



## Seguimiento 3: Análisis en sistemas embebidos

Danna Isabella García Saenz - Solanlly Evenedy Montoya Rivera

Mayo 9, 2024

---

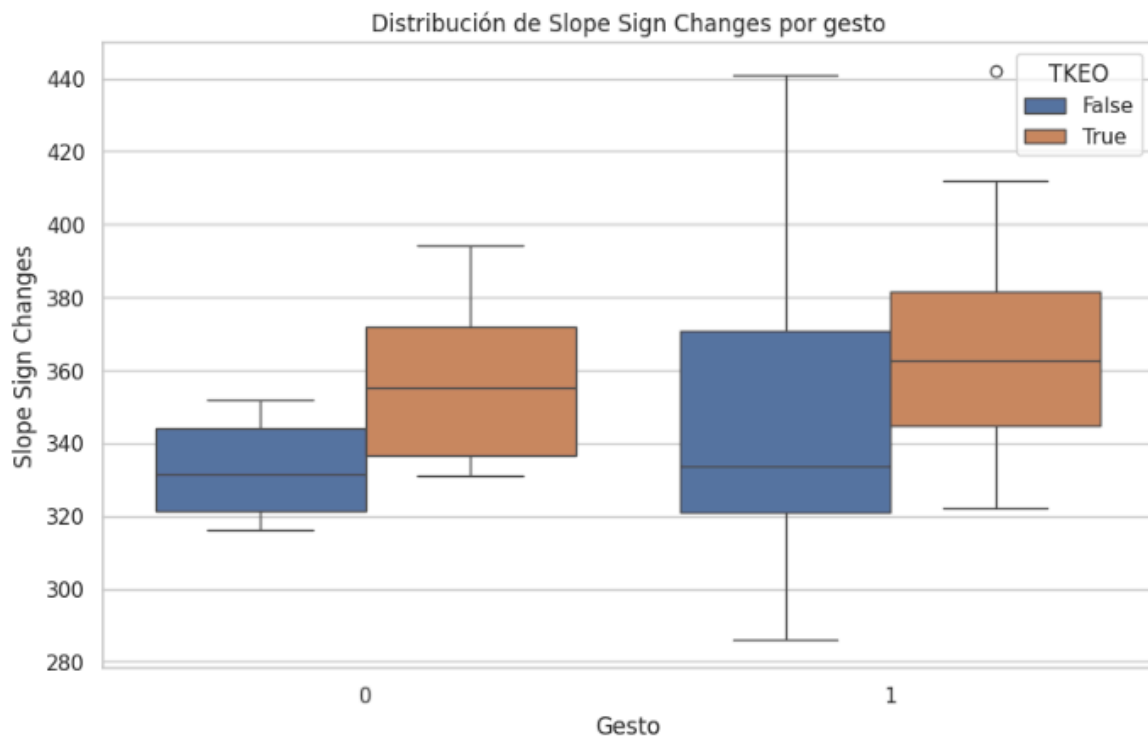
### Introducción

La electromiografía (EMG) es una técnica crucial en el estudio y análisis de la actividad eléctrica de los músculos, brindando una ventana única hacia los procesos neuromusculares durante la contracción y relajación muscular. Esta actividad eléctrica se traduce en potenciales eléctricos generados por las fibras musculares, los cuales pueden ser detectados mediante electrodos de superficie colocados estratégicamente sobre la piel que recubre el músculo de interés [1]. La capacidad de capturar y procesar estas señales EMG permite interpretar con precisión los movimientos musculares, ofreciendo valiosa información sobre la producción de fuerza y el patrón de movimiento.

En este contexto, el presente trabajo se enfoca en el desarrollo de un sistema embebido al menor costo que pueda adquirir, acondicionar y procesar señales EMG del movimiento de extensión y flexión del codo en tiempo real. El objetivo principal es diferenciar de manera fiable entre los movimientos de flexión y extensión del codo, utilizando índices numéricos previamente definidos para mejorar la discriminación entre estos movimientos de interés, permitiendo la detección precisa y confiable del movimiento generado por la articulación de interés, visualizando el resultado a través de un indicador LED, en donde el LED encendido corresponderá a la flexión, mientras que el LED apagado indicará la extensión del codo.

