Práctica 1: Programación de sistemas en tiempo real

JHONATAN FELIPE VALEST FLORES-2184672
JUAN SEBASTIAN ROJAS ARIZA-2164699
JEIFER IVÁN BERNAL TELLEZ – 2194679

https://github.com/Sebasttian01/CommunicationsII_M5

Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones Universidad Industrial de Santander

8 de Septiembre de 2024

Resumen

El siguiente informe explica el desarrollo sobre la generación de ramas para le trabajo de repositorios en el entorno de github, también se hace una introducción a la programación de bloques en el programa GNU radio mediante Python, se analizan los procesos para el tratamiento de un flujo información en formato stream.

Palabras clave: Programación, transmisión, promedio de una señal, desviación estándar, Github

1. Introducción

En el presente informe se presenta una guía de enseñanza enfocada en introducir a los lectores en el mundo de la programación de bloques utilizando GNU Radio y Python. GNU Radio es una herramienta de software de código abierto ampliamente utilizada en el ámbito de la radio definida por software, que permite a los usuarios crear aplicaciones de procesamiento de señales de manera flexible y eficiente. La programación de bloques, una de las características fundamentales de GNU Radio, facilita la construcción de flujos de procesamiento mediante la interconexión de módulos básicos, simplificando la implementación y análisis de complejos algoritmos de tratamiento de señales.

El objetivo de esta guía es proporcionar un acercamiento inicial al uso de GNU Radio a través de ejemplos prácticos en Python, explorando los procesos necesarios para manipular flujos de información en formato stream. A lo largo del informe, se detallarán los pasos clave y consideraciones que deben tenerse en cuenta al trabajar con datos en tiempo real, facilitando la comprensión de los conceptos fundamentales y la implementación de soluciones en aplicaciones de procesamiento de señales.

2. Procedimiento

INTRODUCCION A GITHUB

- **CREAR LLAVE Y MODO TERMINAL**

Para la creación de la llave es necesario tener en cuenta lo siguiente:

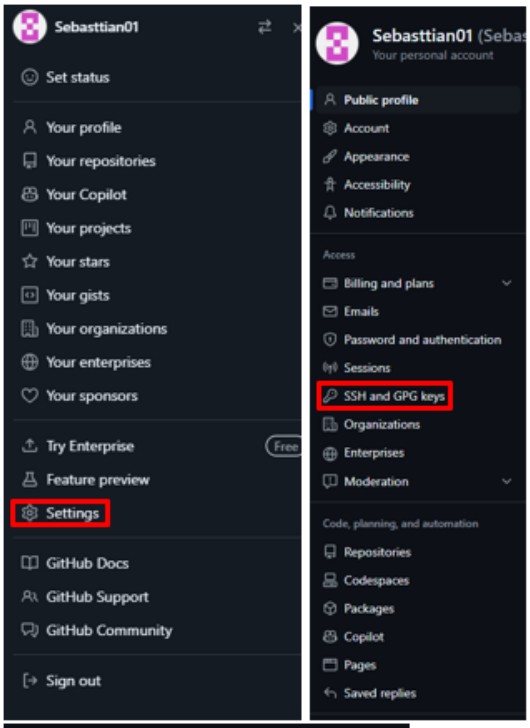
- Tener una cuenta de GitHub creada
- Tener acceso al repositorio al que se desea editar de forma compartida con los compañeros
- Tener instalada la librería de Git en Ubuntu

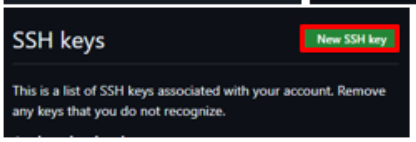
Se procede a generar una nueva clave SSH en el equipo local por medio de la ventana de comandos y se procede a ejecutar el siguiente comando:

