). Ambas representaciones contienen la misma información para describir la señal.

Considere una señal compleja de tiempo continuo $x(t)$ de la forma:

$$x(t) = e^{j2\pi 1000t}$$

Utilice una $f_s = 64\ [kHz]$.

a) Considere la representación fasorial de la señal $x(t)$ ¿A qué velocidad angular se mueve el fasor?, ¿Cuántas vueltas da el fasor en el tiempo de 1 segundo?, ¿Cuánto tiempo tarda el fasor en dar una vuelta?

b) Utilice el bloque Complex to Float y grafique la parte real e imaginaria de la señal, por separado y en el tiempo. ¿Qué señal se ve en cada caso?

c) Utilice el bloque Complex to Mag Phase y grafique ambas salidas. Explique cada gráfico. ¿Cuánto vale la fase a los $10.30\ [ms]$? Dibuje el fasor en ese tiempo de $10.30\ [ms]$. ¿Cuántas vueltas dio?

- **Resolución:**

Identificando la frecuencia en la que se trabaja ($f = 1000\ [Hz]$), podemos calcular la velocidad angular del fasor, esto es:

$$\omega = 2\pi f = 2\pi \cdot 1000 = 6283{,}19\ [rad/s]$$

Teóricamente, se sabe que una vuelta equivale a un ciclo por segundo ($1\ Vuelta = 1\ [Hz]$), por lo tanto, ($1000\ Vueltas = 1000\ [Hz]$).

$$Vueltas = 1000$$

El tiempo que se tarda en dar una vuelta completa es periodo y se calcula como:

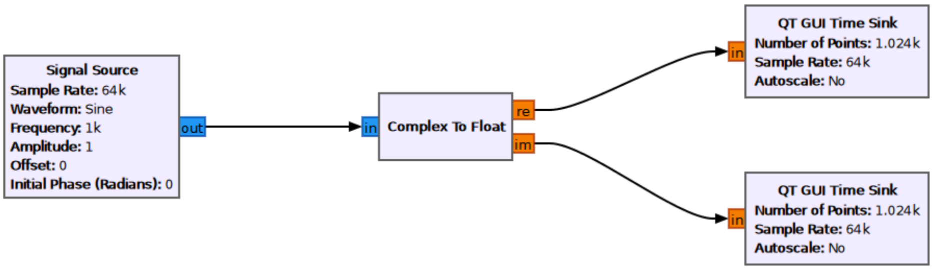
$$T = \frac{1}{f} = \frac{1}{1000\ [Hz]} = 0{,}001\ [s]$$

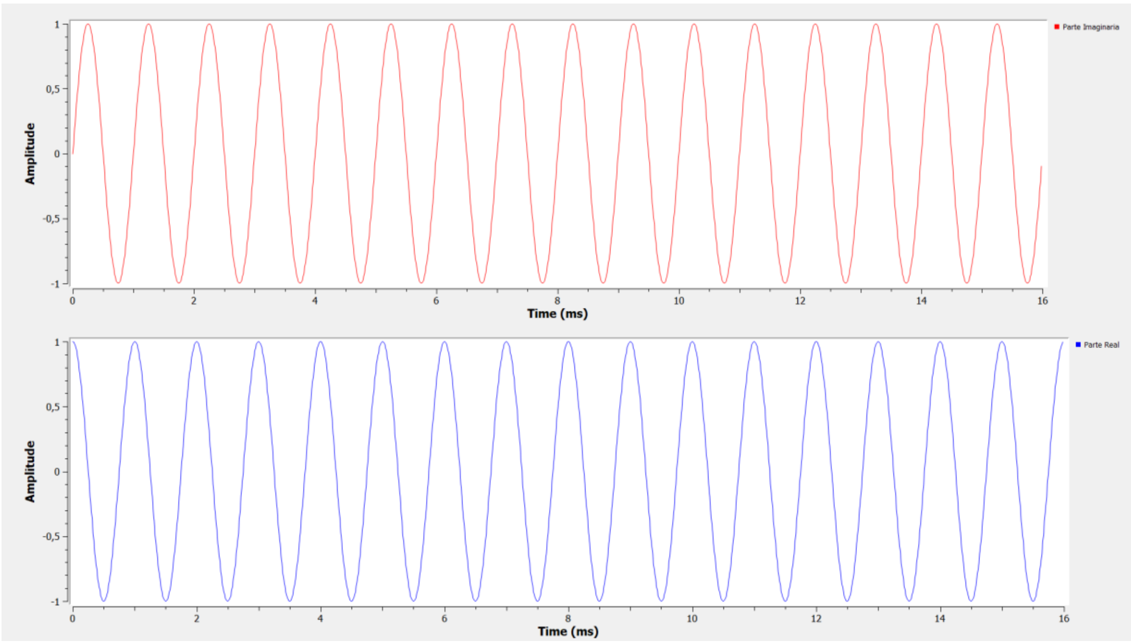
$$T = 1\ [ms]$$

Para el caso b) es necesario identificar la parte Real e Imaginaria de nuestra expresión. A través de Euler tenemos que:

$$Re\big(x(t)\big) = \cos\,(2\pi 1000t)$$

$$Im\big(x(t)\big) = \sin\,(2\pi 1000t)$$





Al graficar estas señales, se observan señales sinusoidales con la misma frecuencia, pero desfasadas $90°$ entre sí.

Para el caso c), identificamos el módulo de nuestra expresión, por lo tanto, tenemos que:

$$|x(t)| = 1$$

Identificando nuestra fase en la expresión, tenemos que:

$$\theta = 2\pi 1000t$$

Al evaluar en nuestro tiempo $t = 10.30\ [ms]$, obtenemos el siguiente resultado:

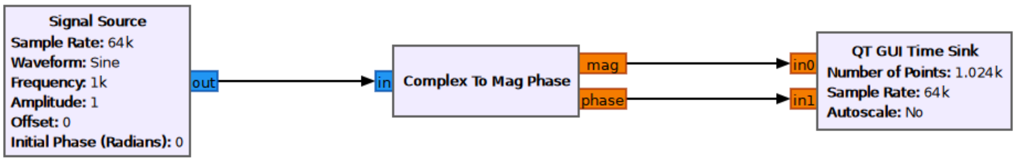
$$\theta = 2\pi \cdot 1000 \cdot (0{,}0103)$$

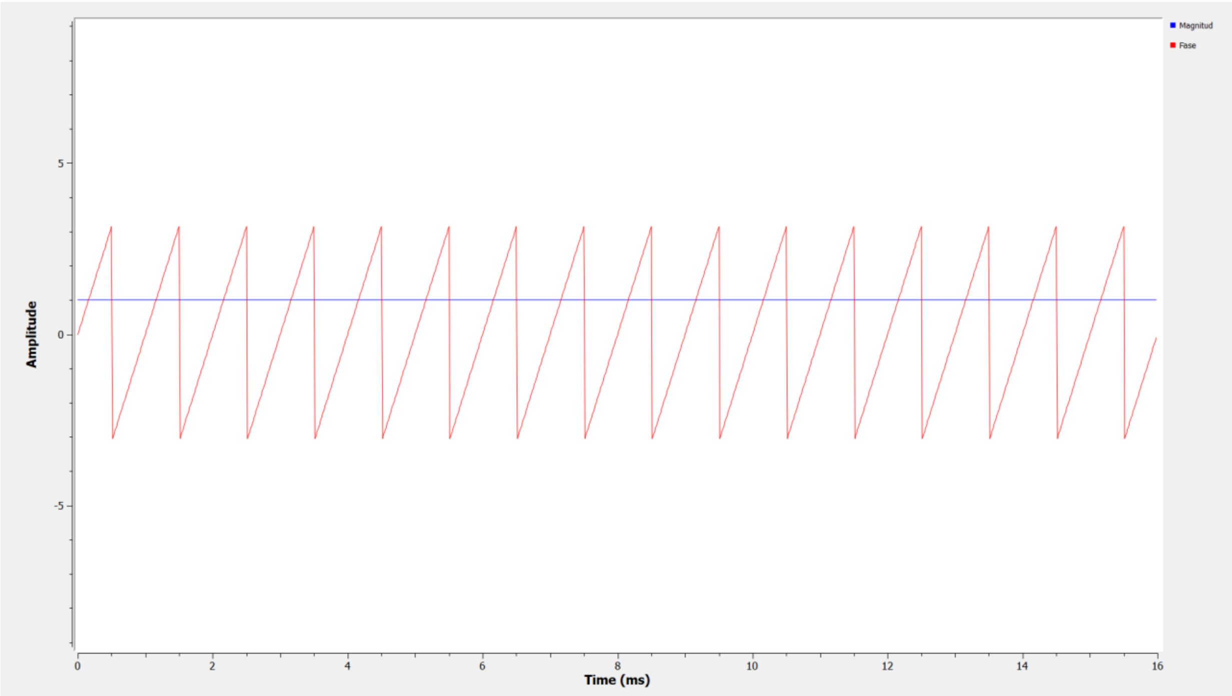
$$\theta = 64{,}72\ [rad]$$

Para obtener el número de vueltas:

