Colocación de las preguntas en cada parte del informe:

**Análisis del Sps:**

g. Se supone que una señal binaria aleatoria de forma rectangular tiene un ancho de banda infinito, ¿coincide esto con lo observado en GNU Radio y por qué?

o. ¿Para qué caso de Sps la PSD de una señal binaria aleatoria bipolar es similar a la PSD de ruido blanco?

**Análisis de Ruido:**

f. Se supone que el ruido blanco tiene un ancho de banda infinito, ¿coincide esto con lo observado en GNU Radio?, ¿por qué?

**Análisis de Imagen RGB:**

**Análisis de Audio:**

**Contraste de Audio e Imagen:**

c. ¿Por qué razón la PSD de las señales binarias que provienen de una señal de audio es diferente a la que proviene de una foto siendo ellas igualmente señales binarias bipolares de forma rectangular?

**Preguntas de Control:**

* **Funcionamiento del sistema:**

**- Preguntas Extras:**

k. ¿Qué pasaría si en el bloque “Unpack K Bits” se configura el parámetro K como 16?

**Graficas:**

Cuando interpolas con las muestras **1,1,1,1,2,1,1,1,1**, el valor "2" en el centro de la secuencia crea un cambio repentino o "pico" en la señal. Este cambio brusco interrumpe la continuidad de los valores "1", lo que ocasiona que la señal final tenga un salto visible en ese punto.

Este tipo de variación rápida provoca que se añadan **frecuencias más altas** en la señal, ya que, en el procesamiento de señales, los cambios abruptos en el tiempo tienden a generar componentes de alta frecuencia. El resultado es una señal que no es completamente suave y presenta una pequeña distorsión o "sobresalto".

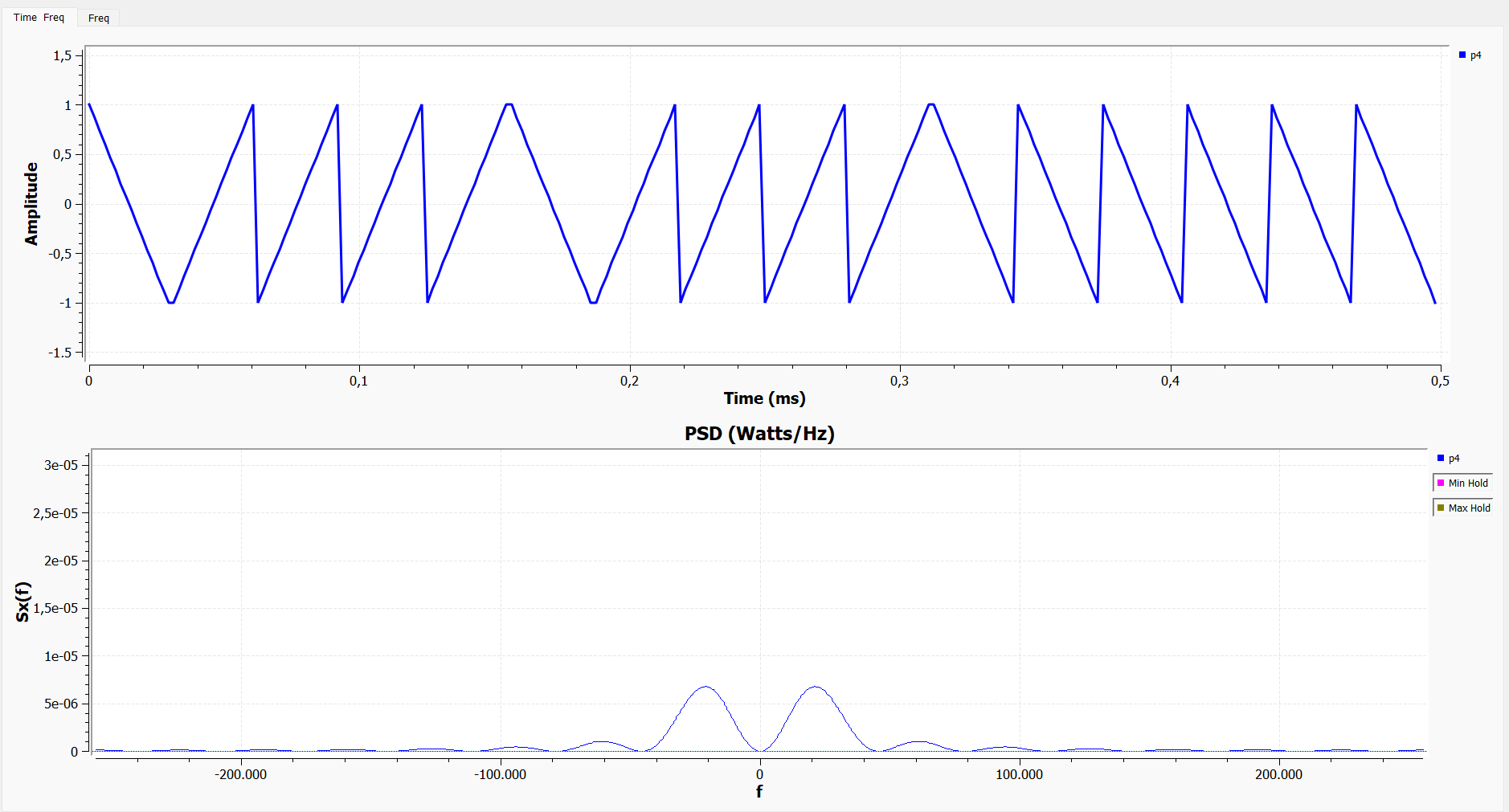
En resumen: la muestra "2" rompe la uniformidad de los valores "1" y provoca un cambio brusco en la señal, lo que genera efectos no deseados en su comportamiento.