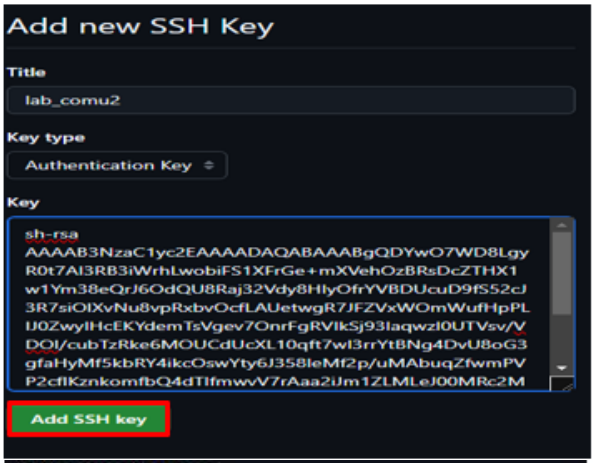
ssh-keygen –t ed25519

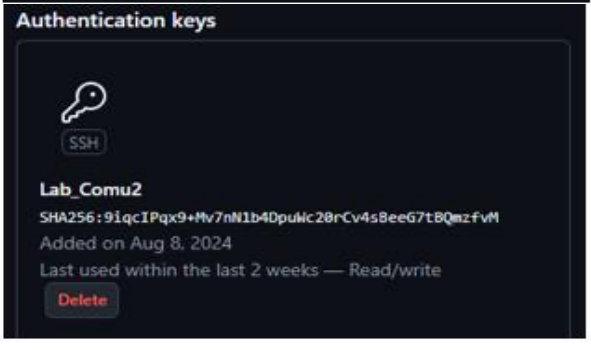
genera una clave SSH en formato randomart, estando en el directorio /home/username/ .ssh se procede a examinar los archivos con el comando ls, el archivo con terminación .pub es el necesario para poder visualizar la clave y llevarla a la plataforma de GitHub para obtener el acceso Luego se procede a escribir el comando cat “nombre_del_archivo”.pub

Luego se procede a copiar y pegar todos los caracteres alfanuméricos en la página de GitHub ingresando a los ajustes del GitHub así:

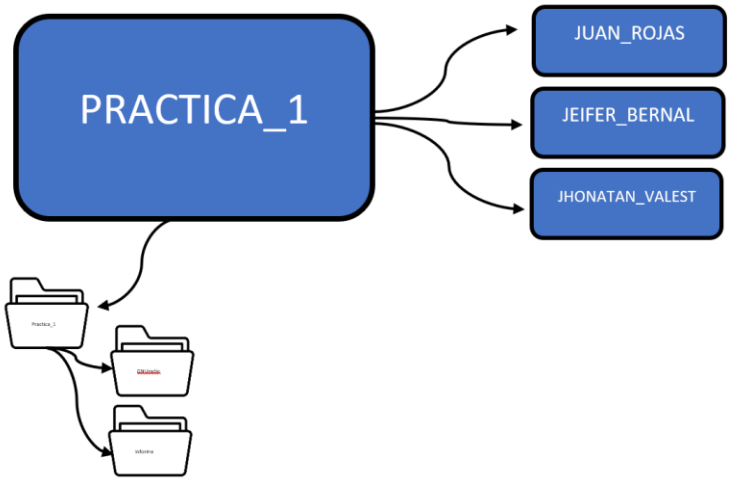








• CREACION DE RAMAS PARA PODER TRABAJAR EN EL ENTORNO DE GITHUB:



Para el desarrollo de esta práctica inicialmente tuvimos en cuenta las ecuaciones para las mediciones de:

Media: $X_m =< x(t) >$

Media Cuadrática: $X_C =< x^2(t) >$

Potencia normalizada: $P = XRMS^2 < |x(t)|^2 >$

Valor RMS: $XRMS = \sqrt{< |x(t)|^2 >}$

Desviación Estándar: $\sigma x = \sqrt{< [x(t) - X_m]^2 >}$

Tras identificar los parámetros a calcular iniciamos el diseño del sistema a analizar, usando la herramienta GNU Radio, donde generamos señales y las analizamos en el dominio del tiempo y la frecuencia.