**Programar en el microcontrolador las rutinas que permiten una mejor discriminación de movimientos encontradas en el proyecto 1 y 2. Tener en cuenta las limitaciones de procesamiento que hay en los microcontroladores**

En el proyectos anterior se evaluaron diferentes métricas y características para discriminar de manera efectiva los movimientos de flexión y extensión a partir de las señales EMG adquiridas y se determinó que la mejor rutina para realizar la discriminación de los movimientos de flexión y extensión del brazo fue mediante el cálculo de los cambios de signo de la pendiente Slope Sign Changes, debido a la gráfica del proyecto 2, en donde, como se puede observar en la gráfica 1, hay una gran apreciación entre gesto, ya fuera con el operador TKEO o sin operador TKEO, en comparación con otras métricas como el RMS o el MAV.



**Gráfica 1.** Boxplot del SSC de los gestos de flexión (1) y extensión (0) con el operador TKEO y sin TKEO del proyecto 2

La principal razón para elegir el slope sign changes radica en su simplicidad computacional y su capacidad para capturar los cambios abruptos en la dirección de la señal EMG asociados con la flexión y extensión, este detecta estos cambios de signo en la pendiente, lo que permite una discriminación de movimientos de una manera mas efectiva, demás su calculo implica operaciones aritméticas más sencillas, lo que lo convierte en una opción más adecuada para su implementación en sistemas embebidos con recursos limitados como los microcontroladores.

```
int calcularSlopeSignChanges() {
    int slopeChanges = 0;
    for (int i = 1; i < numDatos; i++) {
        int slope = datos[i] - datos[i - 1];
        if ((slope > 0 && datos[i] < 512) || (slope < 0 && datos[i] > 512)) {
            slopeChanges++;
        }
    }
    return slopeChanges;
}
```

Se implementó una función que recorre el vector datos y calcula la pendiente (slope) entre cada par de muestras consecutivas, después verifica si la pendiente cambia y si el valor de la muestra actual está por encima o por debajo de un umbral, aprovechamos el hecho de que los movimientos de flexión y extensión del brazo producirán cambios en la pendiente de la señal EMG, en el movimiento de flexión la señal aumentará y por otro lado durante la extensión la señal disminuirá, el código revisa los cambios y además utiliza un umbral para evitar contar cambios de pendiente menores que podrían deberse a ruido o artefactos en la señal.

```
if (slopeChanges > 48) {
    digitalWrite(ledPIN, HIGH); // Encender LED
}
else {
    digitalWrite(ledPIN, LOW); // Apagar LED
}
}
```

El valor final de slopeChanges se utiliza en el loop() principal para encender o apagar el LED dependiendo de si este valor supera un cierto límite (48 en este caso).

Dentro de las limitaciones de procesamiento que hay que tener en cuenta sabemos que el umbral de 512 y el límite de 48 cambios de pendiente son valores no precisos que pueden no ser óptimos para todos los casos y pueden requerir ajustes dependiendo de la calidad de la señal EMG y las características individuales del usuario, sin embargo este enfoque puede ser un buen punto de partida y puede ser mejorado en el futuro mediante la incorporación de técnicas adicionales de procesamiento y un proceso de calibración más acorde.

### **Diseñar una estrategia de filtrado en Python, frecuencia de paso, de rechazo, atenuación/orden del filtro, etc, que se pueda implementar en el microcontrolador teniendo en cuenta las limitaciones de procesamiento. Implementarla en el microcontrolador**

Se realizó la implementación un filtro Butterworth de paso alto de 9º orden diseñado para atenuar las señales de baja frecuencia, en el contexto de la adquisición de señales EMG, este filtro podría ser beneficioso para eliminar componentes de baja frecuencia no deseados, preservando al mismo tiempo los componentes de alta frecuencia asociados a la actividad muscular.