Es como multiplicar con la señal, ya que no es un valor fijo o suave, sino con cambios repentinos que se ven en la señal.

|  |  |  |  |
| --- | --- | --- | --- |
| Señal | Valor de [h] | Constant Source | Interpolating FIR |
| Dientes de Cierra | *np.array*  *([-1, -0.866, -0.733, -0.6,*  *-0.466, -0.333, -0.2, -0.066,*  *0.0666, 0.2, 0.3333, 0.4666,*  *0.6, 0.7333, 0.8666, 1])* | -500m | 16 |
| Unipolar RZ | np.array  ([0.5, 0.5, 0.5, 0.5,  0.5, 0.5, 0.5, 0.5,  0, 0, 0, 0,  0, 0, 0, 0]) | 0 | 16 |
| Manchester NRZ | np.array  ([1, 1, 1, 1,  1, 1, 1, 1,  -1, -1, -1, -1,  -1, -1, -1, -1]) | -500m | 16 |
| OOK | np.sin(np.linspace(0, 4\*np.pi, 32)) | 0 | 32 |
| BPSK | np.sin(np.linspace(0, 8\*np.pi, 64)) | -500m | 64 (por resolución) |
| Latidos del Corazón | np.array([0, -0.25, -0.5, -0.25, 0, 0.25, 0.5, 0.25, 0, 0, 0, 0, 0, 1, 2, 1, 0, -1, -2, -1, 0, 0.5, 1.0, 0.5, 0, 0.25, 0.5, 0.25, 0, -0.25, -0.5, -0.25]) | 0 | 32 |
| Pulsos Rizados | np.array([0, 0.0571, 0.1142, 0.1709, 0.2266, 0.2824, 0.3366, 0.3896, 0.4425, 0.4922, 0.5411, 0.5897, 0.6332, 0.6768, 0.7184, 0.7556, 0.7928, 0.8259, 0.8559, 0.8858, 0.9091, 0.9311, 0.9527, 0.9705, 0.9884, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1]) | -500m | 64 (por resolución) |

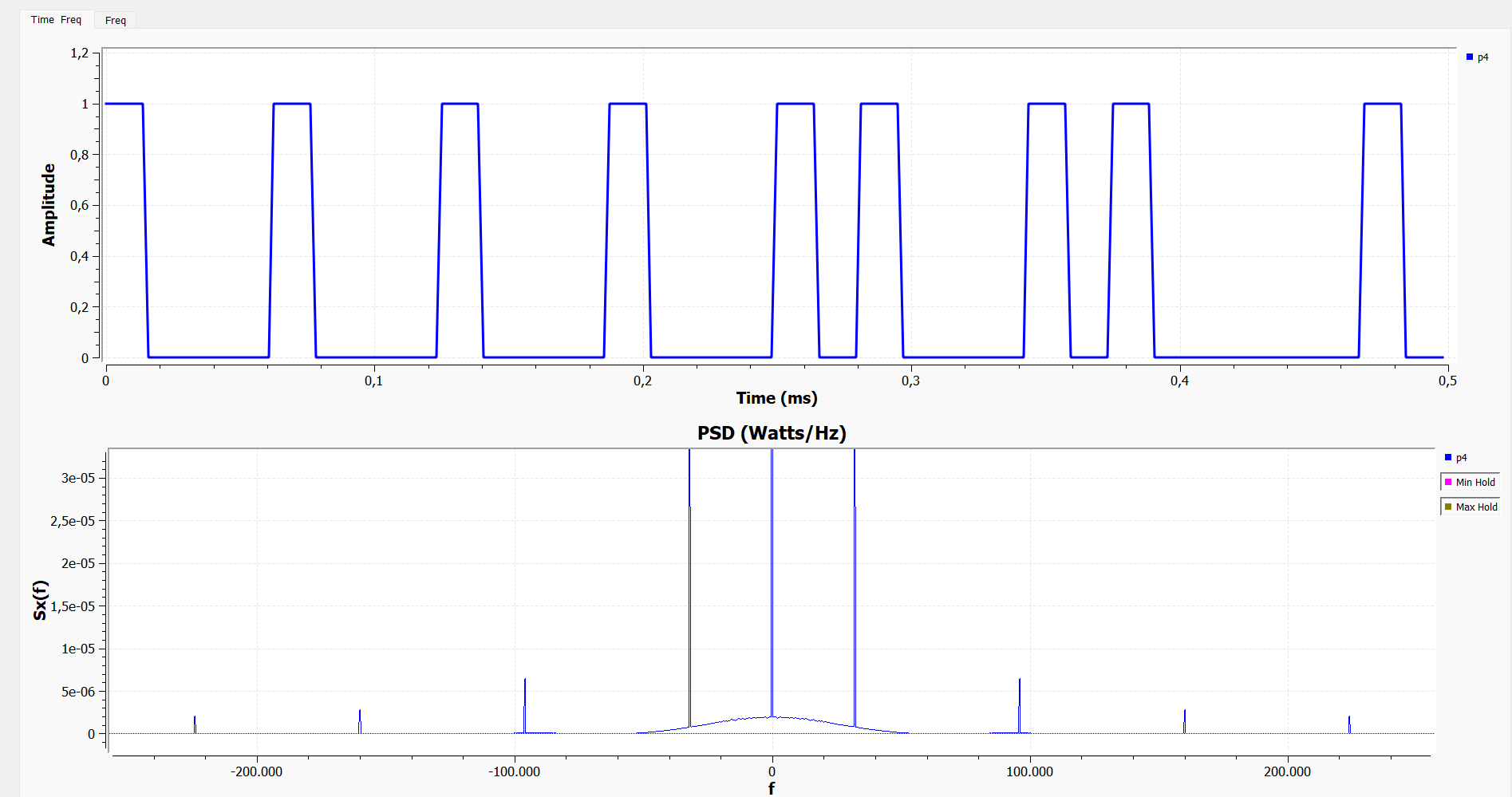
*p. ¿Qué cambios mínimos haría al flujograma, manipulando principalmente h, si desea que los bits en la señal binaria aleatoria tomen la forma de dientes de sierra?*

*np.array ([-1, -0.866, -0.733, -0.6, -0.466, -0.333, -0.2, -0.066, 0.0666, 0.2, 0.3333, 0.4666, 0.6, 0.7333, 0.8666, 1])*

