



GUÍA N° 2.2 – ADQUISICIÓN DE SEÑALES Y GRAFICACIÓN EN ARDUINO

1. OBJETIVOS ESPECÍFICOS DE LA PRÁCTICA

- Adquirir señales conocidas como señal cuadrada, triangular, senoidal, rampa, etc.
- Entender los criterios de selección de la frecuencia de muestreo.
- Manipular y configurar adecuadamente una fuente de alimentación regulable; multímetro digital; Generador de señales y osciloscopio digital.

2. MATERIALES Y EQUIPOS

Equipo Materiales		Cantidad
Modelo	Descripción	
AFG1022	Generador de Señales	1
TBS 1000C Series	Osciloscopio Digital	1
-	Cable BNC Male-Male	1
-	Punta de osciloscopio con conector BNC (Male)	1
-	Par de cables Male -Male	1
SAMD	Arduino 33 IoT	1

3. PROCEDIMIENTO

- Encender el Generador de Señales y el Osciloscopio
- Configurar el Generador de Señales para proporcionar una señal sinusoidal de 1 KHz de frecuencia, 1.5V de Amplitud y 0V de offset, por el canal 1.
- Conectar un extremo del cable BNC en el canal 1 del generador de señales y el otro extremo en el canal 1 del osciloscopio.

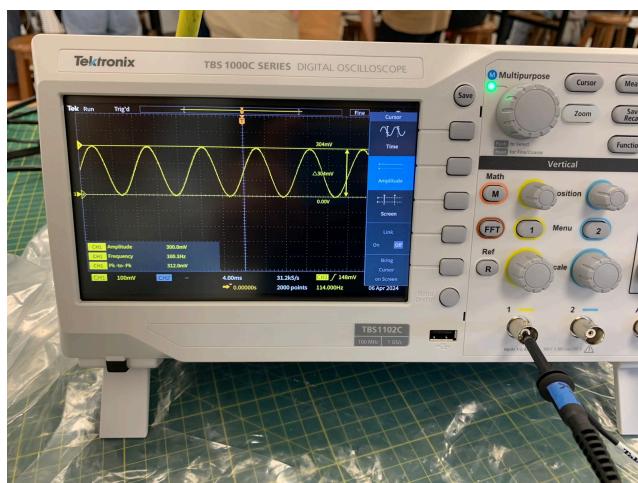


Fig 1. Conexión del canal del generador de señales

- Mediante los controles de Posición Vertical, Horizontal y Disparo ajustar la visualización de la señal sinusoidal.

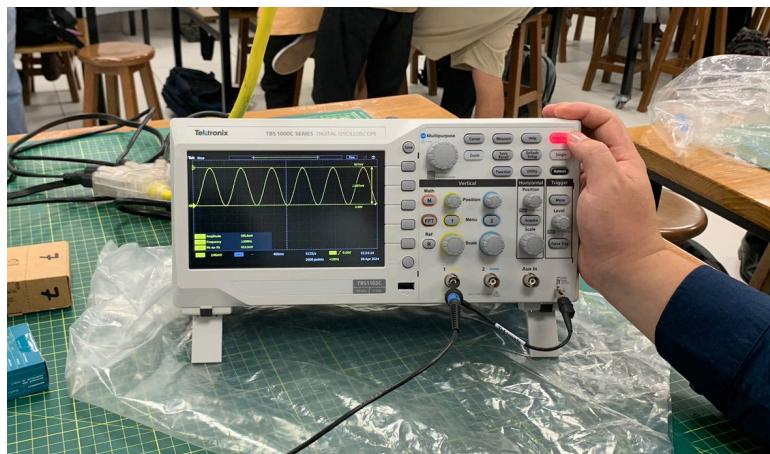


Fig 2. Ajustar la visualización

- Haciendo uso de los cursores, calcular y mostrar en el osciloscopio las medidas de Amplitud y Periodo de la señal