La elección de un filtro de orden 9 se basa en la complejidad de la implementación del filtro, sabemos que los filtros de orden superior proporcionan una transición más pronunciada, lo que da como resultado una mejor separación de los componentes de frecuencia deseados y no deseados, sin embargo también requieren más recursos informáticos que en nuestro caso son limitados, con un orden de 9 se consigue un equilibrio razonable, ya que ofrece una atenuación relativamente pronunciada sin dejar de ser práctico.

```
from scipy.signal import iirfilter, bode
import matplotlib.pyplot as plt
import numpy as np

#especificaciones filtro
N = 9 #orden
Wn = 0.16 #frecuencia de corte (Hz)
fs = 100 #frecuencia de muestreo (Hz)
b, a = iirfilter(N, Wn, btype='high', ftype='butter', fs=fs) #diseño (filtro IIR pasa altas)

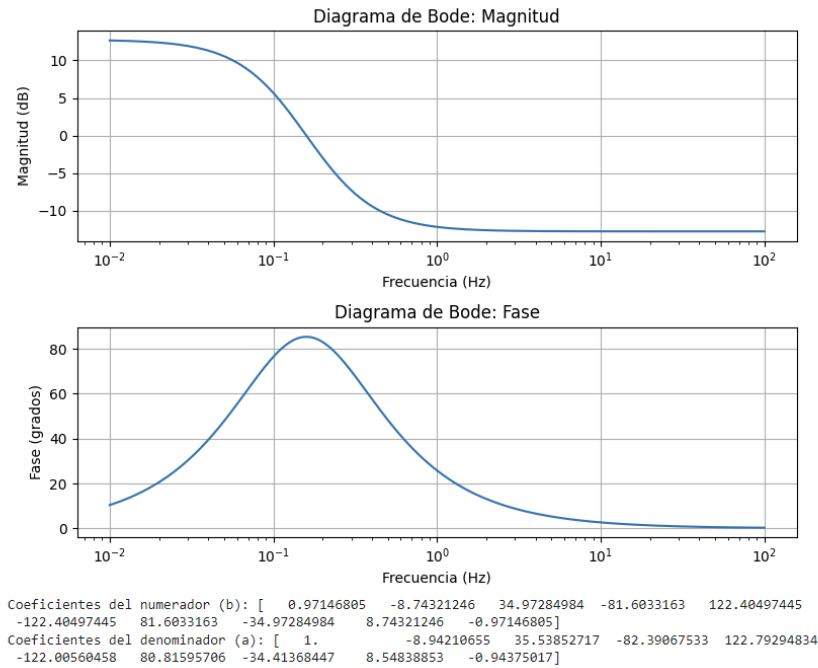
#diagrama de Bode
w, mag, phase = bode(iirfilter(N, Wn, btype='high', ftype='butter'), w=np.logspace(-2, 2, 1000) * 2 * np.pi * fs / 100)

#grafico el diagrama de Bode
fig, axs = plt.subplots(2, 1, figsize=(8, 6))
axs[0].semilogx(w / (2 * np.pi), mag)
axs[0].set_title('Diagrama de Bode: Magnitud')
axs[0].set_xlabel('Frecuencia (Hz)')
axs[0].set_ylabel('Magnitud (dB)')
axs[0].grid(True)

axs[1].semilogx(w / (2 * np.pi), phase)
axs[1].set_title('Diagrama de Bode: Fase')
axs[1].set_xlabel('Frecuencia (Hz)')
axs[1].set_ylabel('Fase (grados)')
axs[1].grid(True)
plt.tight_layout()
plt.show()

#coeficientes del filtro
print("Coeficientes del numerador (b):", b)
print("Coeficientes del denominador (a):", a)
```

*Figura 1. Código implementado para el diseño del filtro*



**Figura 2.** Diagrama de bode y coeficientes obtenidos

El filtro en general es una elección adecuada para la adquisición de señales EMG ya que atenúa eficazmente el ruido de baja frecuencia y los artefactos, al tiempo que permite el paso de las frecuencias relevantes de la actividad muscular que suele encontrarse en el rango de frecuencias de 20 Hz.

