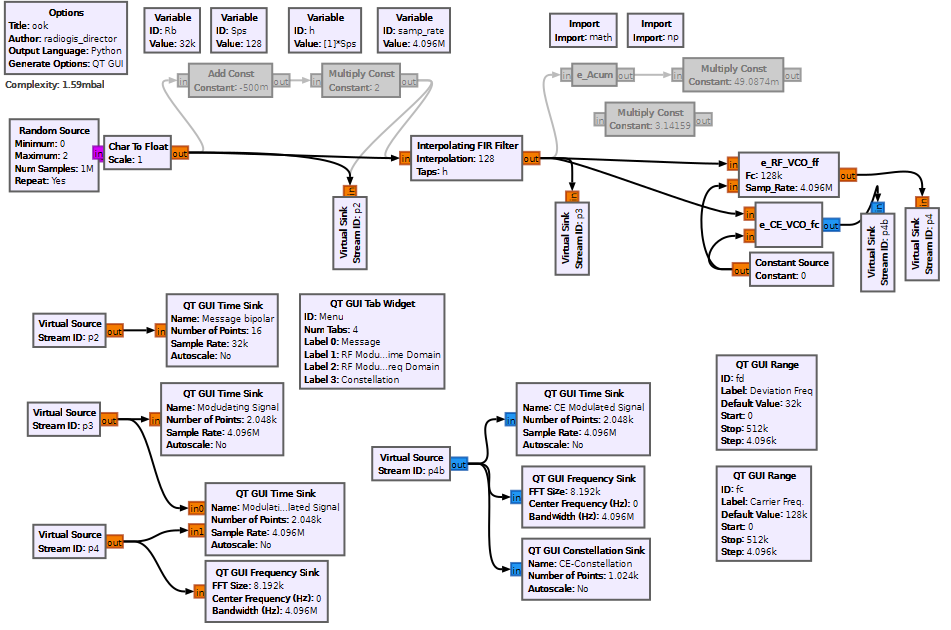
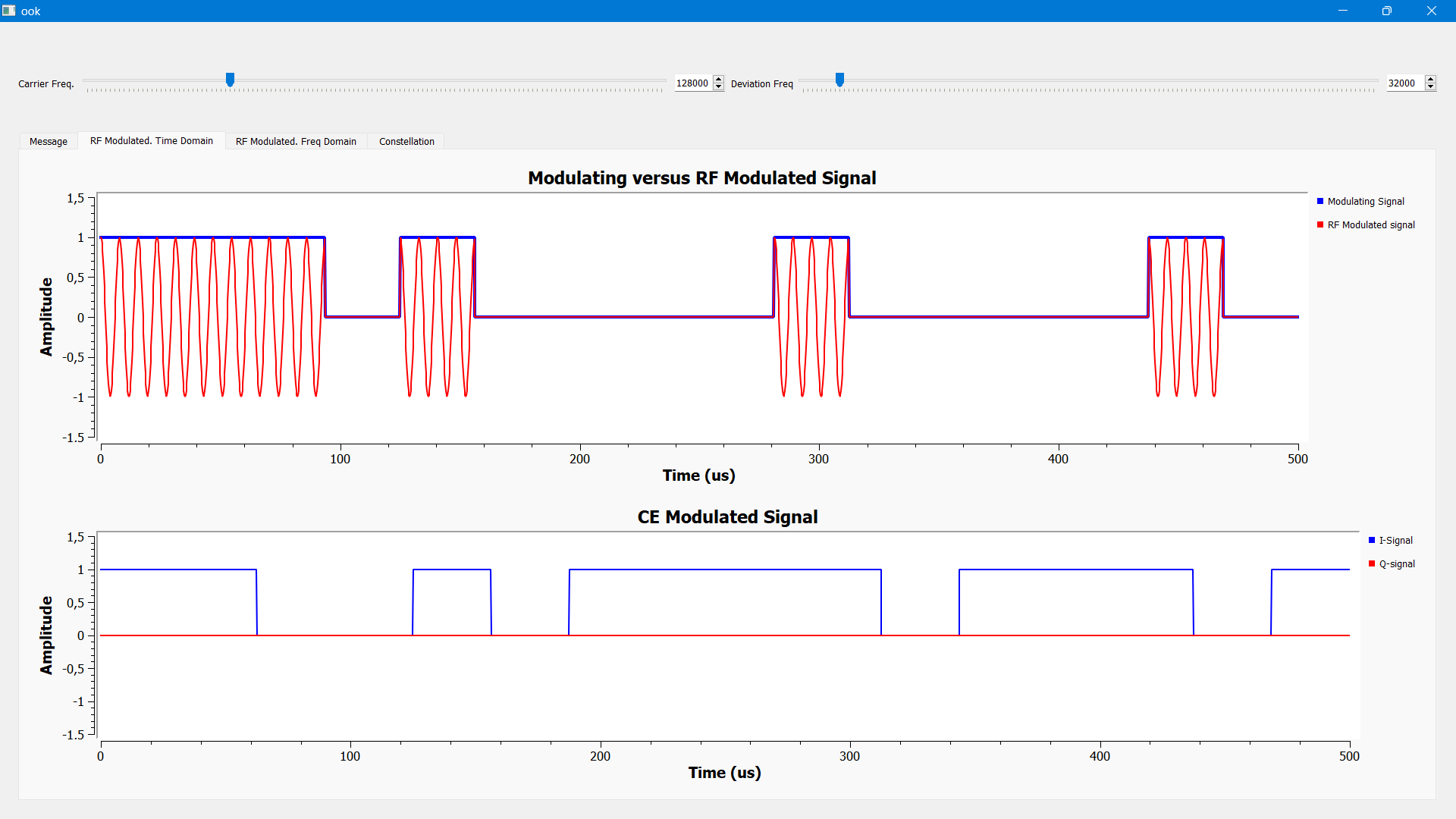
**Analisis Practica 3 de Comunicaciones:**