**

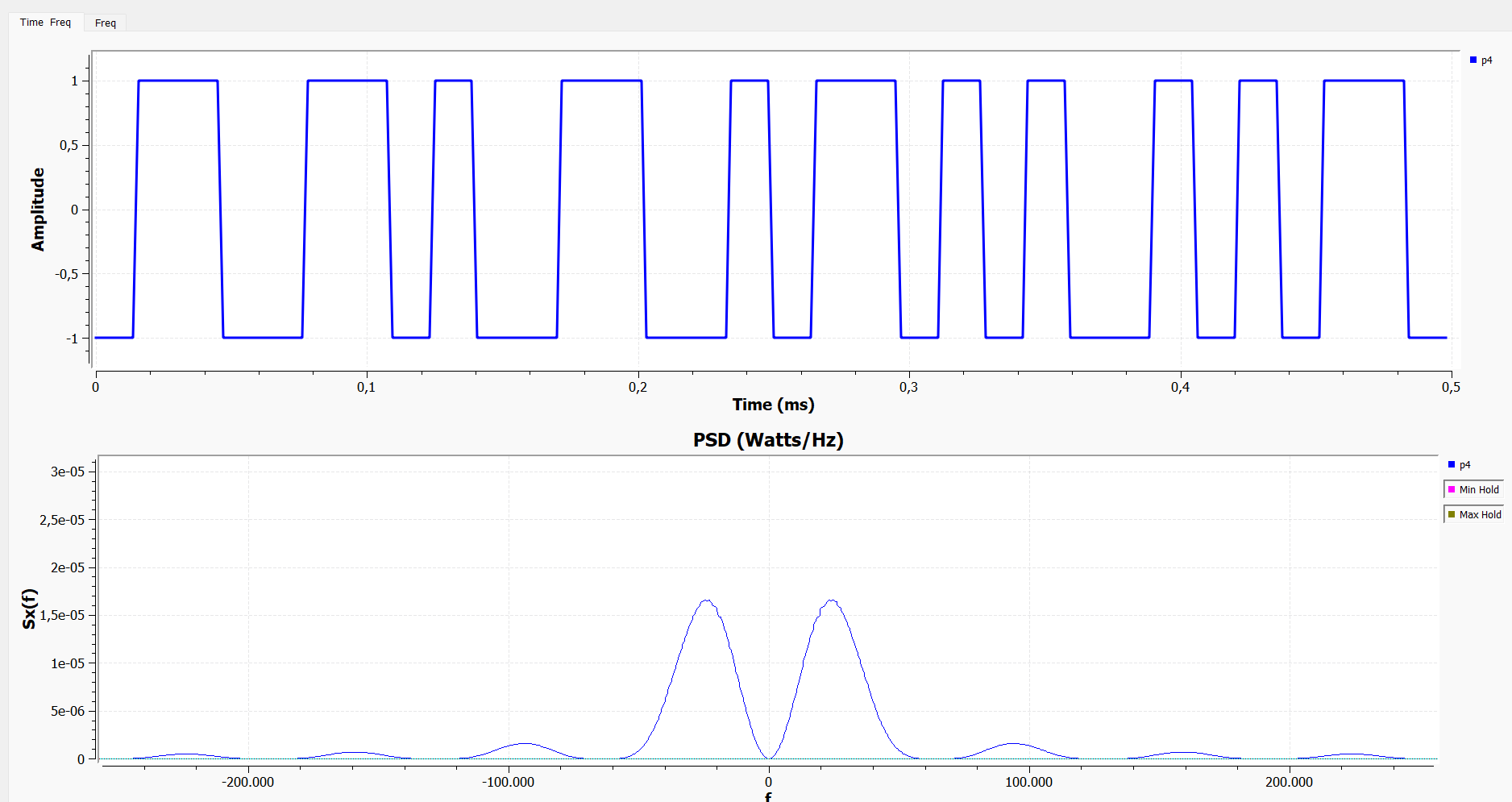
*q. ¿Qué cambios mínimos haría al flujograma, manipulando principalmente h, si desea que la señal binaria aleatoria tenga codificación de línea Unipolar RZ, es decir como se muestra en la Fig. 3?*

*np.array([0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0,0,0,0,0,0,0,0]) y la fuente en 0*

**

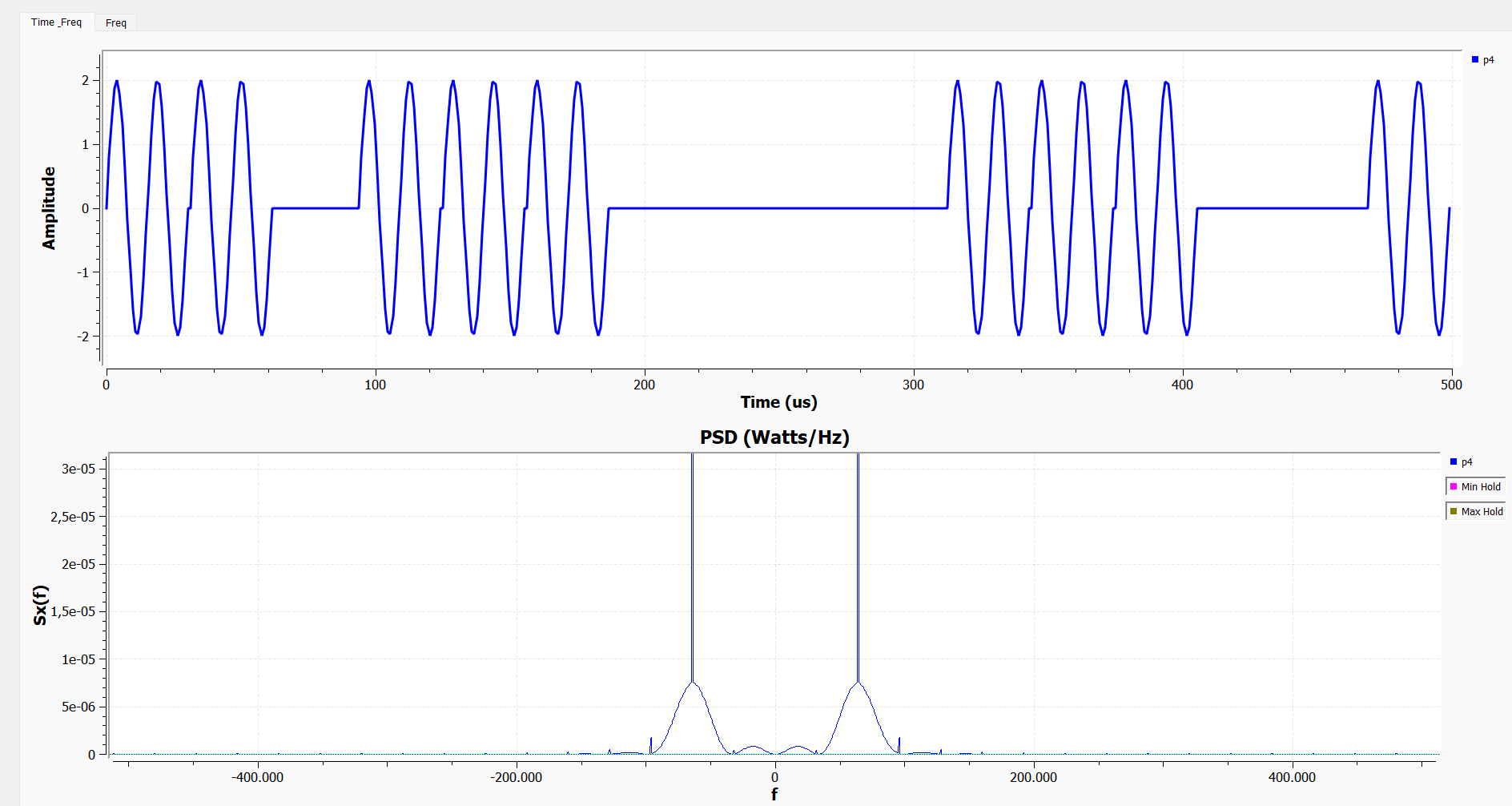
*r. ¿Qué cambios mínimos haría al flujograma, manipulando principalmente h, si desea que la señal binaria aleatoria tenga codificación de línea Manchester NRZ, es decir como se muestra en la Fig. 3?*