Posteriormente empezamos la implementación del filtro en el código arduino

```
const int order = 9; //orden del filtro IIR

float b[order+1] = {0.97146805, -8.74321246, 34.97284984, -81.6033163, 122.40497445, -122.40497445, 81.6033163, -34.97284984, 8.74321246, -0.97146805};
float a[order+1] = {1.0, -8.94210655, 35.53852717, -82.39067533, 122.79294834, -122.00560458, 80.81595706, -34.41368447, 8.54838853, -0.94375017};
```

```
float x[order+1] = {0}; //buffer de entrada
float y[order+1] = {0}; //buffer de salida
```

Los arrays b y a contienen los coeficientes del numerador y denominador respectivamente del filtro IIR pasa altas de orden 9, así mismo los arrays x y se utilizan para almacenar las muestras de entrada y salida del filtro, el tamaño de estos buffers es order+1 debido a que el filtro requiere mantener un historial de muestras pasadas para calcular la salida actual.

```
void applyIIRFilter() {
    float xn = datos[ultPos]; //nuevo dato de entrada
    float yn = 0; //salida filtrada

    //para actualizar el buffer de entrada
    for (int i = order; i >= 1; i--) {
        x[i] = x[i-1];
    }
    x[0] = xn;

    //calcula salida filtrada
    for (int i = 0; i <= order; i++) {
        yn += b[i] * x[i];
    }

    for (int i = 1; i <= order; i++) {
        yn -= a[i] * y[i];
    }

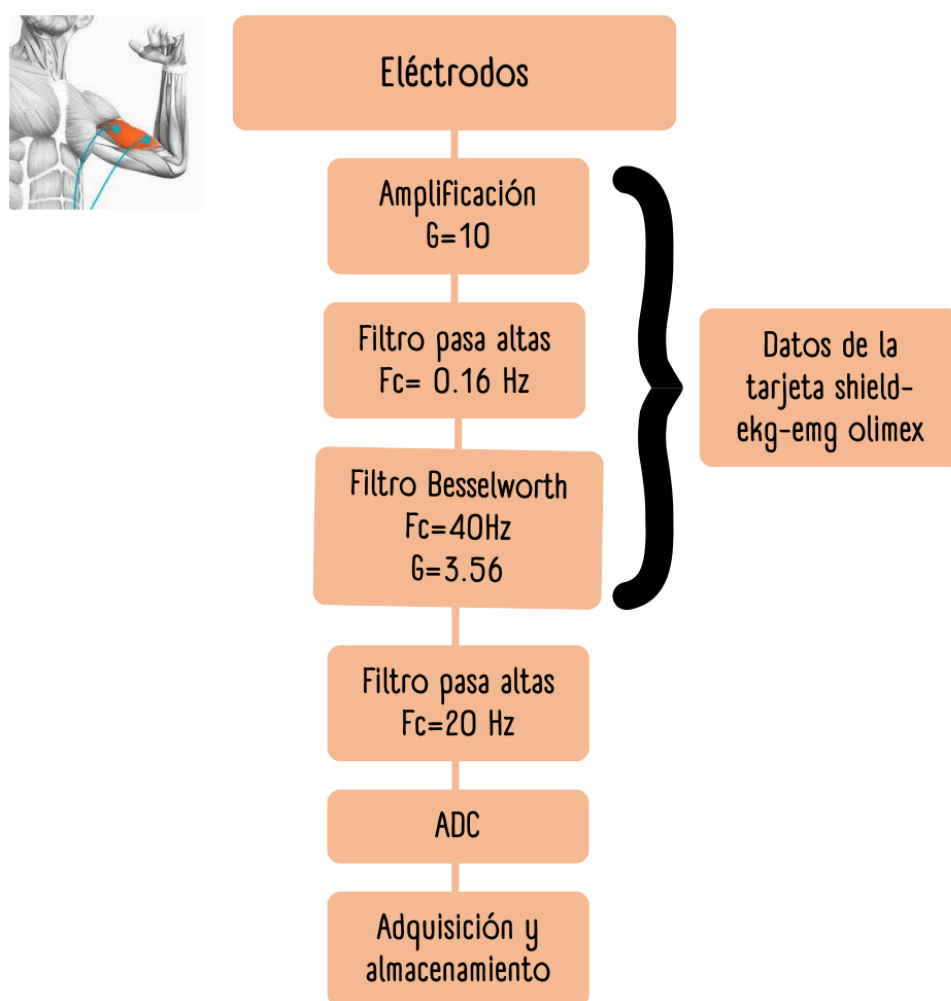
    //para actualizar buffer de salida
    for (int i = order; i >= 1; i--) {
        y[i] = y[i-1];
    }
    y[0] = yn;

    //yn como la salida filtrada
    datos[ultPos] = static_cast<int>(yn); // Reemplazar dato de entrada con salida filtrada
}
```