* **Montaje Base:**

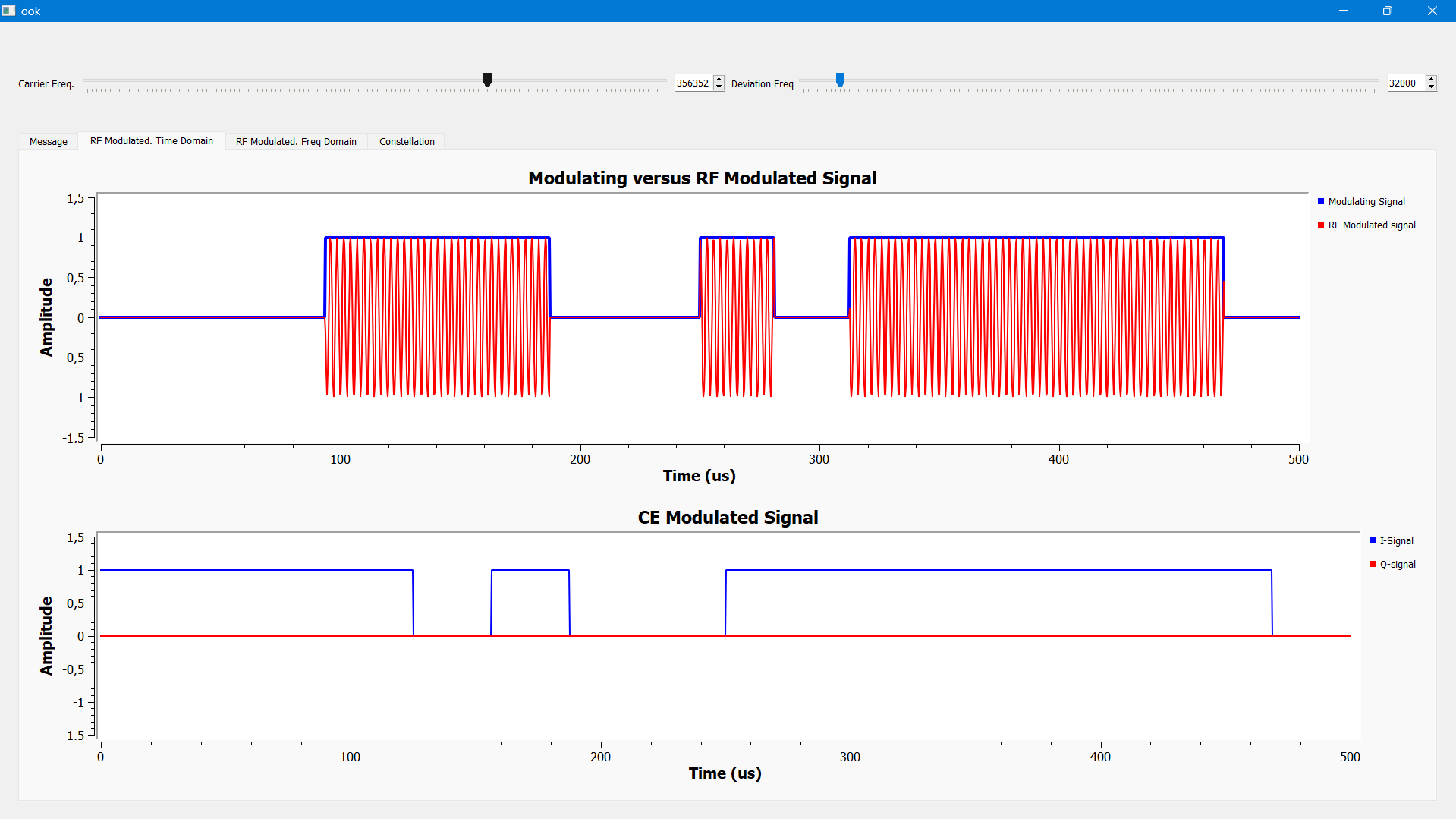


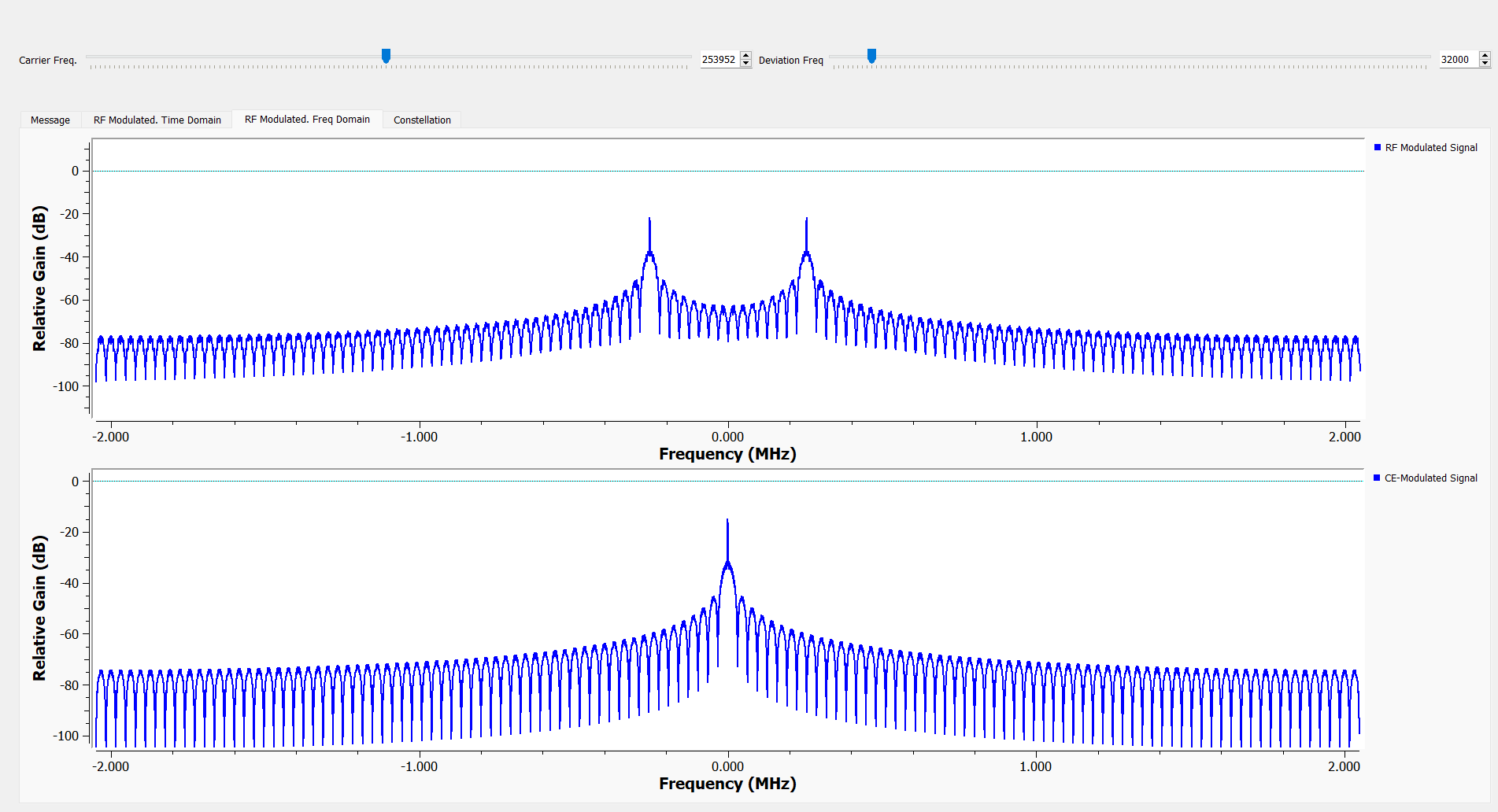
* a. En la pestaña “RF Modulated". Time Domain” observe la señal modulada en versión RF comparada con la señal modulada en versión EC (señal I y Señal Q).