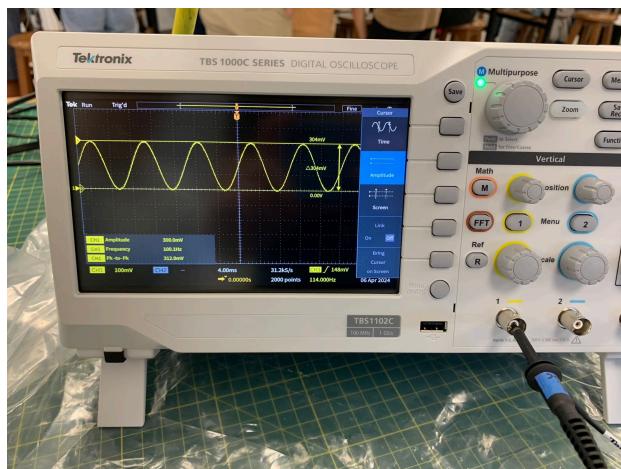


Fig 3. Mostrar medidas

Al realizar la verificación de la señal que sale por la sonda del generador de señal, se observa que a pesar de configurar un voltaje de salida de 1.5V con 0V de offset, la señal es atenuada por la sonda de osciloscopio.

- Conexión generador de señales y Arduino nano 33 IoT

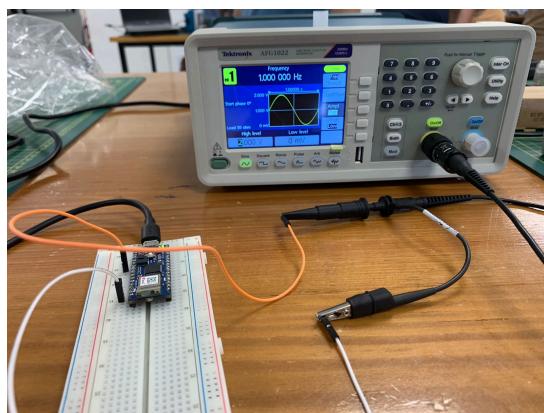


Fig 4. Conexión del arduino sin condensador

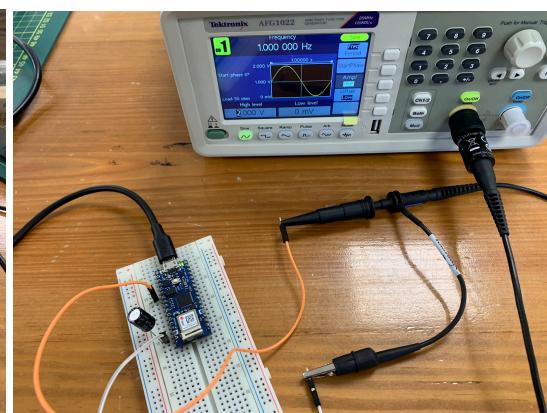


Fig 5. Conexión del arduino con condensador

La conexión inicialmente se realizó utilizando dos jumpers, sin embargo, en las gráficas pleteadas se observaba bastante ruido que impedía distinguir las señales.

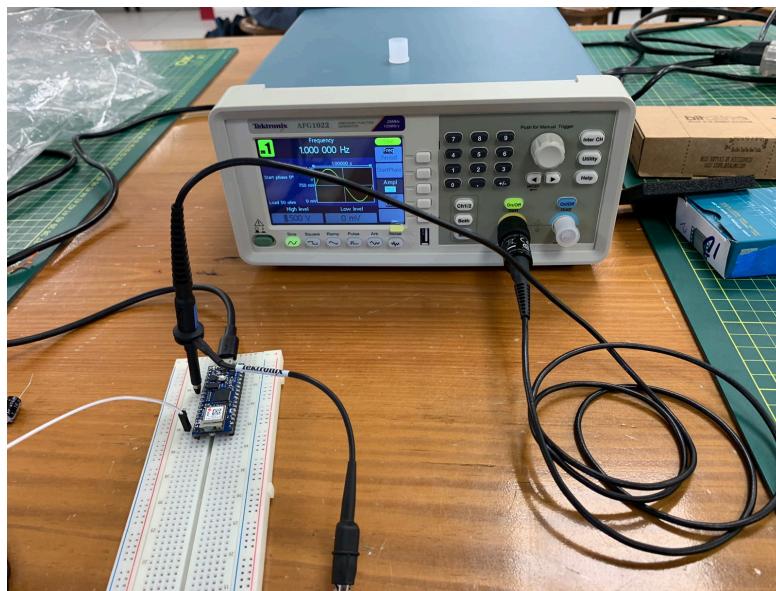


Fig 6. Conexión final

Tras realizar varias pruebas, se observó que de esta manera se mejoraba la conexión del arduino nano 33 IoT con el generador de señales.