Se implementa una función donde se actualizan los buffers de entrada y salida con las nuevas muestras, se calcula la salida filtrada utilizando los coeficientes b (numerador), a (denominador) y las muestras en los buffers y se reemplaza el dato de entrada con la salida filtrada.

**Definir el diagrama de flujo de adquisición, filtrado, procesamiento de la señal EMG y generación de la salida del microcontrolador para indicar si hubo flexión o extensión. Justificar cada operación en el diagrama de flujo y cuál podría haber sido su alternativa.**

Se definieron 2 diagramas de flujo, en donde la Figura 3 corresponde al primer diagrama relacionado al registro de los potenciales de la señal EMG, en donde se especifican las etapas realizadas en la tarjeta shield ekg-emg olimex, el cual compete a las etapas de amplificación y filtrado, después se detalla el filtrado correspondiente al filtro pasa altas realizado por las estudiantes con una frecuencia de corte de 20 Hz debido a las frecuencias relacionadas a la señal muscular de interés.

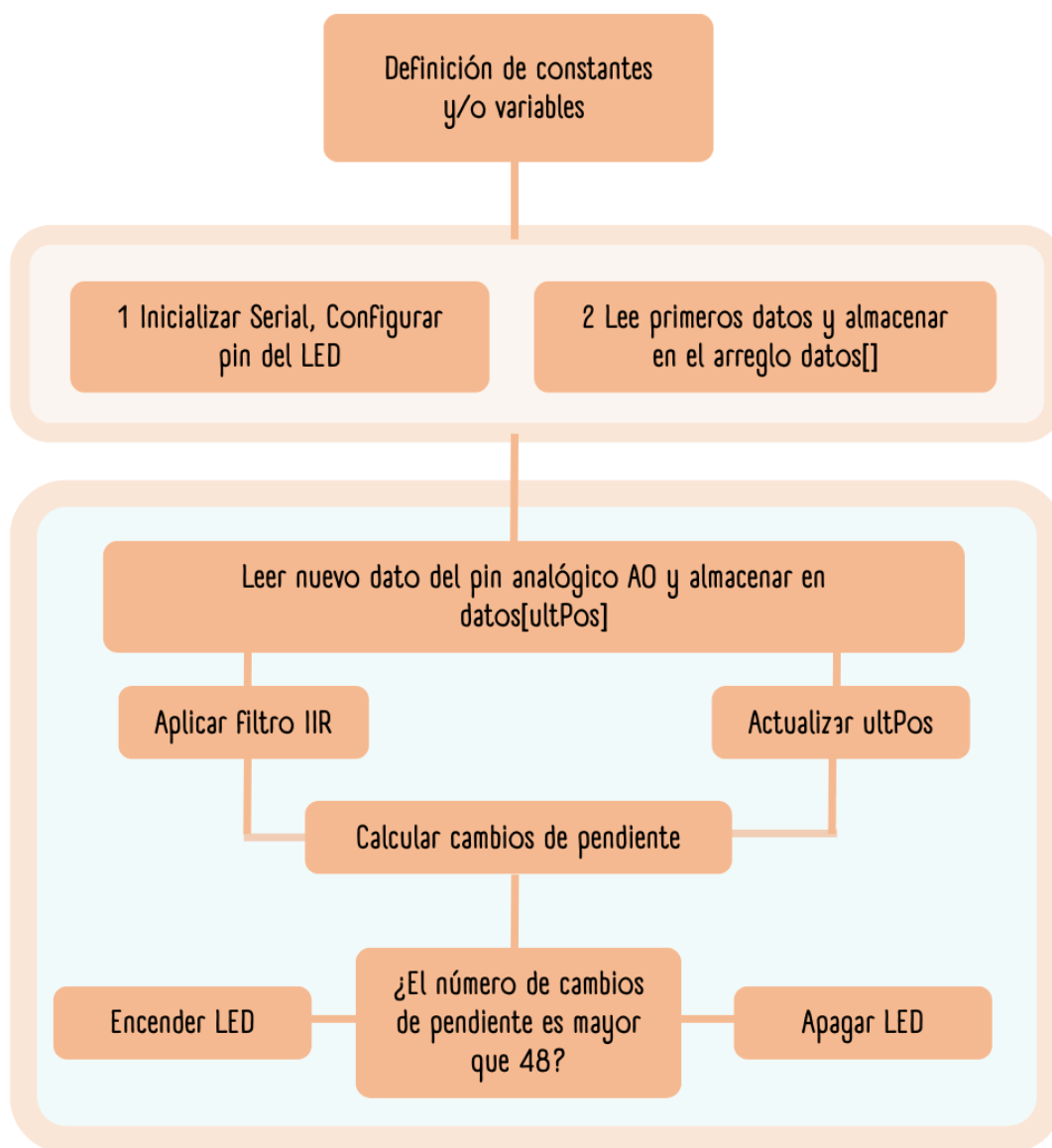


**Figura 3.