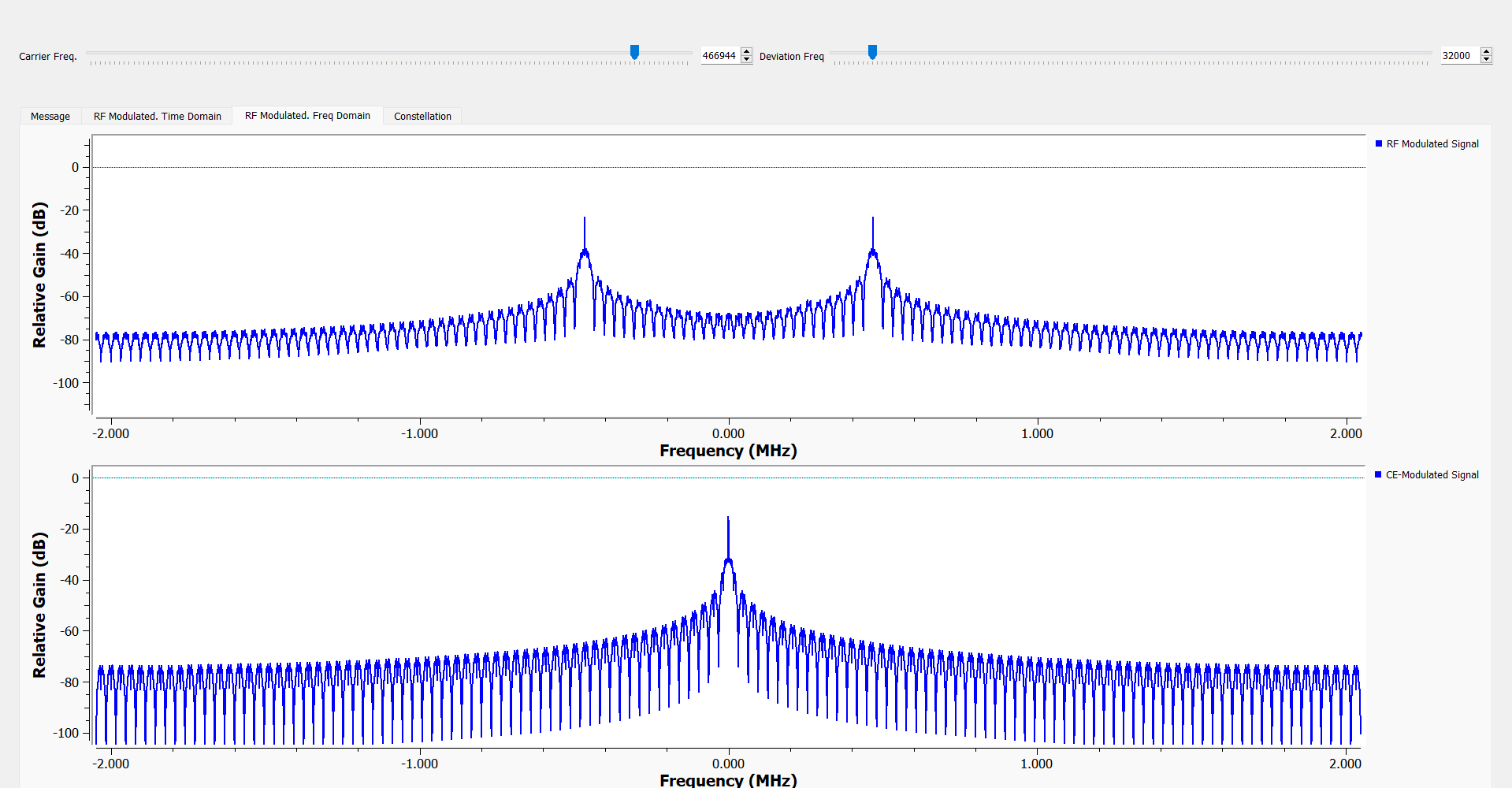


Solo se ve afectaciones en la señal modulada en RF (Radio Frecuencia):

1. Las señales RF en la parte superior son más complejas, mostrando la naturaleza de las ondas de radio.
2. Las señales en la versión EC son más simples y discretas, reflejando cómo la información se transmite a través de cambios en la corriente.







Para contrastar las señales OOK (On-Off Keying) en versiones EC (Corriente Eléctrica) y RF (Radiofrecuencia), se pueden considerar varios aspectos clave:

1. **Forma de la Señal**:
   * **OOK en RF**: La señal modulada muestra una portadora de alta frecuencia que se enciende y apaga según los datos. La representación en el tiempo incluye ondas sinusoidales rápidas, lo que permite transmitir información a través del aire.
   * **OOK en EC**: La señal se presenta con cambios abruptos de voltaje (presencia o ausencia de corriente). La forma de la señal es más rectangular y discreta, lo que facilita la identificación de los niveles de "1" y "0".
2. **Ancho de Banda**:
   * **RF**: Generalmente, requiere un mayor ancho de banda debido a la complejidad de la portadora y su frecuencia más alta. Esto puede llevar a interferencias en entornos congestionados.
   * **EC**: Por su naturaleza más simple, tiende a requerir un ancho de banda menor, lo que puede ser ventajoso en aplicaciones de corto alcance.
3. **Distancia de Transmisión**:
   * **RF**: Puede cubrir distancias más largas y es adecuado para comunicaciones inalámbricas. Sin embargo, sufre más atenuación y ruido.
   * **EC**: Se utiliza para distancias cortas debido a la limitación del medio conductor y la atenuación en la señal eléctrica.
4. **Interferencia**:
   * **RF**: Más susceptible a interferencias de otras señales y factores ambientales (como el clima).
   * **EC**: Menos susceptible a interferencias externas, pero puede ser afectada por ruidos eléctricos en el medio conductor.
5. **Aplicaciones**:
   * **RF**: Ideal para comunicaciones móviles, radiodifusión y sistemas de control remoto.
   * **EC**: Común en aplicaciones industriales, control de sistemas y transmisión de datos en entornos cerrados.

Estas diferencias resaltan cómo la elección entre OOK en RF y EC depende del contexto y requisitos específicos de la aplicación.