Luego se procedió a crear un bloque de Python, mediante Python block en GNU-radio para realizar el cálculo de los parámetros ya mencionados anteriormente mediante el código observado en Figure 1 Codigo bloque para hallar promedios de tiempo:

```python
import numpy as np
from gnuradio import gr

class blk(gr.sync_block):

    def __init__(self):  # only default arguments here
        gr.sync_block.__init__(
            self,
            name='Promedios_de_tiempos',   # will show up in GRC
            in_sig=[np.float32],
            out_sig=[np.float32,np.float32,np.float32,np.float32,np.float32]
        )
        self.acum_anterior = 0
        self.Ntotales = 0
        self.acum_anterior1 = 0
        self.acum_anterior2 = 0

    def work(self, input_items, output_items):
        x = input_items[0]    #Senial de entrada.
        y0 = output_items[0] #Promedio de la senial
        y1 = output_items[1] #Media de la senial
        y2 = output_items[2] #RMS de la senial
        y3 = output_items[3] #Potencia promedio de la senial
        y4 = output_items[4] #Desviacion estandar de la senial

        #Calculo del promedio
        N = len(x)
        self.Ntotales = self.Ntotales + N
        acumulado = self.acum_anterior + np.cumsum(x)
        self.acum_anterior = acumulado[N-1]
        y0[:]=acumulado/self.Ntotales

        #Calculo de la media cuadratica
        x2=np.multiply(x,x)
        acumulado1=self.acum_anterior1 + np.cumsum(x2)
        self.acum_anterior1 = acumulado[N-1]
        y1[:] = acumulado1/self.Ntotales

        #Calculo de la RMS
        y2[:] = np.sqrt(y1)

        #Calculo de la potencia promedio
        y3[:] = np.multiply(y2,y2)

        #Calculo de la desviacion estandar
        x3 = np.multiply(x-y0,x-y0)
        acumulado2 = self.acum_anterior2 + np.cumsum(x3)
        self.acum_anterior2 = acumulado2[N-1]
        y4[:] = np.sqrt(acumulado2/self.Ntotales)

        return len(x)
```

Figure 1 Codigo bloque para hallar promedios de tiempo

• Promedio de la señal (y0): Se calcula el promedio de la señal de entrada. Este cálculo se realiza sumando todos los valores de la señal y dividiendo el resultado por el número total de muestras.

• Media cuadrática de la señal (y1): Se calcula la media cuadrática de la señal de entrada. Esto implica elevar al cuadrado cada muestra de la señal, sumar estos valores y luego dividir el resultado por el número total de muestras.

• RMS (Root Mean Square) de la señal (y2): Se calcula la raíz cuadrada de la media cuadrática obtenida anteriormente. Esto proporciona la raíz cuadrada del promedio de los valores cuadráticos de la señal, lo que representa una medida de la magnitud de la señal.

• Potencia promedio de la señal (y3): Se calcula la potencia promedio de la señal, lo cual implica elevar al cuadrado el RMS de la señal. Esto representa la potencia media de la señal.

• Desviación estándar de la señal (y4): Se calcula la desviación estándar de la señal, que es una medida de dispersión que indica cuánto se desvían los valores de la señal respecto al promedio calculado.

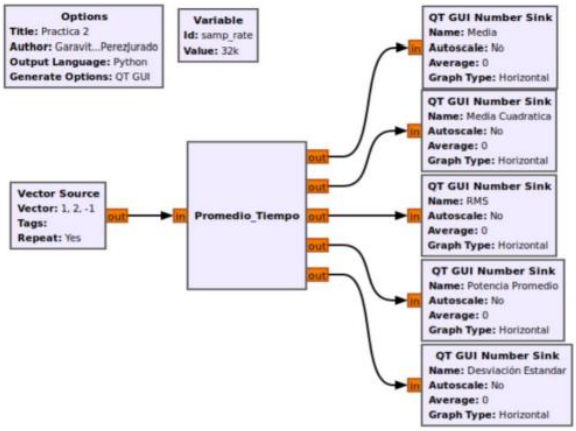


Figure 2 Diagrama para evaluar el bloque de promedios de tiempo

3. Conclusiones

- El bloque Python block facilita la integración de stream tags en el flujo de trabajo de GNU Radio, permitiendo un procesamiento de señales más completo y eficiente.
- Los stream tags son pequeñas etiquetas que viajan junto a los datos en un flujo de datos paralelo al principal, llevando información adicional sobre esos datos, como lo que representan y de dónde vienen.
- los stream tags proporcionan una manera poderosa de agregar metadatos a los flujos de datos en GNU Radio, aunque su implementación puede añadir complejidad al sistema.

4. Referencias

[1] wiki.gnuradio, «wiki.gnuradio,» 17 agosto 2023. [En línea]. Available: https://wiki.gnuradio.org/index.php?title=Stream_Tags. [Último acceso: 2024 03 10].

[2] O. R. T. Homero Ortega, Comunicaciones Digitales basadas en radio definida por software., bucaramanga: Publicaciones UIS, 2019.