$$Vueltas = \frac{\theta}{2\pi} = \frac{64{,}72}{2\pi}$$

$$Vueltas = 10.3$$



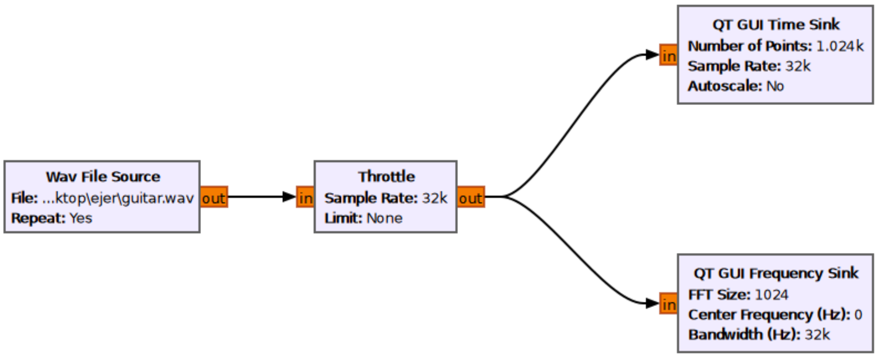


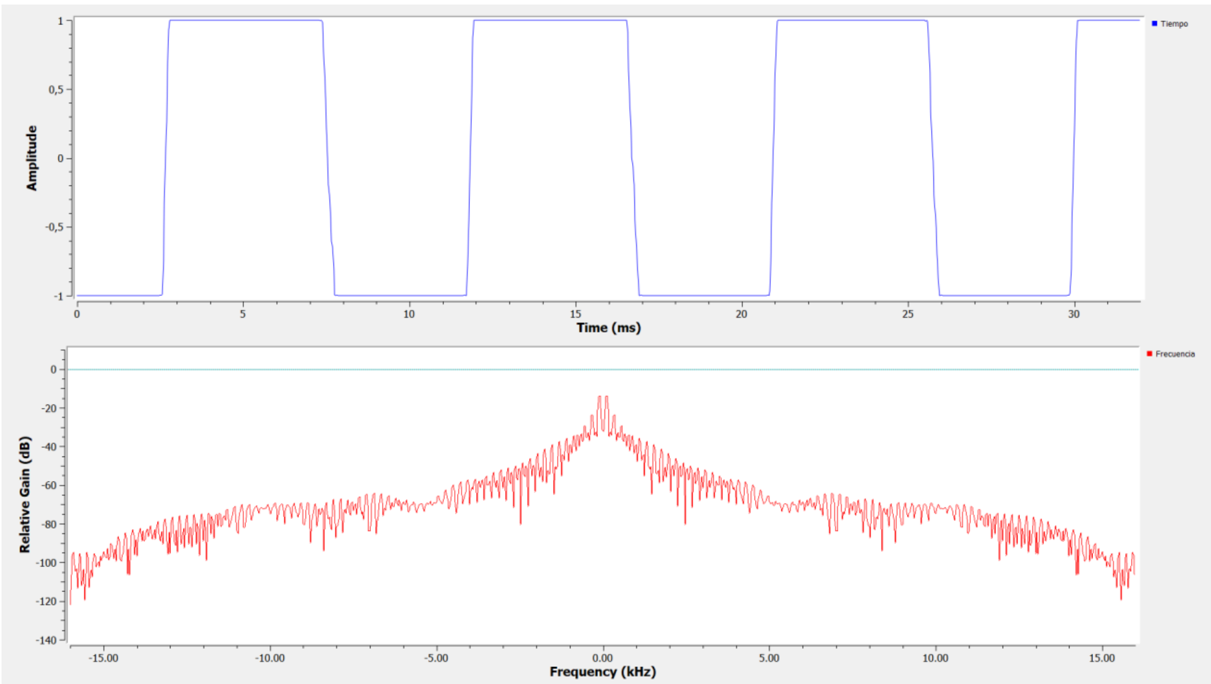
Ejercicio N°7

Se ha grabado el tono (el sonido del tono) que produce una cuerda de guitarra tocada al aire, afinada en el estándar $A4 = 440\ [Hz]$, a una tasa de $32\ [kHz]$, en el archivo guitar.wav.

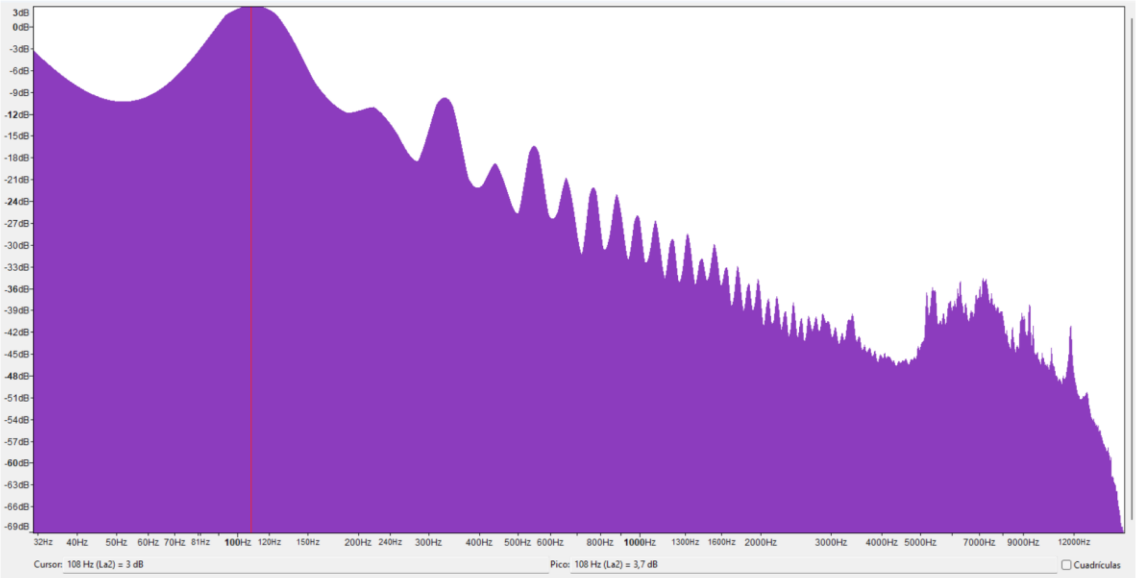
a) Usando Companion: ¿Qué forma de onda tiene el tono de la cuerda de guitarra grabada?, ¿Qué frecuencia ha identificado?

b) ¿A qué nota musical de la guitarra corresponde este tono?





Para el caso b) se utiliza el programa Audacity y se identifica el pico de frecuencia



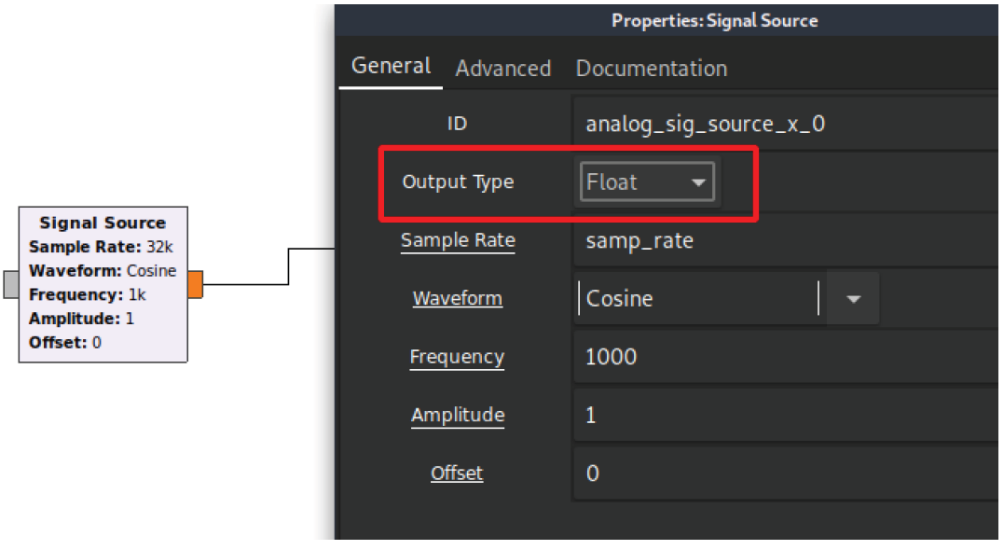
En esta imagen se identifica que la frecuencia del tono grabado (pico de frecuencia)

$$f = 108\ [Hz]$$

Una vez obtenida la frecuencia, se busca identificar su nota. La nota LA o A2 corresponde a la quinta curda y tiene una frecuencia $f = 110\ [Hz]$, por lo que podemos confirmar que la nota musical corresponde a la nota La (o A2 en caso de que se trabaje en cifrado americano).