- Ploteo de señales



Fig 7. Plot de la señal sin condensador

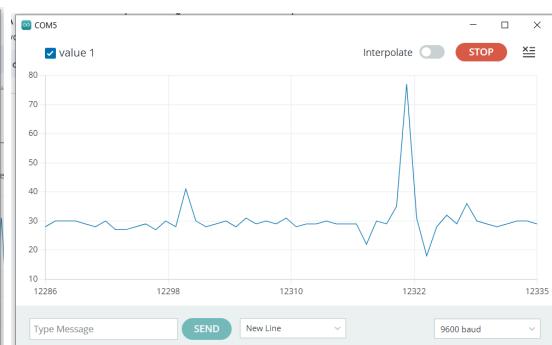


Fig 8. Plot de la señal con condensador

Con las conexiones hechas se obtuvo el ploteo de las señales. Inicialmente, se realizó sin conectar el condensador y luego con el condensador. El condensador actúa formando un circuito RC con la resistencia interna del arduino, el filtro es del tipo pasa altas.

4. ENTREGABLE

- Plotear al menos 3 señales en Arduino IDE provenientes del generador de señales.



Fig 9. Plot señal sinusoidal

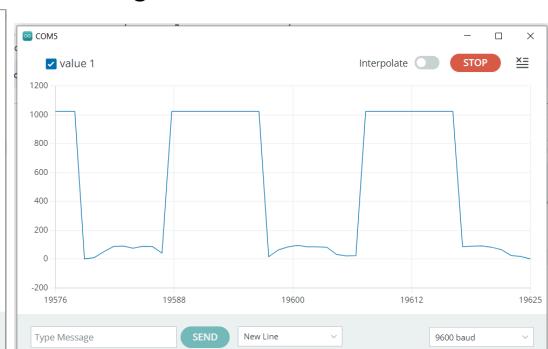


Fig 10. Señal cuadrática

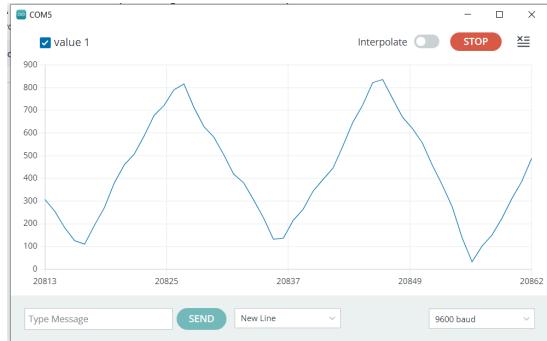


Fig 11. Plot señal triangular

Se plotearon las 3 señales (Fig 9-11) provenientes del generador de señales en el Arduino IDE y se obtuvieron gráficas que representan el tipo de señal generada.

- Comparar las señales graficadas del Arduino IDE con las gráficas obtenidas del osciloscopio.