Las diferencias en el ancho de banda entre OOK en versiones RF y EC pueden resumirse de la siguiente manera:

1. **OOK en RF**:
   * **Ancho de Banda Mayor**: La modulación OOK en RF generalmente requiere un mayor ancho de banda debido a la presencia de una portadora de alta frecuencia. Este mayor ancho de banda es necesario para permitir que la señal modulada mantenga la calidad y precisión en la transmisión, especialmente en condiciones de interferencia o ruido.
   * **Uso del Espectro**: El uso ineficiente del espectro electromagnético puede conducir a la congestión en frecuencias específicas, lo que puede afectar el rendimiento de otras comunicaciones.
2. **OOK en EC**:
   * **Ancho de Banda Menor**: La versión EC de OOK suele tener un ancho de banda más reducido, ya que la señal se representa mediante cambios en la amplitud de corriente en un medio conductor. Esto permite un uso más eficiente del ancho de banda en sistemas de comunicación de corto alcance.
   * **Menor Susceptibilidad a Interferencias**: Un ancho de banda menor también puede resultar en una menor susceptibilidad a interferencias externas, lo que es ventajoso en entornos controlados.

Estas diferencias reflejan cómo la elección del medio de transmisión (RF o EC) y el tipo de modulación afectan la capacidad de ancho de banda y la calidad de la comunicación.

* **Punto 6 Análisis de los Bloques.**



*import numpy as np*

*from gnuradio import gr*

*import math*

*class blk(gr.sync\_block):*

*"""This block is a RF VCO and works as following: ….."""*

*def \_\_init\_\_(self, fc=128000, samp\_rate=320000):*

*gr.sync\_block.\_\_init\_\_(*

*self,*

*name='e\_RF\_VCO\_ff',*

*in\_sig=[np.float32, np.float32],*

*out\_sig=[np.float32]*

*)*

*self.fc = fc*

*self.samp\_rate = samp\_rate*

*self.n\_m=0*

*def work(self, input\_items, output\_items):*

*A=input\_items[0]*

*Q=input\_items[1]*

*y=output\_items[0]*

*N=len(A)*

*n=np.linspace(self.n\_m,self.n\_m+N-1,N)*

*self.n\_m += N*

*y[:]=A\*np.cos(2\*math.pi\*self.fc\*n/self.samp\_rate+Q)*

*return len(output\_items[0])*

*Analisis:  
Las entradas = Amplitud y Fase  
Salida = Señal Modulada Radio Frecuencia*

*Este código define un bloque de modulador de frecuencia controlada por voltaje (VCO) en GNU Radio. Vamos a desglosarlo:*

1. ***Importaciones****:*
   * *numpy as np: Se utiliza para cálculos numéricos y manejo de arrays.*
   * *from gnuradio import gr: Importa el módulo de GNU Radio que permite crear bloques de procesamiento de señales.*
   * *import math: Proporciona funciones matemáticas, como el cálculo de cosenos.*
2. ***Definición de la clase blk****:*
   * *Hereda de gr.sync\_block, lo que significa que este bloque se basa en la sincronización de datos.*
3. ***Método \_\_init\_\_****:*
   * *fc=128000: Frecuencia de portadora por defecto.*
   * *samp\_rate=320000: Tasa de muestreo por defecto.*
   * *gr.sync\_block.\_\_init\_\_: Inicializa el bloque con un nombre y especifica las señales de entrada y salida:*
     + *in\_sig=[np.float32, np.float32]: Dos señales de entrada, ambas de tipo float32.*
     + *out\_sig=[np.float32]: Una señal de salida de tipo float32.*
   * *self.n\_m=0: Inicializa un contador para llevar la cuenta de las muestras procesadas.*
4. ***Método work****:*
   * *input\_items: Lista que contiene las señales de entrada.*
   * *output\_items: Lista para almacenar las señales de salida.*
   * *A=input\_items[0]: Primera señal de entrada (amplitud).*
   * *Q=input\_items[1]: Segunda señal de entrada (fase).*
   * *y=output\_items[0]: Asigna la salida.*
   * *N=len(A): Obtiene el número de muestras en A.*
   * *n=np.linspace(self.n\_m, self.n\_m+N-1, N): Genera un array que representa las muestras actuales, desde n\_m hasta n\_m + N - 1.*
   * *self.n\_m += N: Actualiza el contador de muestras.*
   * *y[:] = A \* np.cos(2 \* math.pi \* self.fc \* n / self.samp\_rate + Q): Calcula la señal de salida como el producto de A y el coseno de la frecuencia modulada, añadiendo la fase Q.*
   * *return len(output\_items[0]): Devuelve el número de muestras procesadas.*