Ejercicio N°9

Cuando generamos una señal con GNURADIO Companion, por ejemplo con el bloque Signal Source, podemos elegir el tipo de variable para la secuencia de salida (Output Type):



El tipo de variable Float, usa 32 bits para representar una muestra, que es una representación de un punto decimal de precisión simple, o también llamado float32s.

a) Responde:

  i) Esta forma de representar una muestra: ¿Introduce algún error de cuantización?

  ii) ¿Hay alguna forma de evitar el error de cuantización cuando trabajamos señales digitales?

b) Use el bloque Quantizer, para cuantizar señales. Cree, en Companion, una señal tipo seno, de $100\ [Hz]$ de frecuencia, salida Float, y con frecuencia de muestreo de $10\ [kHz]$, y amplitud 1. Compare gráficamente la señal original con la misma señal, pero cuantizada a 12 bits

  i) ¿Observa diferencias entre ambas señales?

  ii) ¿Cuántos niveles de cuantización son 12 bits?

c) Ahora proponga una cuantización de 4 niveles. ¿Cuántos bits se necesitan para lograr 4 niveles? Compare gráficamente la señal sin cuantizar con la cuantizada, en Companion.

d) Calcule de manera teórica la relación señal – ruido de cuantización (SQNR) para señales sinusoidales que usan 4 niveles de cuantización. También calcule el SQNR en dB.

   i) ¿Cuánto aumenta el SQNR por cada bit que se agregue en el cuantizador?, ¿Por qué?

e) Vamos a desestimar el error de cuantización introducido por usar variables Float. Considerando una cuantización de 4 niveles, estime el error de cuantificación $e_q[n]$ para una señal sinusoidal usando bloques del Companion, y grafique $e_q[n]$ usando el bloque QT GUI Time Sink.

f) Utilizando el bloque Python Block, reemplace el código por el siguiente código:

```
import numpy as np
from gnuradio import gr

class blk(gr.sync_block):

  def __init__(self):
    gr.sync_block.__init__(
      self,
      name='Potencia media',
      in_sig=[np.float32],
      out_sig=[np.float32]
    )

  def work(self, input_items, output_items):
    prom_cuadratico = np.mean(input_items[0] **2)
    prom_vec = np.repeat(prom_cuadratico,len(output_items[0]))
    output_items[0][:] = prom_vec
    return len(output_items[0])
```

   i) Grafique $P_x[n]$ y $P_q[n]$ utilizando el bloque QT GUI Time Sink.

   ii) Compute y grafique la relación SQNR en Companion, usando el bloque Potencia media.

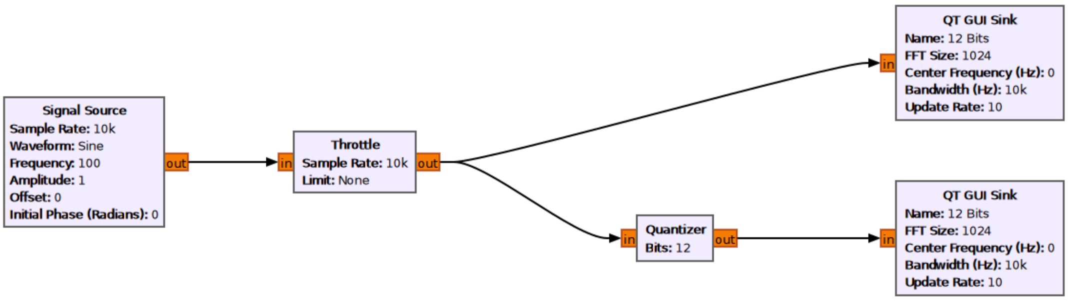
   iii) Compare le relación SQNR que obtuvo teóricamente, con la relación SQNR que obtuvo en Companion. ¿A que puede deberse las diferencias en los valores?

- **Resolución:**

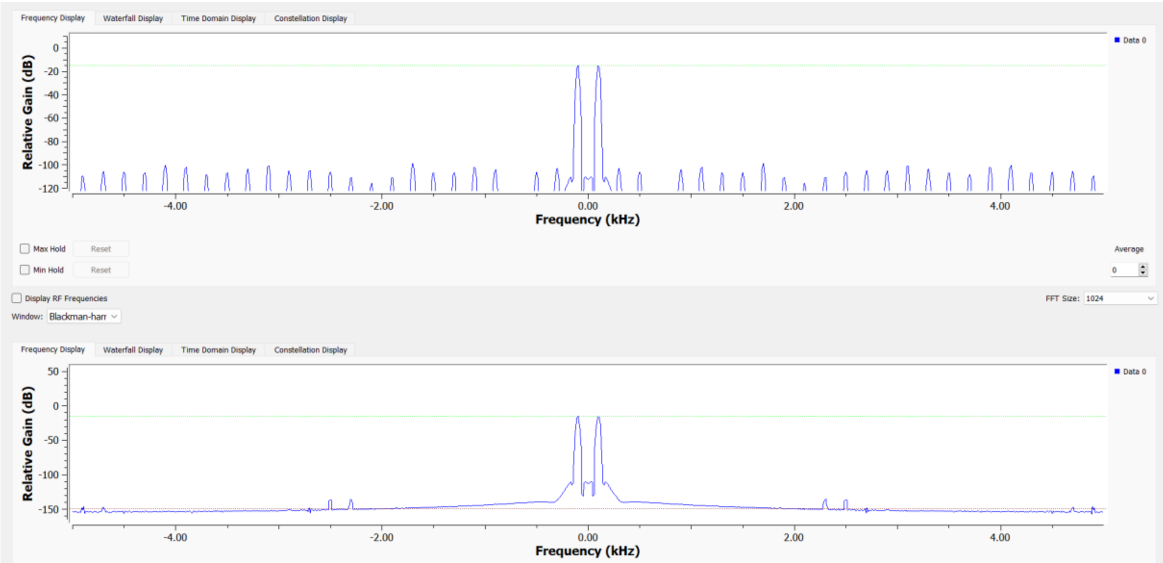
Para el caso a), podemos afirmar que al utilizar float32 si se presenta un error. Esto ocurre cuando no podemos representar todos los valores posibles debido a la limitación de la cantidad de bits disponibles.

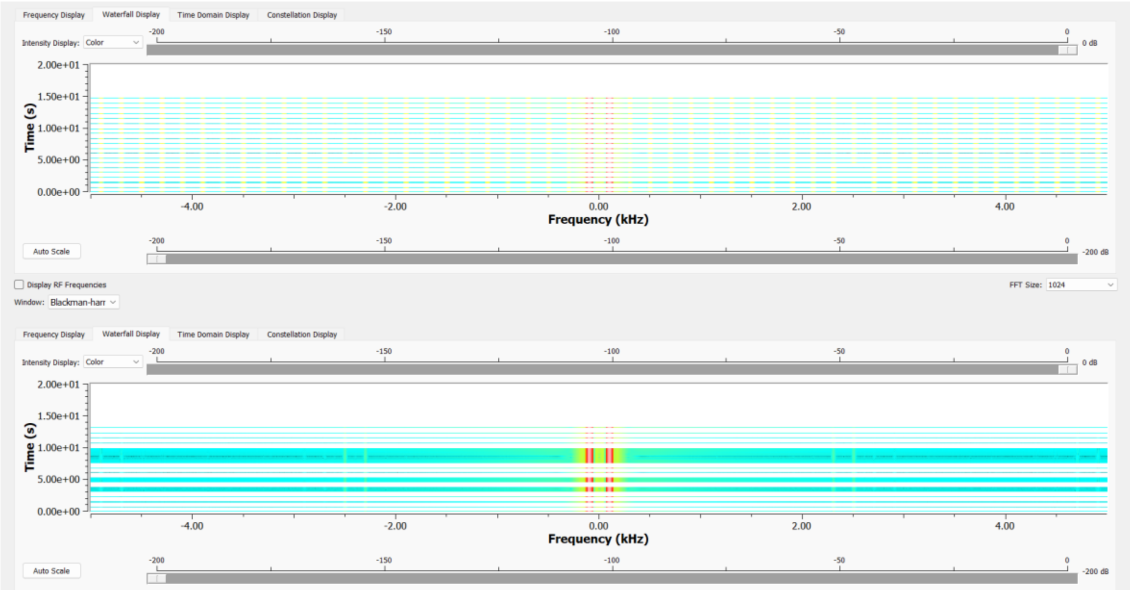
Cuando trabajamos con float32 no podemos evitar este error de cuantización, pero si lo podemos reducir. Esto se realiza a través de la utilización de filtros o técnicas de redondeo.