np.array([1,1,1,1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1])

**

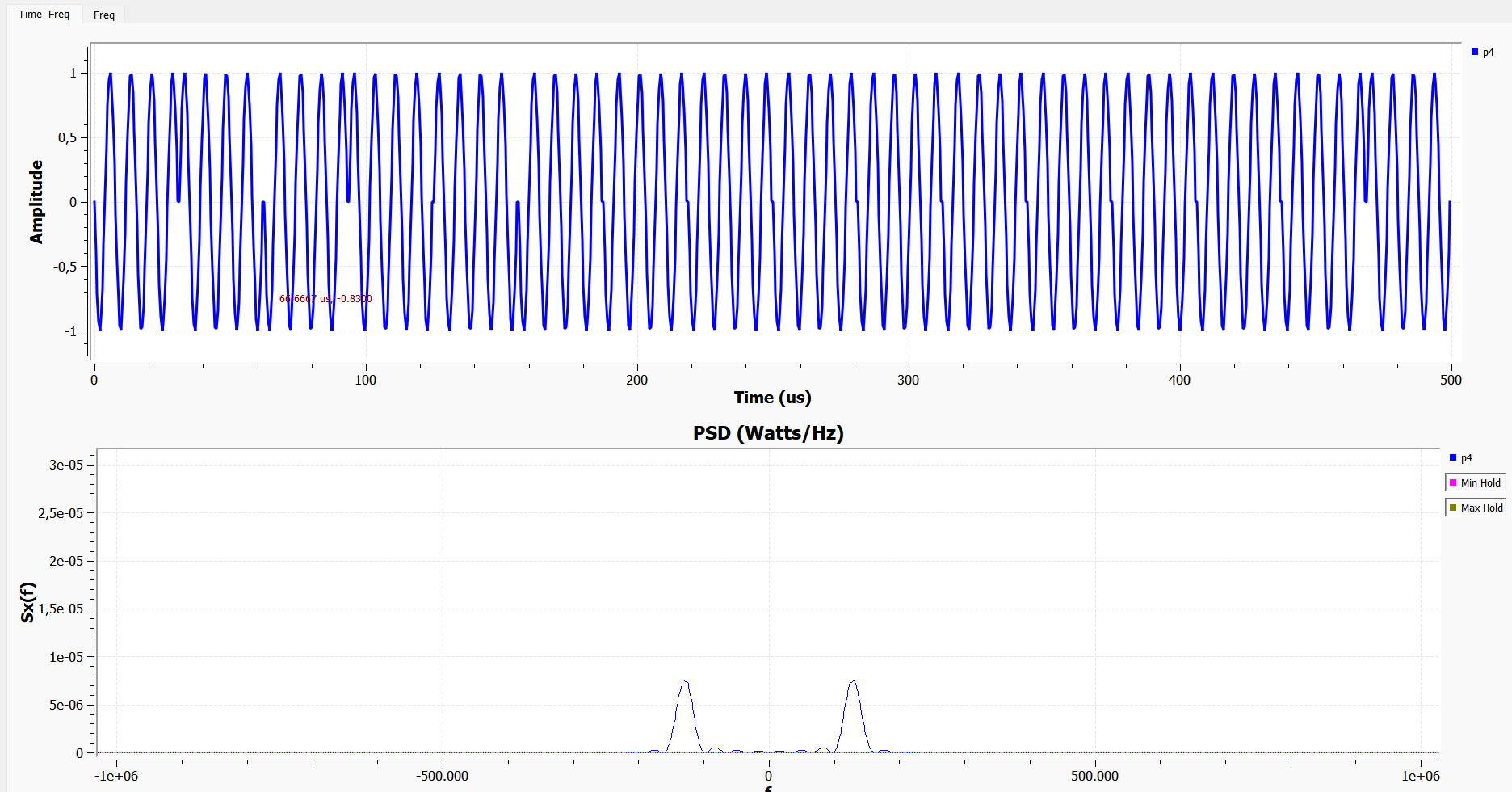
*s. ¿Qué cambios mínimos haría en el flujograma, aprovechando h y el FIR Interpolating Filter para que la señal binaria tenga la forma de señal OOK como se muestra en la Figura 4?*

np.sin(np.linspace(0, 4\*np.pi, 32))