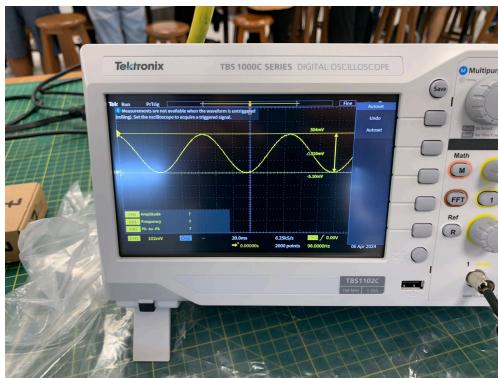


Fig 12. Señal sinusoidal del generador de señales



Fig 13. Señal cuadrática del generador de señales

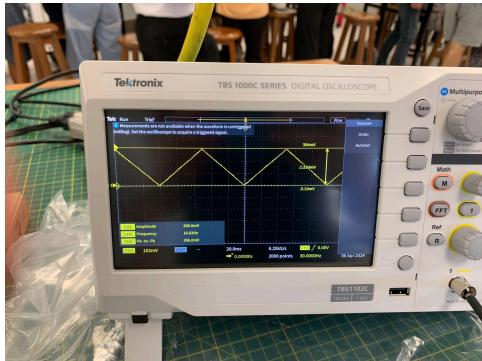


Fig 14. Señal triangular del generador de señales

Las señales obtenidas ploteando en el Arduino IDE con las gráficas del osciloscopio, se observa que las señales en el Arduino IDE presentan un cierto ruido en las señales comparado a las señales vistas desde el osciloscopio. La diferencia entre estas señales se puede deber a que se utilizó un jumper para establecer la conexión. Durante la experiencia se utilizaron dos jumpers y el ruido disminuyó al conectar una punta de la sonda directamente al protoboard sin utilizar el jumper. Además, se pueden generar interferencias por la presencia de otros equipos electrónicos presentes en la mesa

- Graficar en Arduino cloud.

```

1 unsigned long lastMsg = 0;
2 float F=1; // 1 hz
3 double Fs = 20*F; // 20 hz
4 double Ts_ms = (1/Fs)*1000; // 50 ms
5
6 void setup() {
7   Serial.begin(9600);
8   while (!Serial);
9 }
10
11 void loop() {
12
13   unsigned long now = millis();
14
15   if (now - lastMsg > Ts_ms) {
16     lastMsg = now;
17
18     int r1 = analogRead(A0); //Leer el valor en A0
19     //Serial.print("Señal1:");
20     Serial.println(r1);
21
22   }
23
24 }
25
26

```

Fig 15. Código en Arduino cloud.

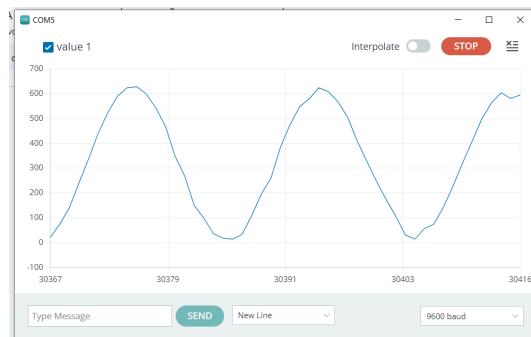


Fig 16. Gráfica sinusoidal de frecuencia 1 Hz, frecuencia de muestreo 20 Hz

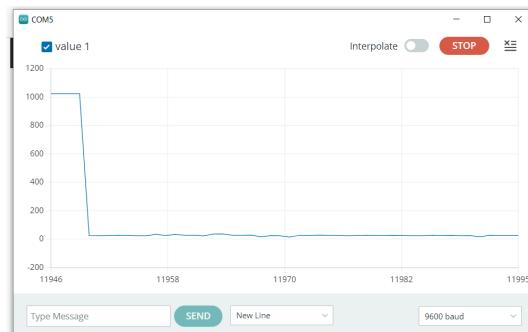


Fig 17. Gráfica cuadrática de frecuencia 1 Hz, frecuencia de muestreo 20 Hz y condensador (al inicio)

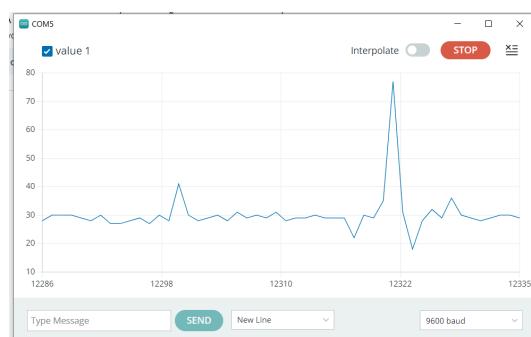


Fig 18. Gráfica cuadrática de frecuencia 1 Hz, frecuencia de muestreo 20 Hz y condensador (ya con tiempo)