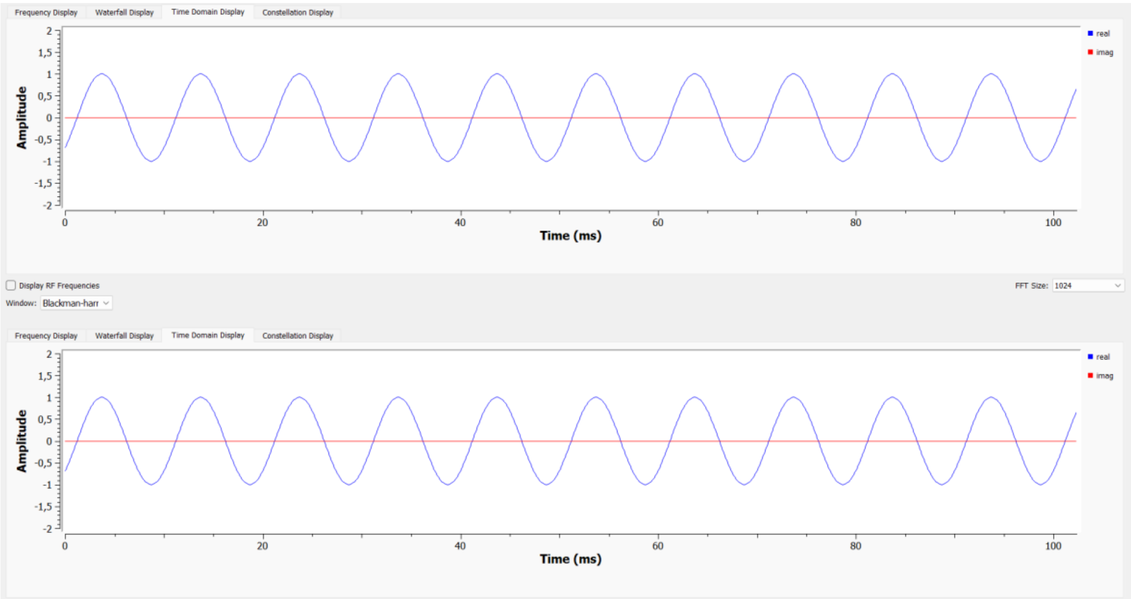
Para el caso b) se nos pide compara la señal original con una cuantizada a 12 bits.

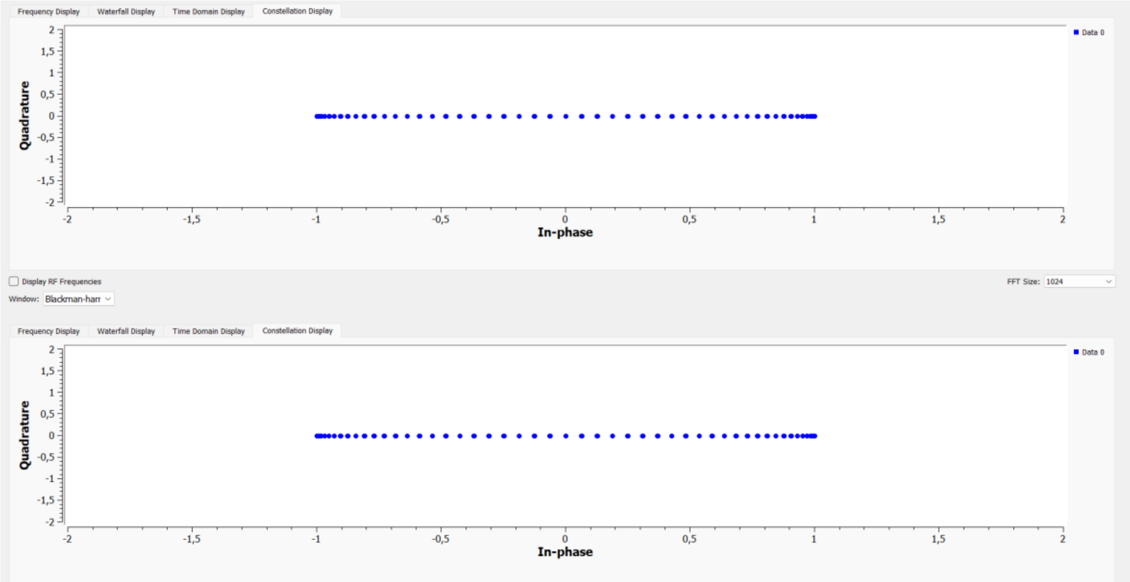


Una vez ejecutado el programa, la gráfica inferior corresponde a nuestra señal original, mientras que la superior corresponde a la señal cuantizada a 12 bits.




Frequency Display
Waterfall Display
Time Domain Display
Constellation Display
Intensity Display: Color
Time (s)
Frequency (kHz)
Auto Scale
Display RF Frequencies
Window: Blackman-harr
FFT Size: 1024


Frequency Display
Waterfall Display
Time Domain Display
Constellation Display
Amplitude
Time (ms)
real
imag
Display RF Frequencies
Window: Blackman-harr
FFT Size: 1024



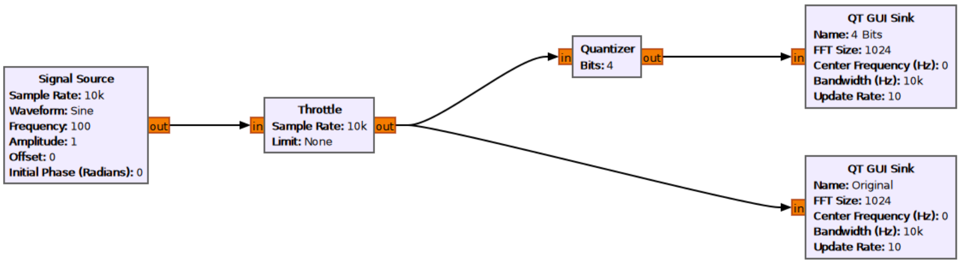
Cuando cuantizamos la señal, no puede representar exactamente los valores de la señal original en todos los puntos. Debido a esto, las señales presentan diferencias. Además de esto, se presenta un error de cuantización, debido a la limitación de 12 bit.

Si utilizamos 12 bits, vamos a tener 4096 niveles de cuantización, esto es:

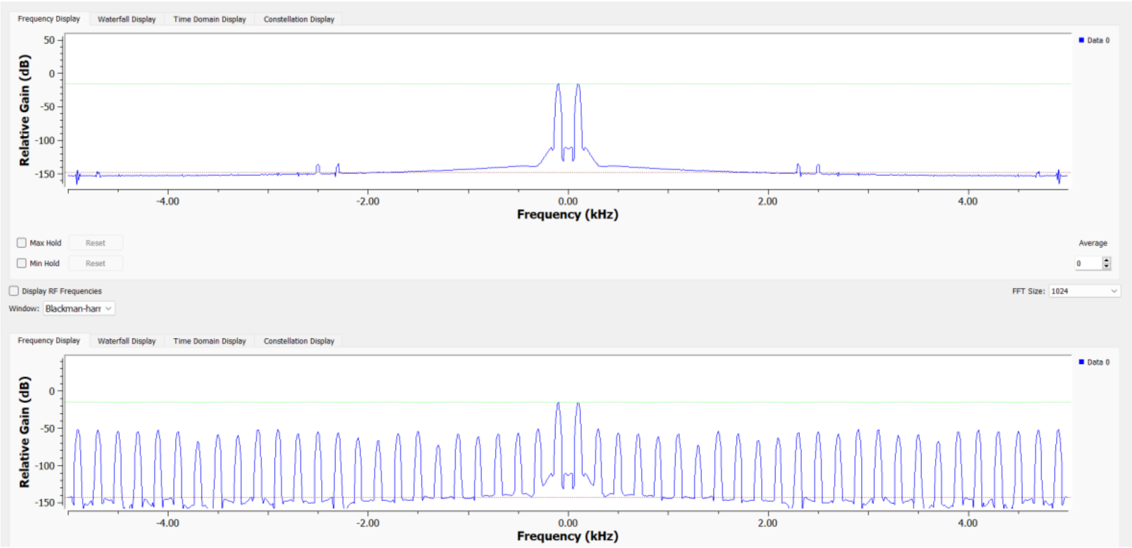
$$2^{12} = 4096\ Niveles\ de\ cuantización$$