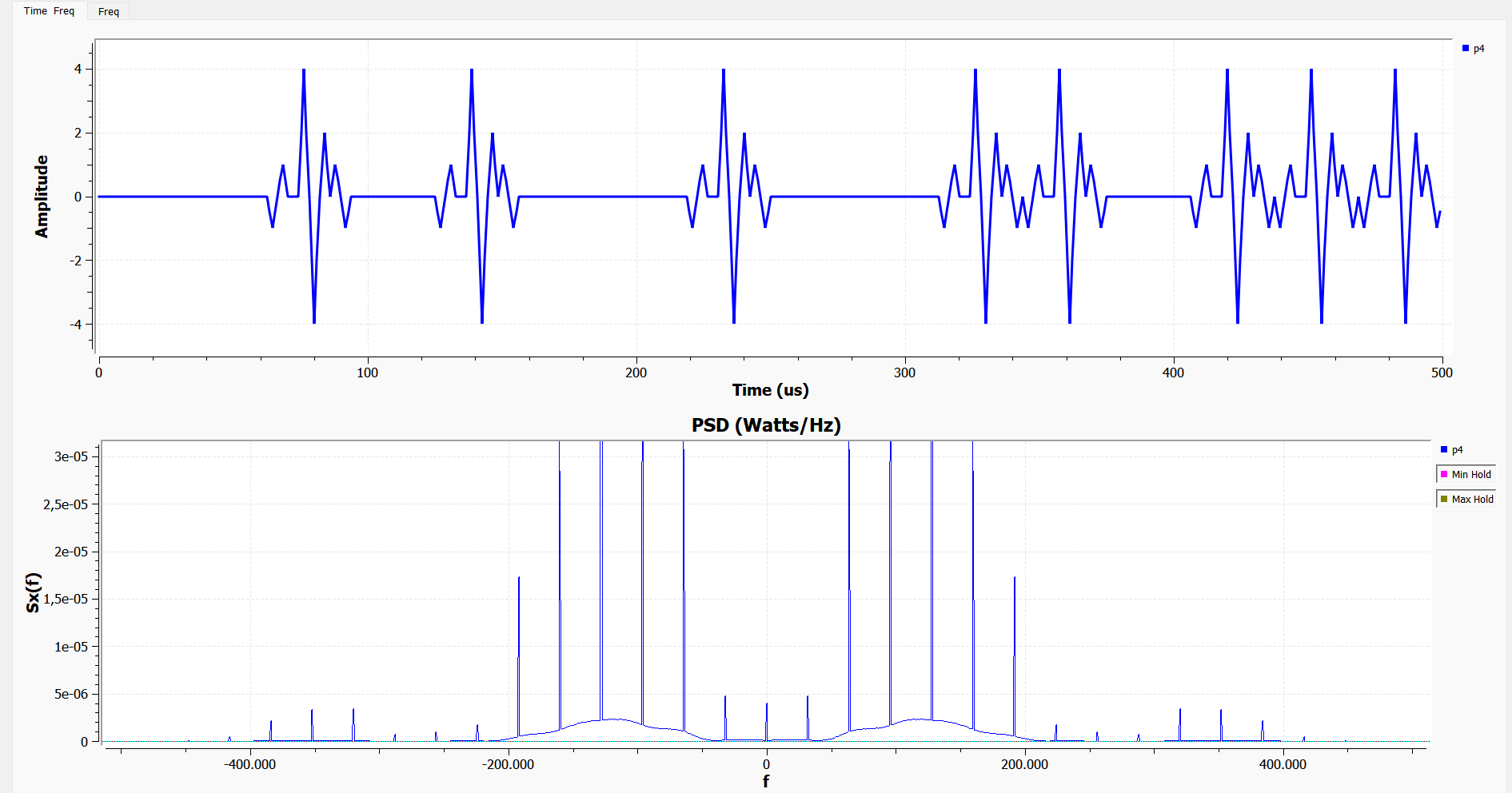
*t. ¿Qué cambios mínimos haría en el flujograma, aprovechando h y el FIR Interpolating Filter para que la señal binaria tenga la forma de señal BPSK como se muestra en la Figura 4?*

np.sin(np.linspace(0, 8\*np.pi, 64))



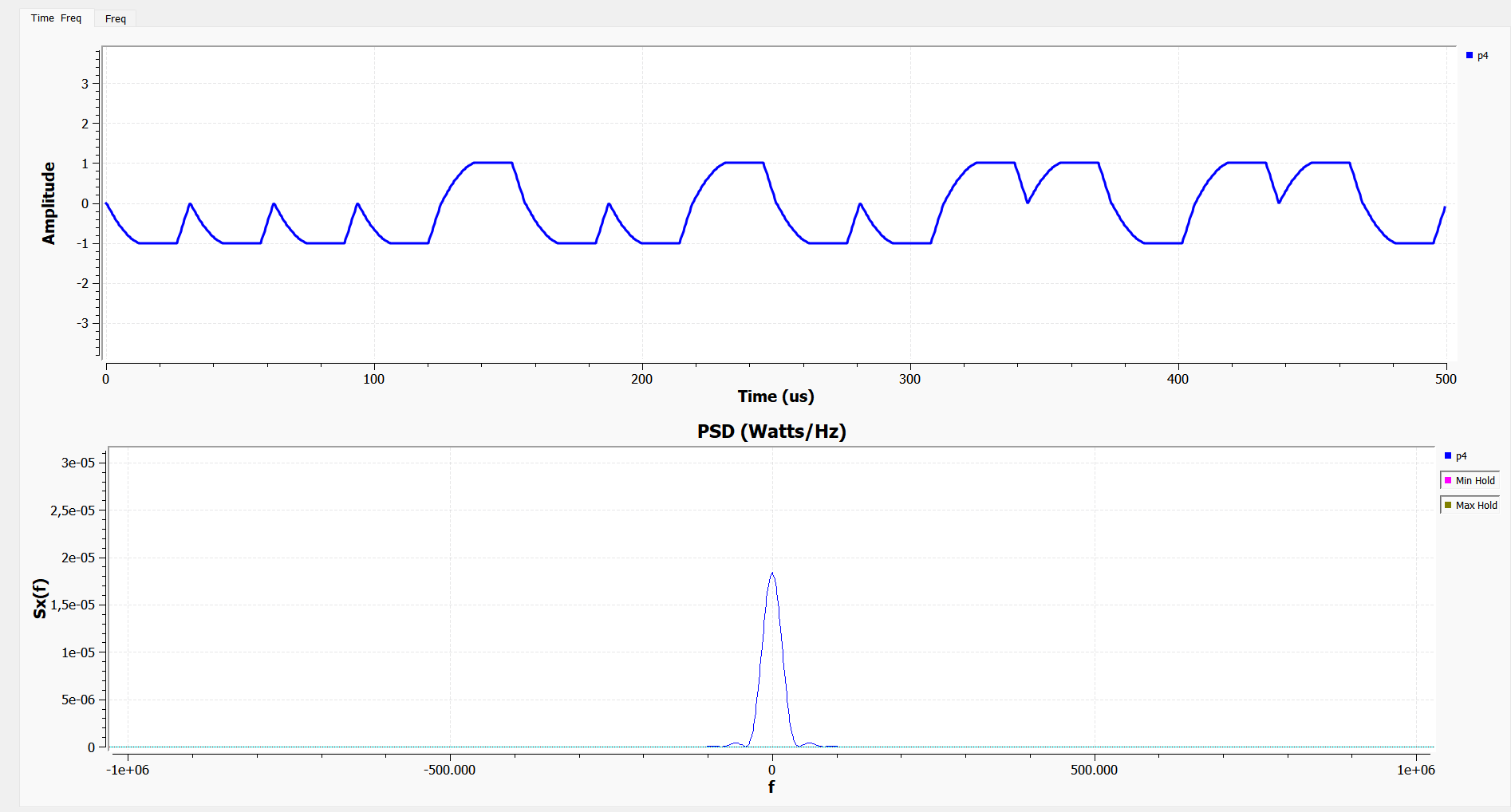
*u. ¿Qué cambios mínimos haría en el flujograma, aprovechando h y el FIR Interpolating Filter para que la señal binaria tenga la forma de los latidos del corazón como se muestra en la Figura 5?*