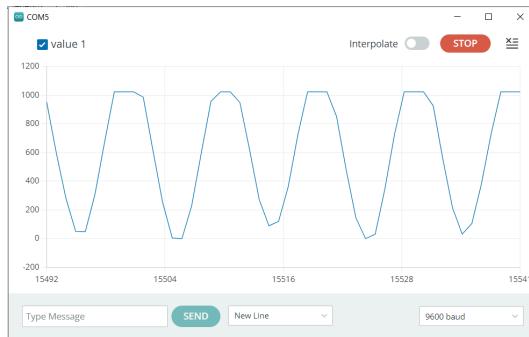


Fig 19. Gráfica sinusoidal de frecuencia 1 Hz, frecuencia de muestreo 10 Hz

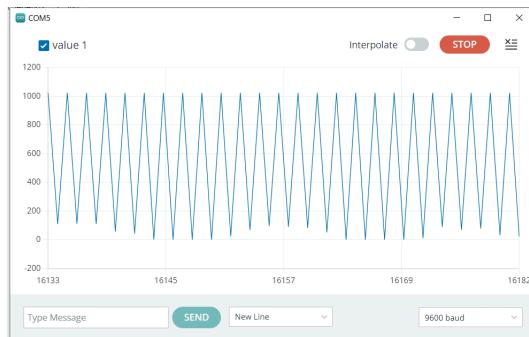


Fig 20. Gráfica sinusoidal de frecuencia 1 Hz, frecuencia de muestreo 2 Hz



Fig 21. Gráfica sinusoidal de frecuencia 500 Hz, frecuencia de muestreo 20 Hz con capacitor

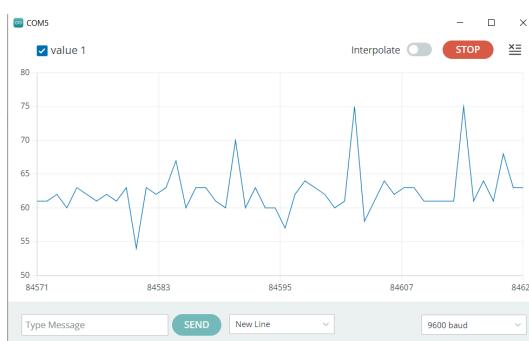


Fig 22. Gráfica sinusoidal de frecuencia 10 Hz, frecuencia de muestreo 20 Hz con capacitor

Observaciones e Interpretaciones:

- Condensador en el circuito:

El condensador en el circuito estaría actuando como un filtro pasa alta, ya que permite el paso de frecuencias más altas mientras atenúa las frecuencias más bajas. Así afecta la forma en que se detecta y registra la señal en el Arduino por el generador. En el laboratorio contamos con un condensador de valores muy altos que no facilitaron la visualización para ver la frecuencia de corte, ya que era de 16V y 470 uF. Necesitaríamos la resistencia interna y se calcularía: frecuencia de corte = $1/(2 \times \pi \times \text{Resistencia} \times \text{Capacitancia})$

- Sin con el condensador:

La señal captada por el Arduino tendrá no presentará ninguna atenuación extra en las frecuencias bajas, entonces se conserva la señal original del generador de ondas.

- Filtro pasa alta frecuencia:

Significa que permite el paso de frecuencias altas y atenúa las frecuencias más bajas. Esto puede ser útil para eliminar componentes de baja frecuencia no deseados o ruido de la señal captada por el Arduino.

- Frecuencia:

Al variar la frecuencia de la señal de entrada nos dimos cuenta que esta afecta la forma en que se captura y registra en el Arduino para la tomas de datos. Frecuencias más altas pueden requerir una frecuencia de muestreo más alta para capturar con precisión la forma de onda completa. Es por eso que requerimos un mayor valor que el doble de la frecuencia máxima para tener una mejor visualización de ploteo.

- Atenuación:

La atenuación de la señal ocurre debido a la resistencia interna del generador de ondas, la resistencia de los cables y otras resistencias en el circuito. La presencia del condensador también influye en la atenuación de la señal.

- Ruido del cable:

El ruido del cable se visualiza por la interferencia electromagnética ambiental o a la capacitancia entre los cables de señal y tierra. Se manifiesta como fluctuaciones aleatorias en la señal capturada por el Arduino.

- Cables BNC:

La boquilla, la calidad y la longitud de los cables BNC afectan la integridad de la señal capturada por el Arduino. Así inferimos que sin boquilla y con cables de alta calidad se minimiza la atenuación y la interferencia no deseada.

- Comparar con el osciloscopio:

Es importante comparar la señal capturada por el Arduino con la señal observada en un osciloscopio para verificar la precisión y la integridad de la captura de datos. El osciloscopio en el laboratorio nos sirvió de guía para proporcionar una representación visual más precisa de la forma de onda y puede revelar detalles sutiles que pueden perderse en la lectura digital del Arduino, por ejemplo con los datos del voltaje