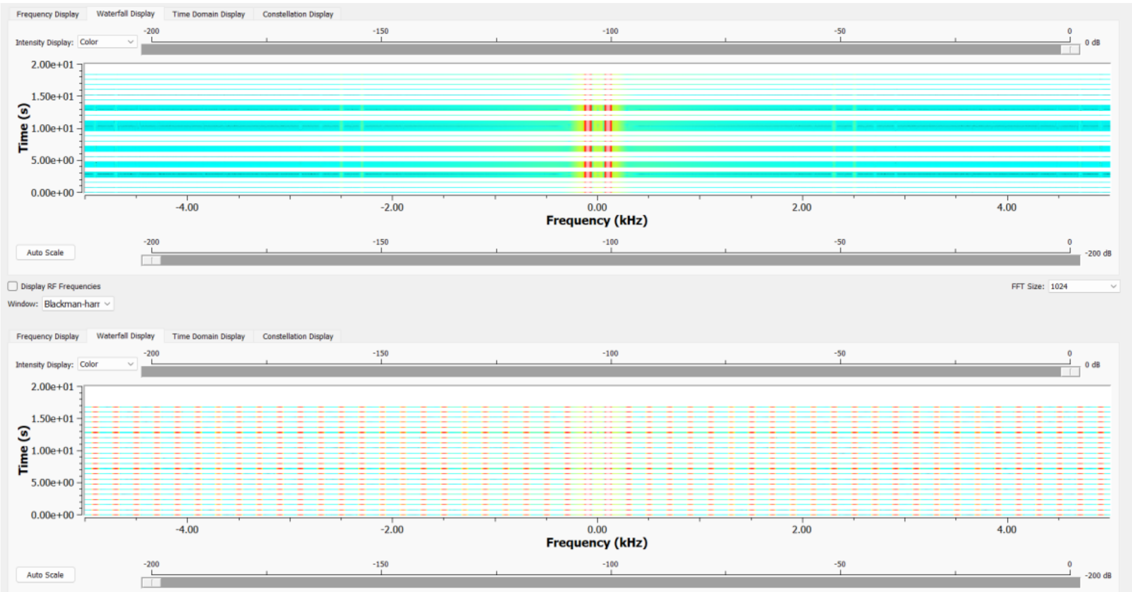
Para el caso c), se nos pide realizar una cuantización de 4 niveles, es decir, 2 bits.

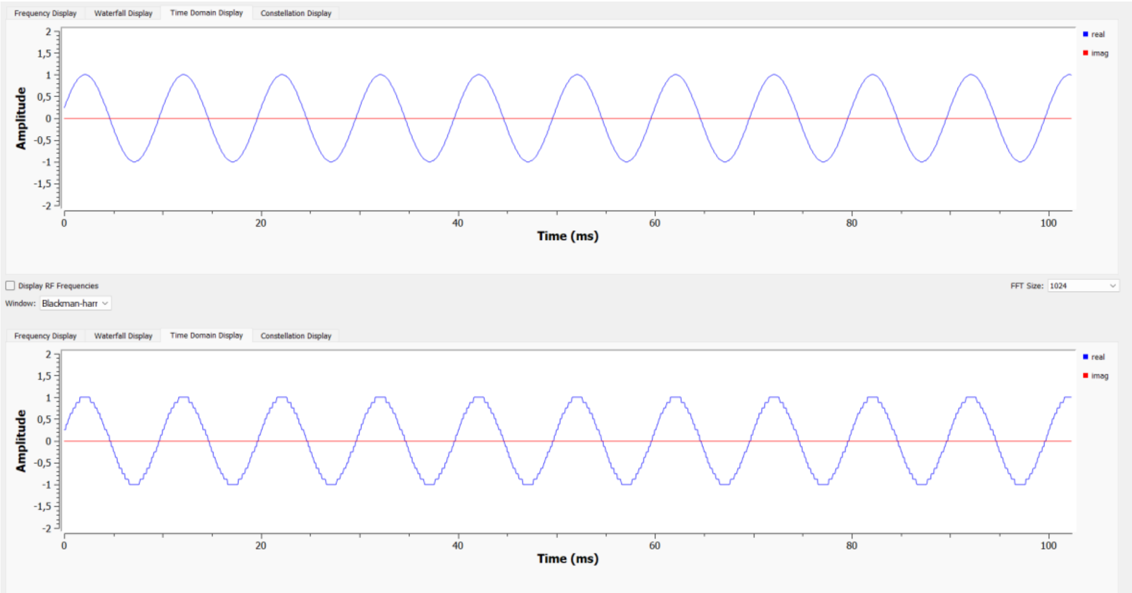
$$2^{2} = 4\ Niveles\ de\ cuantización$$

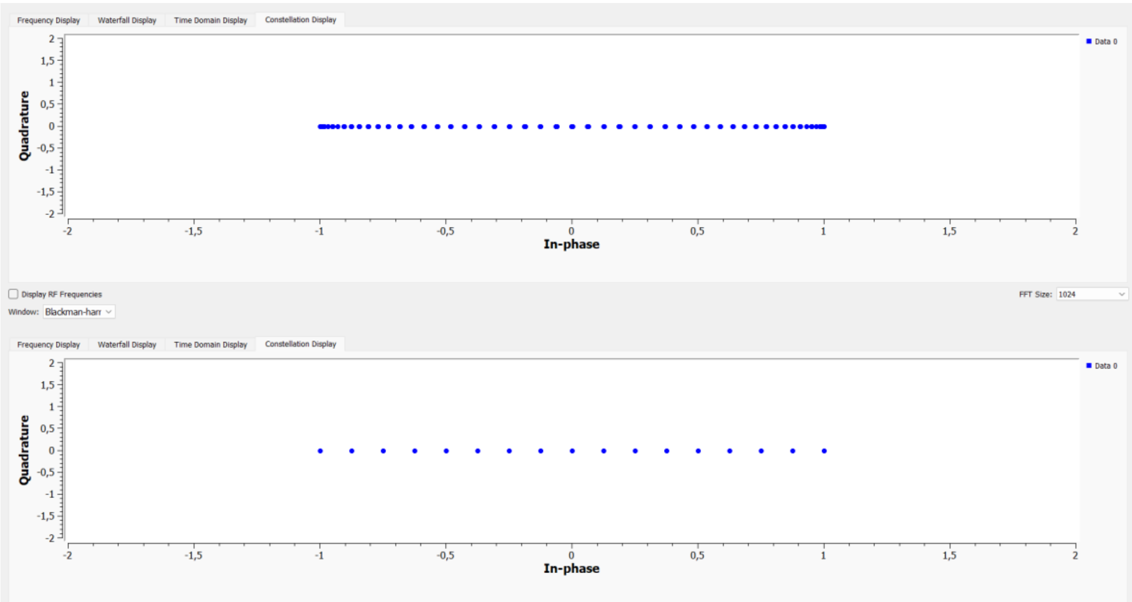


Una vez ejecutado el programa, en la gráfica superior se observa la señal original, mientras que en la inferior se observa la señal cuantizada a 2 bits.


Frequency Display
Waterfall Display
Time Domain Display
Constellation Display
Relative Gain (dB)
Frequency (kHz)
Data 0
Max Hold
Min Hold
Reset
Average
Display RF Frequencies
Window: Blackman-harr
FFT Size: 1024


Frequency Display
Waterfall Display
Time Domain Display
Constellation Display
Intensity Display: Color
Time (s)
Frequency (kHz)
Auto Scale
Display RF Frequencies
Window: Blackman-harr
FFT Size: 1024





Para el caso d), debemos calcular la relación señal – ruido de cuantización (SQNR) para señales que utilizan 4 niveles de cuantización, esto es:

$$SQNR = \frac{Potencia\ señal}{Potencia\ ruido\ de\ cuantización} = \frac{P_s}{P_r}$$

La señal sinusoidal de amplitud máxima (A) tiene una potencia promedio:

$$P_s = \frac{A^2}{2}$$

Mientras que para la señal de ruido con un paso Δ:

$$P_r = \frac{\Delta^2}{12}$$

Para un cuantizador con $L = 2^n$ niveles, el paso es:

$$\Delta = \frac{2A}{L} = \frac{2A}{2^n}$$

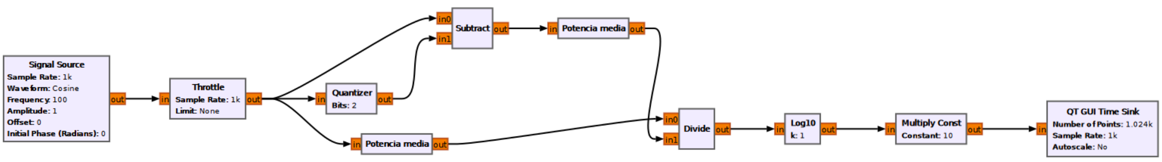
Reemplazando estos términos en la ecuación de $SQNR$