np.array([0, -0.25, -0.5, -0.25, 0, 0.25, 0.5, 0.25, 0, 0, 0, 0, 0, 1, 2, 1, 0, -1, -2, -1, 0, 0.5, 1.0, 0.5, 0, 0.25, 0.5, 0.25, 0, -0.25, -0.5, -0.25])

**

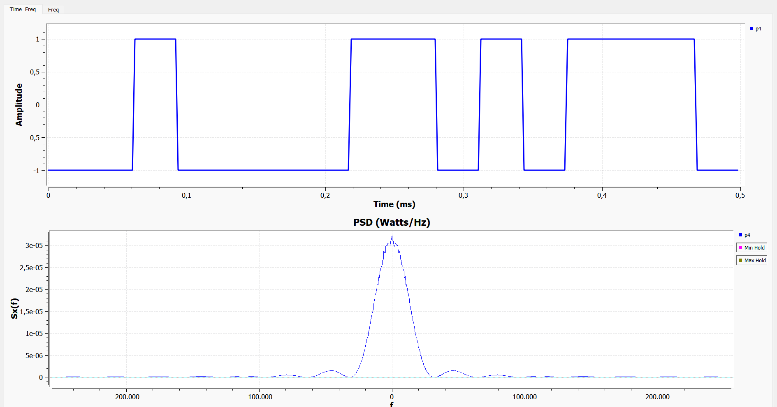
*v. ¿Qué cambios mínimos haría en el flujograma, aprovechando h y el FIR Interpolating Filter para que la señal binaria tenga la forma que se muestra en la Figura 6?*

np.array([0.0, 0.0571, 0.1142, 0.1709, 0.2266, 0.2824, 0.3366, 0.3896, 0.4425, 0.4922, 0.5411, 0.5897, 0.6332, 0.6768, 0.7184, 0.7556, 0.7928, 0.8259, 0.8559, 0.8858, 0.9091, 0.9311, 0.9527, 0.9705, 0.9884, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1])

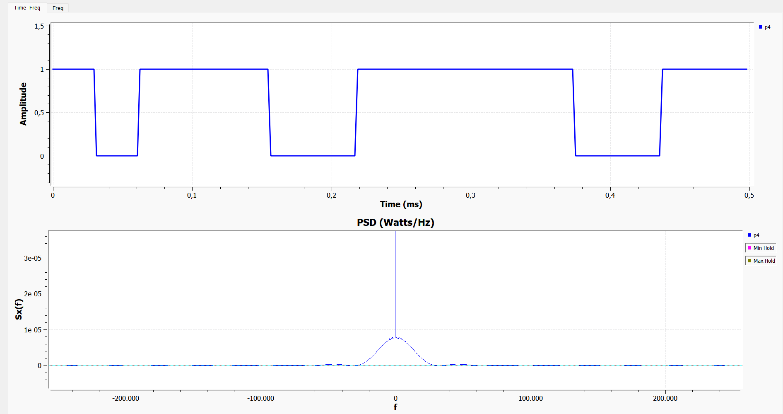
**

*w. Explique usando gráficas de PSD la diferencia que existe entre la PSD de una señal binaria bipolar y una unipolar.*

e. ¿Qué pasaría con la PSD si no se hace la conversión a señal bipolar, sino que la señal binaria en p4 solo tiene valores de 0 ó 1 en lugar de -1 ó 1?

**

Bipolar

**

Unipolar

Preguntas a Responder  
a. ¿Qué papel juega la siguiente combinación de bloques?

b. ¿Qué papel juega el bloque “Interpolationg FIR Filter”, ¿cómo funciona?

1. ¿Por qué el parámetro “Interpolation” en el bloque vale “SPS” y qué pasa si se coloca otro valor?

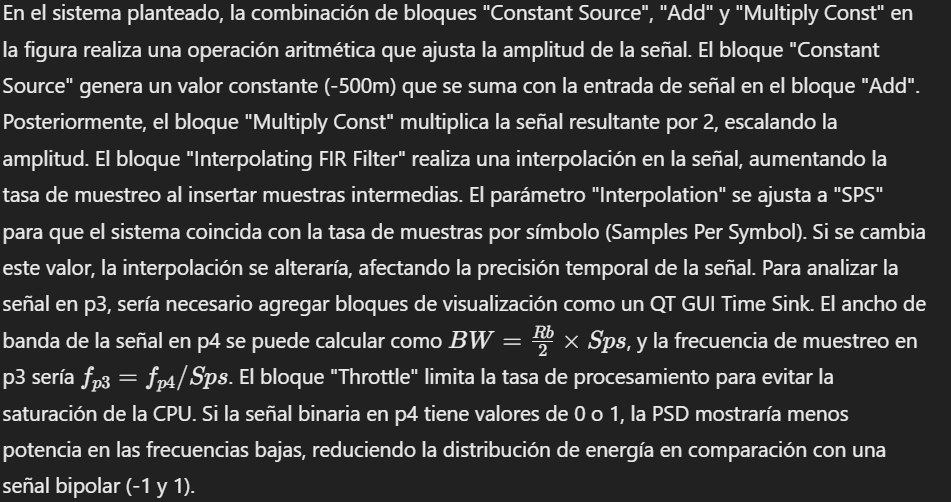
2. Si tuviese que analizar la señal en p3, ¿qué cambios realizaría en la instrumentación (esquema de GNU Radio)?

3. ¿Qué fórmula permite conocer el ancho de banda de la señal en p4 si se conoce Rb y Sps?

4. ¿Qué fórmula permite conocer la frecuencia de muestreo en p3, si se conoce la frecuencia de muestreo en p4 y Sps?

d. ¿Qué papel juega el bloque “Throttle”?

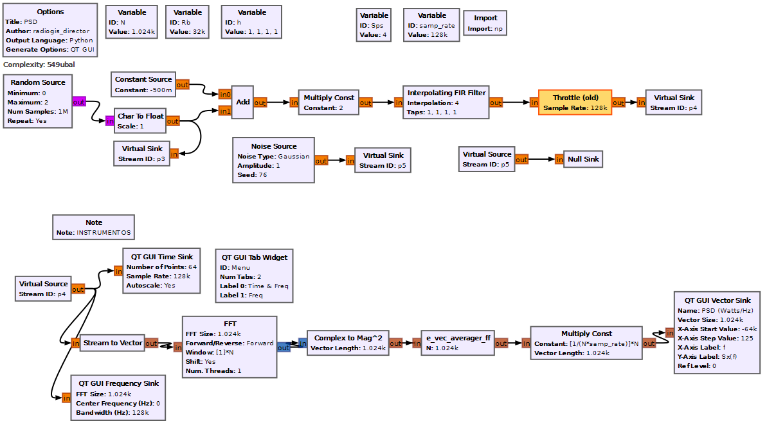
Para analizar la señal en p3, sería necesario agregar bloques de visualización como un QT GUI Time Sink. El ancho de banda de la señal en p4 se puede calcular como BW=Rb2×SpsBW = \frac{Rb}{2} \times SpsBW=2Rb​×Sps, y la frecuencia de muestreo en p3 sería fp3=fp4/Spsf\_{p3} = f\_{p4} / Spsfp3​=fp4​/Sps. El bloque "Throttle" limita la tasa de procesamiento para evitar la saturación de la CPU. Si la señal binaria en p4 tiene valores de 0 o 1, la PSD mostraría menos potencia en las frecuencias bajas, reduciendo la distribución de energía en comparación con una señal bipolar (-1 y 1).



El esquema comienza con la implementación del bloque “**Random Source” el cual es el** encargado de generar una señal aleatoria unipolar, la cual una vez generada pasa por combinación de los bloques "**Constant Source**", "**Add**" y "**Multiply Const**" los cuales se encargan de ajustar la amplitud de la señal sumando un valor constante (-500m) y luego multiplicando el resultado por 2 para amplificar la señal, resultado así en una señal binaria aleatoria bipolar. Después se dirige al bloque "**Interpolating FIR Filter**" el cual realiza una interpolación en la señal, aumentando la tasa de muestreo al insertar muestras intermedias. El parámetro "*Interpolation*" se ajusta a "*Sps*"(Samples Per Symbol). Si se cambia este valor, la interpolación se alteraría, afectando la precisión temporal de la señal, ayudando a que sea más o menos precisa según el valor de “Sps” seleccionado. Finalmente, el "**Throttle**" limita la tasa de procesamiento para evitar la saturación de la CPU.

Pasando a la parte de instrumentación del esquemático, ya que los bloques correspondientes a “**Virtual Sink**”, “**Null sink**” y “**Virtual Source**” desempeñan las funciones de almacenar, eliminar y generar señales respectivamente. Inicialmente, estos bloques se utilizan para guardar la señal unipolar generada, una señal de ruido (“**Noise Source”**) y la señal bipolar aleatoria. El bloque "Null Sink" elimina la señal de ruido, ya que no es necesaria en el procesamiento inicial. En la instrumentación se inicial con la generación de una de las señales guardadas, en este caso p4, con el bloque “**Virtual Source**”, sobre el cual se puede cambiar dicho parámetro para generar la señal necesaria para el análisis (Puede ser p3 o p5, sin embargo, es necesario borrar las que no sean utilizadas en dicho análisis), a continuación se utiliza “**Stream to Vector”** para convertir la señal en vectores que luego se procesan mediante una “**FFT**” transformando la señal al dominio de la frecuencia. Finalmente, la magnitud al cuadrado de la FFT se calcula con “**Complex to Mag^2”**, mostrando la potencia de la señal en los bloques “**QT GUI Time Sink”** y “**QT GUI Frequency Sink”**, donde se visualizan las señales en tiempo real en el dominio del tiempo y frecuencia.

El esquema comienza con el bloque "Random Source", que genera una señal aleatoria unipolar. Luego, esta señal pasa por los bloques "Constant Source", "Add" y "Multiply Const", que ajustan su amplitud sumando un valor constante (-500m) y multiplicándola por 2, resultando en una señal binaria bipolar. A continuación, el bloque "Interpolating FIR Filter" realiza una interpolación, aumentando la tasa de muestreo insertando muestras intermedias, ajustada por el parámetro "Sps" (Samples Per Symbol). Cambiar este valor afecta la precisión temporal de la señal. El bloque "Throttle" regula la tasa de procesamiento para evitar la saturación de la CPU. En la instrumentación, los bloques "Virtual Sink", "Null Sink" y "Virtual Source" almacenan, eliminan o generan señales. Se guarda la señal unipolar, ruido y la señal bipolar, eliminando la señal de ruido innecesaria. Posteriormente, se genera la señal desde "Virtual Source" (p4) (Puede ser p3 o p5, sin embargo, es necesario borrar las que no sean utilizadas en dicho análisis), que se convierte en vectores con "Stream to Vector", para ser procesada con una "FFT" al dominio de la frecuencia. Finalmente, la magnitud al cuadrado de la FFT se calcula con "Complex to Mag^2" y se visualiza en "QT GUI Time Sink" y "QT GUI Frequency Sink" en tiempo real.