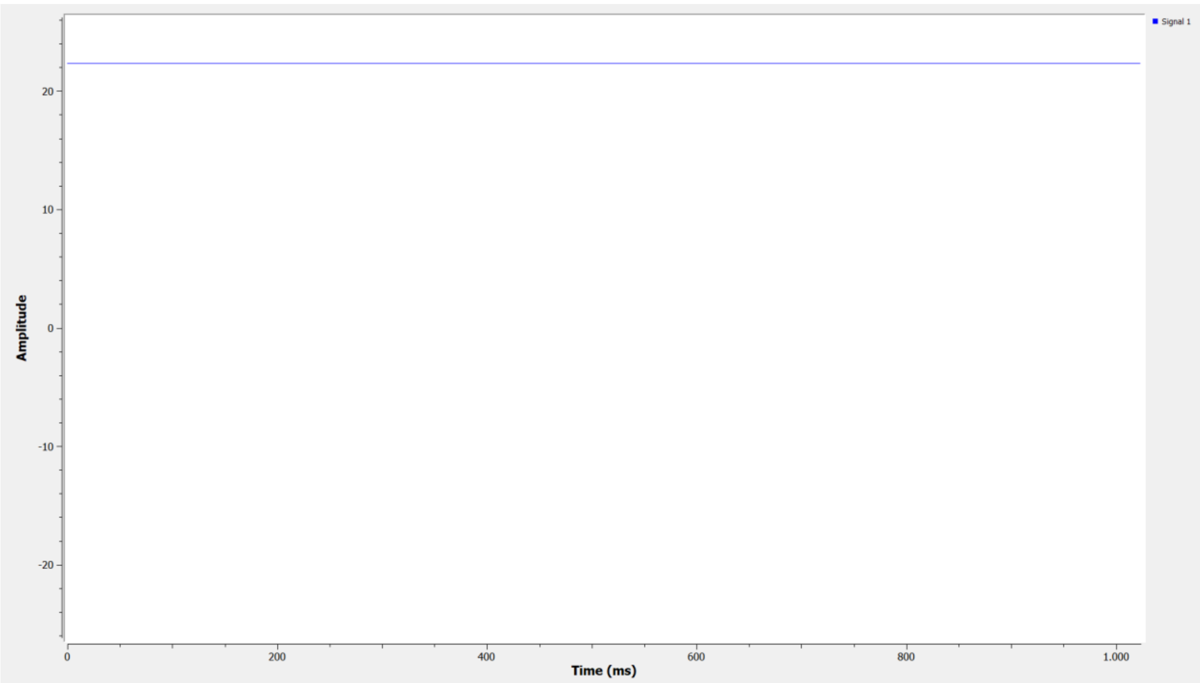
$$SQNR = \frac{\frac{A^2}{2}}{\frac{\Delta^2}{12}} = \frac{\frac{A^2}{2}}{\frac{(2A/2^n)^2}{12}} = \frac{\frac{A^2}{2}}{\frac{4A^2}{12 \cdot 2^{2n}}} = \frac{\frac{1}{2}}{\frac{4}{12 \cdot 2^{2n}}} = \frac{1}{2} \cdot \frac{12 \cdot 2^{2n}}{4} = \frac{3 \cdot 2^{2n}}{2}$$

Reemplazando $n = 2$, tenemos:

$$SQNR = \frac{3 \cdot 2^{2 \cdot 2}}{2} = 24$$

$$SQNR = 24$$

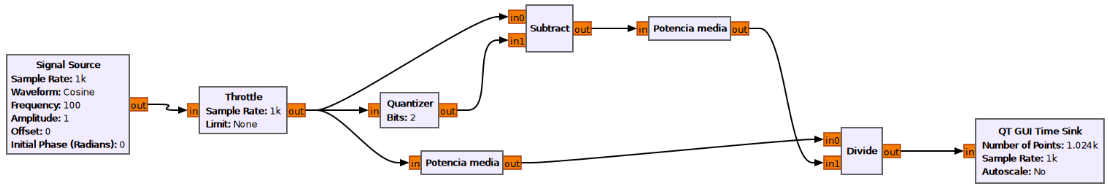


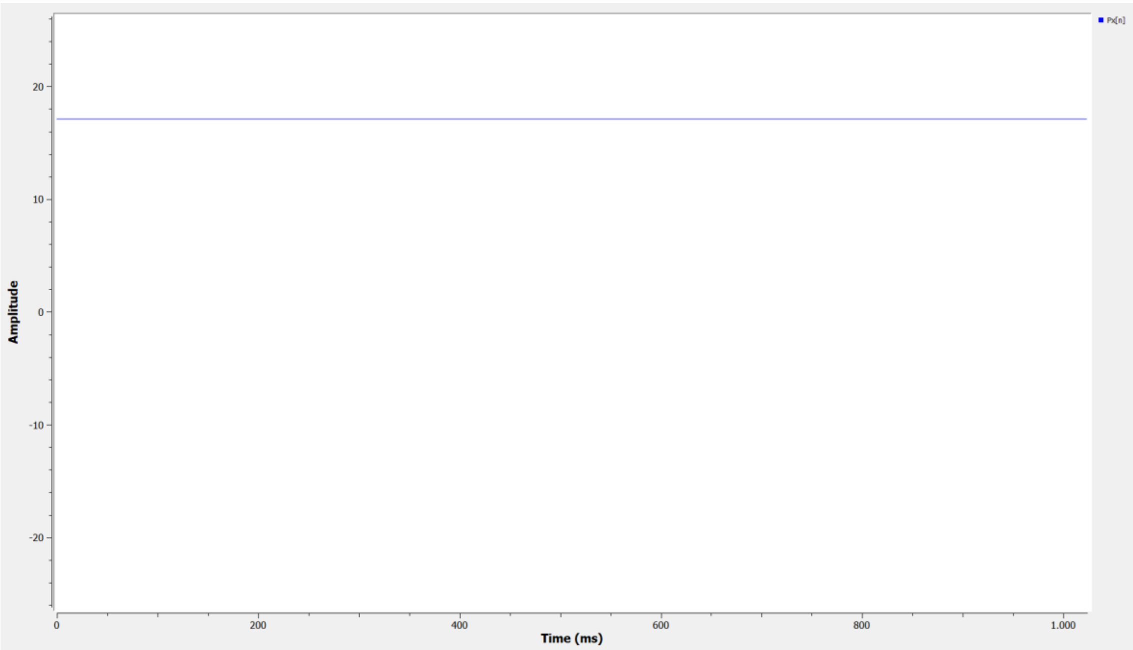


Haciendo este cálculo en dB:

$$SQNR_{dB} = 10\log_{10}(SQNR) = 10\log_{10}(24)$$

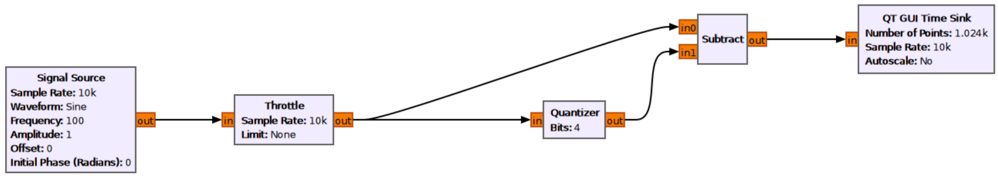
$$SQNR_{dB} = 13{,}8\ [dB]$$

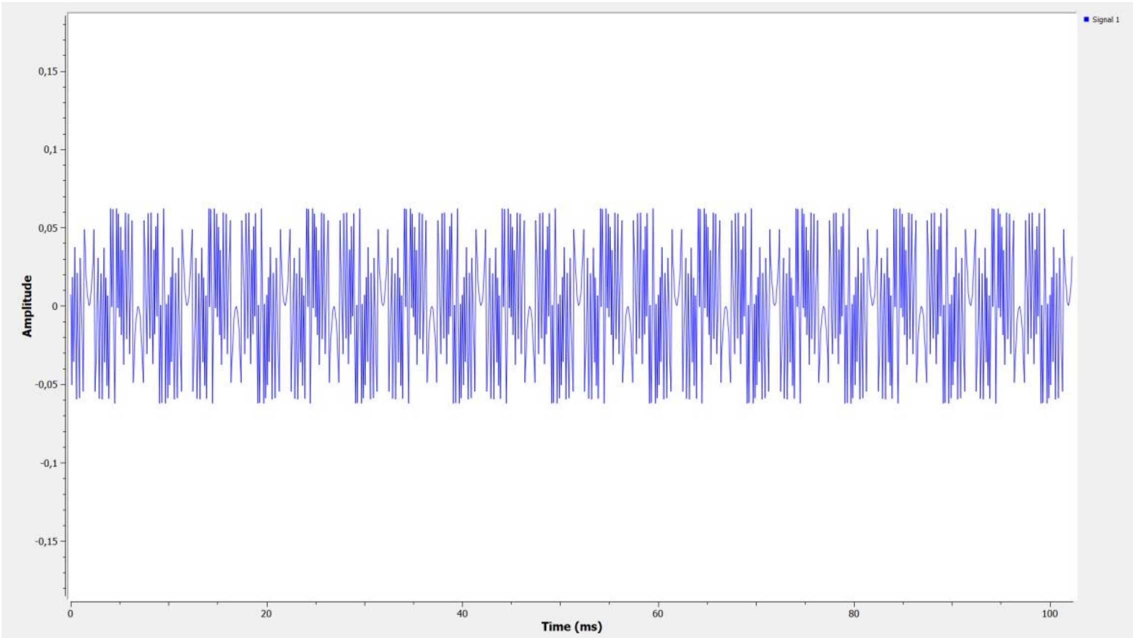




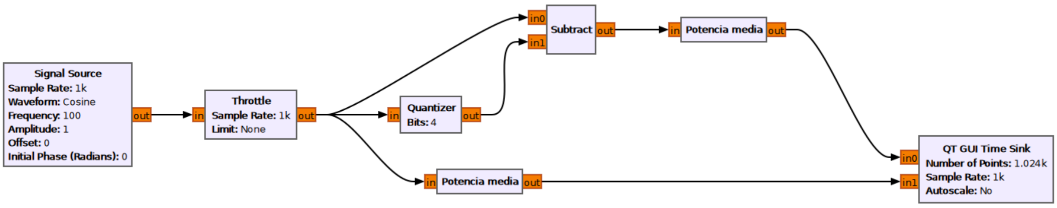
Cada bit adicional nos da el doble de niveles de cuantización, haciendo cálculos para varios bits adicionales podemos verificar que se agrega un aproximado de $6\ [dB]$ en el SQNR.

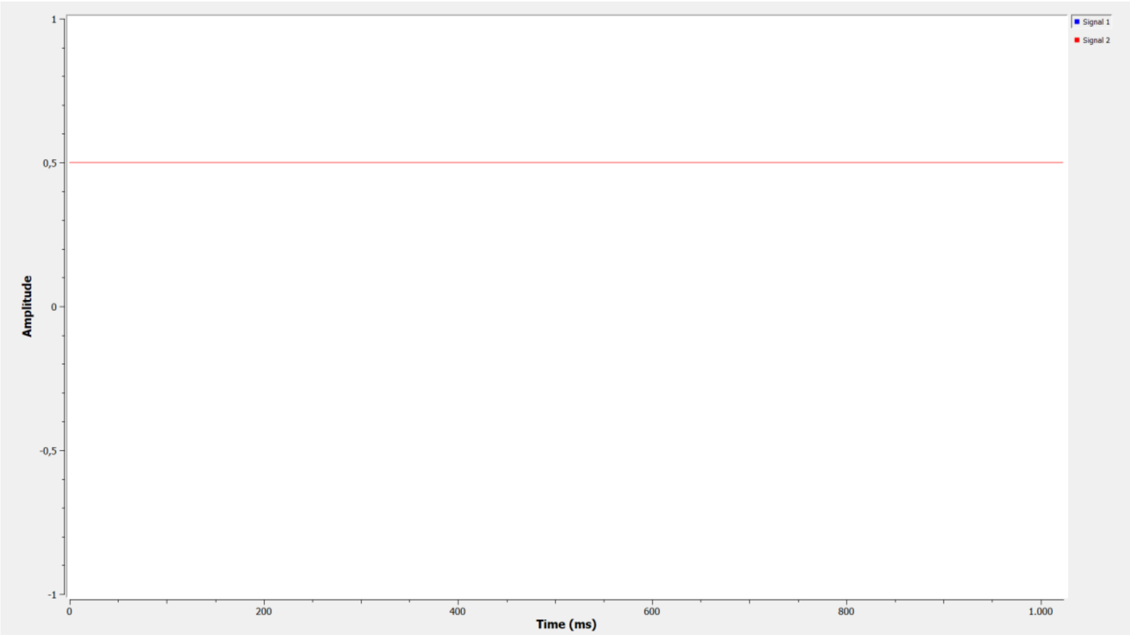
Para el caso e), graficamos el error de cuantificación $e_q[n]$, considerando una cuantización de 4 niveles.





Para el caso f) se modifica el Python Block con el código brindado y se realizan las gráficas de $P_q[n]$ y de $P_x[n]$.





Signal 1
Signal 2
Amplitude
1
0,5
0
-0,5
-1
0
200
400
600
800
1.000
Time (ms)

Ejercicio N°11

Cree un tipo seno de $300\ [Hz]$ y frecuencia de muestreo de $48\ [kHz]$, usando un creador de tonos online.

a) Reproduzca la señal en el Companion.

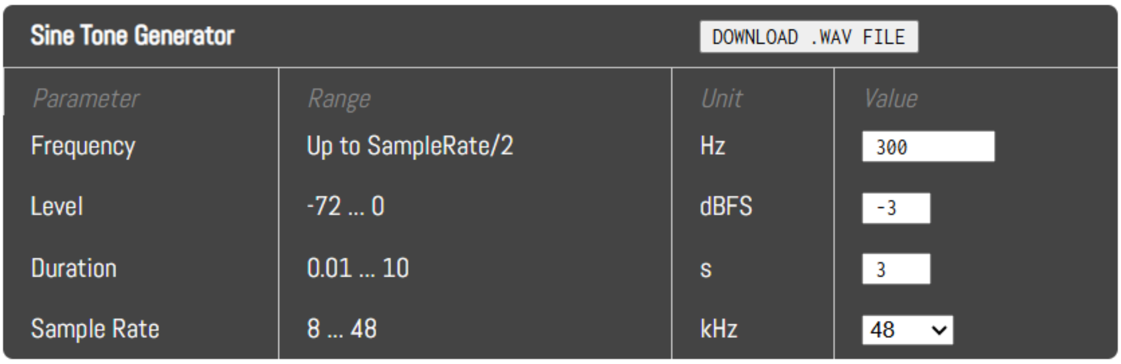
b) Utilice una combinación de bloques de GNURADIO Companion para diesmado e interpolación tal que pueda llevar la señal a una tasa de muestreo de $32\ [kHz]$. Verifique gráficamente en Companion.

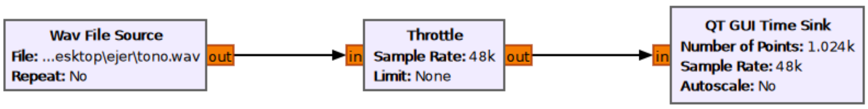
c) ¿Qué bloque debe ir primero (interpolación o diezmado) y por qué?

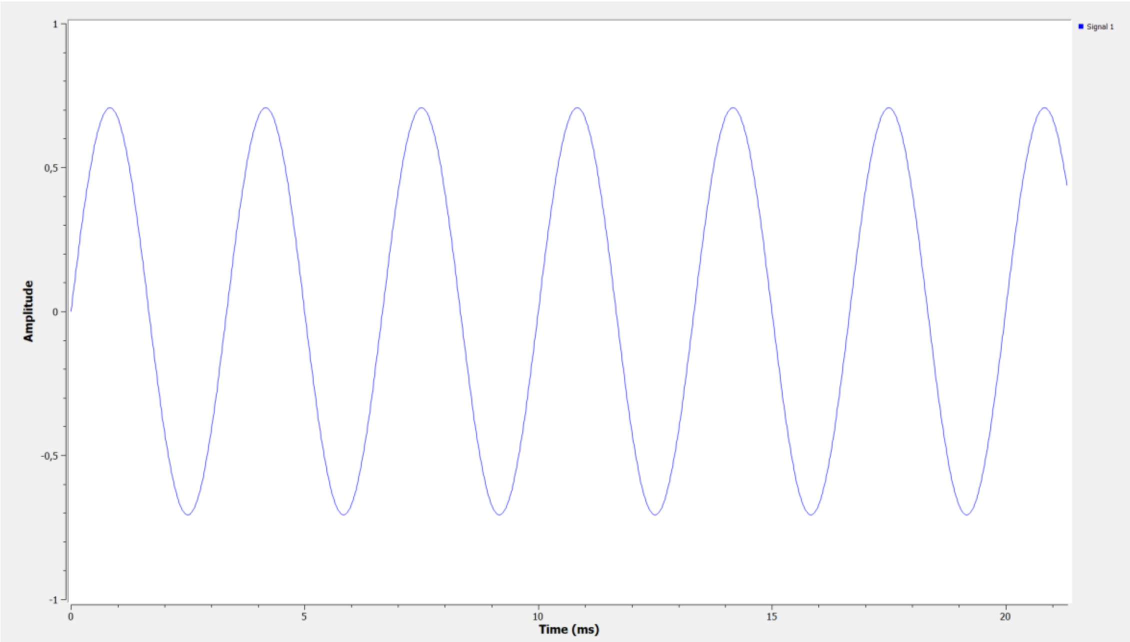
d) Reemplace la implementación por un bloque Rational Resampler.