pasa por “**Constant Source”** un bloque **Add**, que suma este valor a la señal. Luego, la señal resultante se amplifica con el bloque **Multiply Const** (Constante = 2), ajustando su amplitud. Posteriormente, el **Interpolation FIR Filter** aumenta la tasa de muestreo de la señal según el parámetro **SPS**, generando varias muestras de salida por cada entrada, lo que mejora la resolución de la señal. Después, el bloque **Throttle** controla la tasa de procesamiento para evitar que el flujo de datos exceda las capacidades del hardware, limitando a 512k muestras por segundo. Adicionalmente, se introduce ruido gaussiano mediante un **Noise Source**, el cual se suma a la señal antes de ser dirigida a un **Virtual Sink** o **Null Sink** que, respectivamente, almacenan o descartan la señal. En la parte inferior del esquema, se utiliza **Stream to Vector** para convertir la señal en vectores que luego se procesan mediante una **FFT** (Transformada Rápida de Fourier), transformando la señal al dominio de la frecuencia. Finalmente, la magnitud al cuadrado de la FFT se calcula con **Complex to Mag^2**, mostrando la potencia de la señal en los bloques **QT GUI Time Sink** y **QT GUI Frequency Sink**, donde se visualizan las señales en tiempo real en el dominio del tiempo y frecuencia.

h. ¿Qué fórmula podría ayudar a calcular el número de lóbulos de la PSD de señal binaria aleatoria de forma rectangular cuando se conoce la frecuencia de muestreo y Sps? Nota: el lóbulo de la mitad se cuenta como dos porque tiene el doble de ancho que los demás.

i. ¿Cómo se calcula todo el rango de frecuencias que ocupa el espectro cuando se conoce Rb y Sps?

j. ¿Cómo se calcula la resolución espectral del analizador de espectros, cuando se conoce N y la frecuencia de muestreo?

l. ¿Cómo calcularía la frecuencia de muestreo a la entrada del bloque “Unpack K Bits” si conoce el número de lóbulos de la PSD y el ancho de banda de la señal?

m. ¿Cómo calcularía la frecuencia de muestreo a la salida del bloque “Unpack K Bits” si conoce la frecuencia de muestreo a la entrada?

n. ¿Cómo calcularía la frecuencia de muestreo a la salida del bloque “Char to Float” si conoce la frecuencia de muestreo a la entrada?

Al modificar los valores de una muestra en una secuencia interpolada, se altera la forma y el comportamiento de la señal. Si los valores de la muestra generan variaciones bruscas o transiciones repentinas, como picos o saltos, la señal presentará componentes de alta frecuencia que afectan su suavidad, provocando distorsiones no deseadas. Por otro lado, al ajustar los valores de la muestra de manera controlada, las transiciones serán más suaves, lo que permitirá mantener la integridad de la señal sin añadir frecuencias altas ni sobresaltos. En resumen, al variar los valores de una muestra, se puede controlar la forma de la señal y su comportamiento, garantizando una interpolación más precisa y estable si se realiza de forma adecuada.

|  |  |  |  |
| --- | --- | --- | --- |
| Señal | Valor de [h] | Constant Source | Interpolating FIR |
| Dientes de Cierra | *np.array ([-1, -0.866, -0.733, -0.6, -0.466, -0.333, -0.2, -0.066, 0.0666, 0.2, 0.3333, 0.4666, 0.6, 0.7333, 0.8666, 1])* | -500m | 16 |
| Unipolar RZ | np.array ([0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0, 0, 0, 0, 0, 0, 0, 0]) | 0 | 16 |
| Manchester NRZ | np.array ([1, 1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1]) | -500m | 16 |
| OOK | np.sin(np.linspace(0, 4\*np.pi, 32)) | 0 | 32 |
| BPSK | np.sin(np.linspace(0, 8\*np.pi, 64)) | -500m | 64 |
| Latidos del Corazón | np.array([0, -0.25, -0.5, -0.25, 0, 0.25, 0.5, 0.25, 0, 0, 0, 0, 0, 1, 2, 1, 0, -1, -2, -1, 0, 0.5, 1.0, 0.5, 0, 0.25, 0.5, 0.25, 0, -0.25, -0.5, -0.25]) | 0 | 32 |
| Pulsos Rizados | np.array([0, 0.0571, 0.1142, 0.1709, 0.2266, 0.2824, 0.3366, 0.3896, 0.4425, 0.4922, 0.5411, 0.5897, 0.6332, 0.6768, 0.7184, 0.7556, 0.7928, 0.8259, 0.8559, 0.8858, 0.9091, 0.9311, 0.9527, 0.9705, 0.9884, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1]) | -500m | 64 |

* La práctica permitió profundizar en la Densidad Espectral de Potencia (PSD) como una herramienta fundamental para el análisis de señales. Se observó cómo la PSD proporciona información crítica sobre la distribución de energía de una señal a través de diferentes frecuencias, lo que es esencial para optimizar sistemas de comunicación.
* A medida que se variaron los parámetros de muestras por símbolo (Sps), se notó un impacto significativo en la representación espectral de la señal. Específicamente, se comprobó que un mayor número de muestras por símbolo mejora la resolución en el dominio de la frecuencia, permitiendo un análisis más preciso de la PSD y revelando patrones que podrían no ser evidentes con configuraciones de Sps más bajas.

Al ajustar la señal de ruido blanco en la fuente

virtual y analizar su PSD, se confirmó que el rui-

do blanco tiene características aleatorias sin una

forma discernible y con una densidad espectral

constante en todas las frecuencias. Este compor-

tamiento cumple con las propiedades teóricas del

ruido blanco.

se observó que el proceso de interpolación permite

una representación más clara de las características

dominantes a lo cual le damos el crédito a su

proceso de suavizado

El uso de interpolación y la modificación del nú-

mero de bits en el procesamiento de la señal realzó

los picos de frecuencia correspondientes a las notas

clave del audio, proporcionando una representa-

ción más detallada del contenido frecuencial.

Se determino que aumentar el valor de Sps se mejora la precisión en la representación de la señal binaria aleatoria bipolar tanto en el dominio del tiempo como en el de la frecuencia. Esto se debe a que permite una mayor tasa de muestreo y por ende una mejor resolución. Con un Sps bajo (Sps=1), la PSD de la señal se asemeja a la de un ruido blanco, mientras que al aumentar Sps se obtiene una mayor definición de las componentes de frecuencia, lo cual optimiza el análisis sin alterar la tasa de bits o el tiempo de bit.