- **Resolución:**

Para el caso a) se accede al creador de tonos online y se ingresa las configuraciones que nos pide el ejercicio para reproducirlo en GNU.



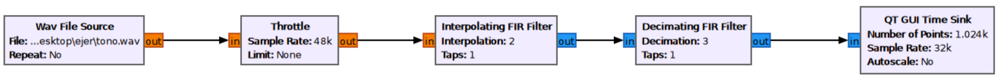


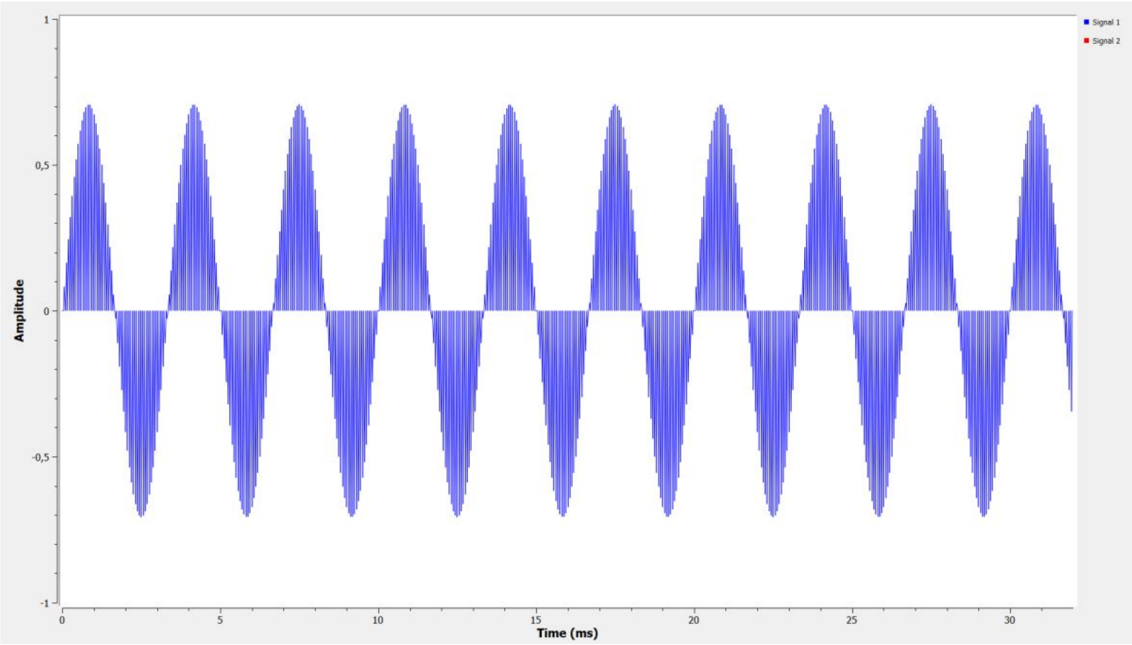


Para el caso b), comenzamos simplificando la relación $32\,[kHz]$ y $48\,[kHz]$, obteniendo los valores de interpolación y diezmado, esto es:

$$\frac{32000}{48000} = \frac{2}{3}$$

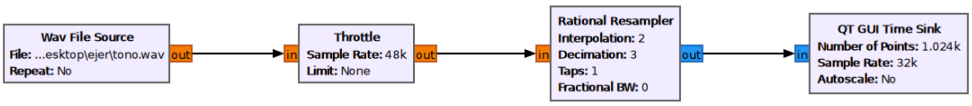
Donde la interpolación corresponde a 2 y el diezmado a 3. En GNU a la interpolación la hacemos a través del bloque Interpolating FIR Filter y el diezmado a través de Decimating FIR Filter.



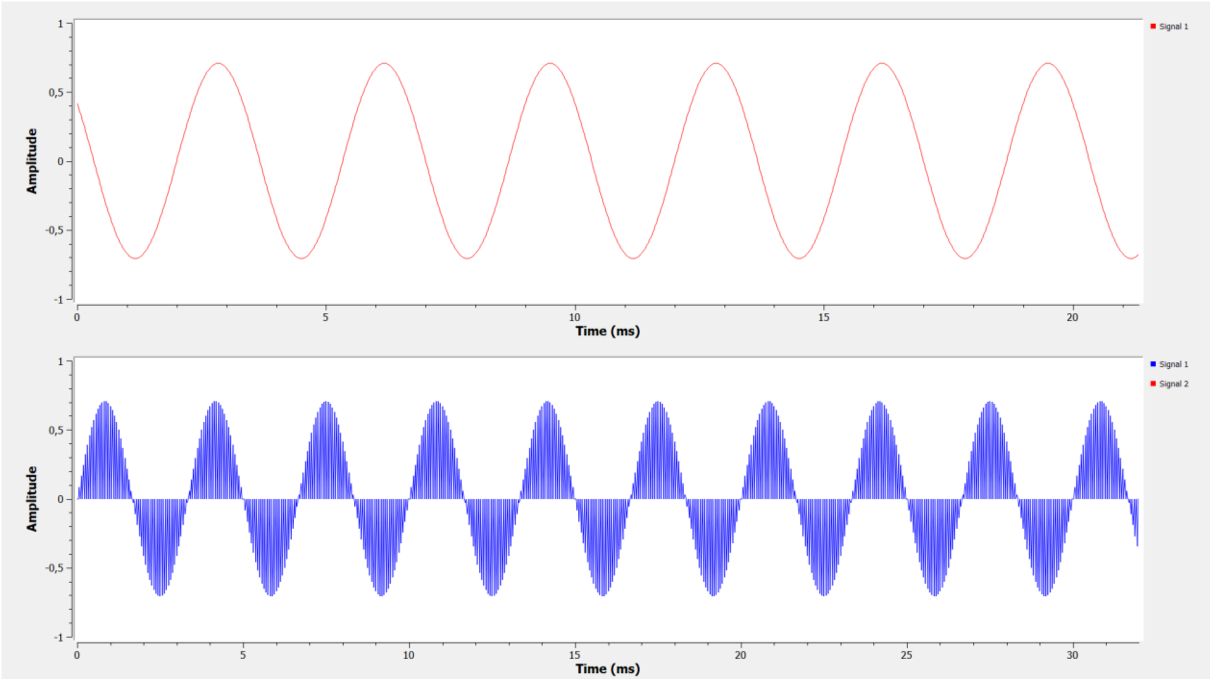


Para el caso c), tiene que ir la interpolación antes que el diezmado. Esto se debe a que, si lo hacemos al revés, podemos perder información (Aliasing).

En el caso d), se sustituye los bloques Interpolating FIR Filter y Decimating FIR Filter por el bloque Rational Resampler.



Para ver las diferencias entre las señales en los incisos a) y b) se las puede comparar, donde la señal azul corresponde a la señal original, mientras que la señal roja corresponde a la señal utilizando diezmado e interpolación.



Amplitude
Time (ms)
Signal 1
Amplitude
Time (ms)
Signal 1
Signal 2

Ejercicio N°12

Se necesitan obtener muestras de una RTL – SDR, a una frecuencia de muestreo de $1.2\ [MHz]$, para procesar cierta información. Considerando los dos canales I-Q:

a) Calcule en bitstream, o tasa de bis por segundo.

b) ¿Cuántos niveles de cuantización tienen las muestras que se toman?

c) ¿Cuál es la frecuencia de muestreo máxima que soporta la SDR?

- **Resolución:**

Para el caso a), utilizamos la siguiente ecuación:

$$Tasa\ de\ bits = f_s \cdot Bits\ por\ muestra \cdot \text{Número}\ de\ canales$$

$$Tasa\ de\ bits = 1.2\ [MHz] \cdot 8 \cdot 2$$

$$Tasa\ de\ bits = 19{,}2\ [Mbps]$$

Para el caso b), sabemos que la cantidad de niveles es $2^{bits\ por\ muestra}$ y en este caso tenemos 8 bits por muestra, es decir:

$$Niveles = 2^8 = 256$$

Para el caso c), investigamos en la web de GNU Radio (https://wiki.gnuradio.org/index.php?title=RTL-SDR_FM_Receiver) y deducimos que la frecuencia de muestreo se puede configurar hasta $3.2\ [MHz]$ pero tendremos pérdidas. Mientras que la frecuencia máxima de muestreo que soporta el RTL – SDR sin pérdidas significativas será aproximadamente de $2.56\ [MHz]$.