** Diagrama de flujo de registro de los potenciales de la señal EMG

El segundo diagrama representado en la Figura 4 describe el proceso completo que abarca desde la adquisición de señales hasta la generación de una salida por parte del microcontrolador para indicar si se ha producido una flexión o extensión, basándose en los datos de una señal electromiográfica (EMG), en el diagrama expuesto en la Figura 4

se detallaron 2 bloques grandes, siendo el primer bloque (color rosa) el setup() del código y el segundo bloque (color azul) correspondiente al loop() del código de arduino.

En el análisis realizado, se determinó que el criterio para distinguir entre flexión y extensión es el valor de SLOPE SIGN CHANGES (SSC) de la señal, estableciendo que los datos de flexión de SSC son mayores que 48, dicho valor se detalló por dos personas a las que se les estaba midiendo la señal de manera constante. En otras palabras, cuando se detecta una flexión, se activa un LED, mientras que si se detecta una extensión, se apaga el LED.



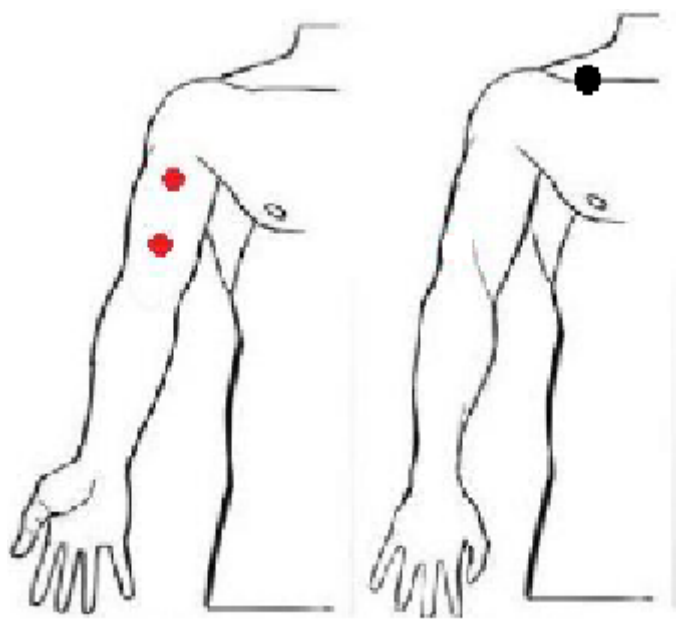
**Figura 4.** Diagrama de flujo de almacenamiento y procesamiento de la señal con la indicación de si hubo o no flexión de la articulación.

#### Implementar el flujo propuesto en el microcontrolador y hacer correcciones en caso de que fuera necesario

Se implementó el flujo propuesto en el microcontrolador y se hizo la corrección de la frecuencia de corte del filtro pasa altas de 20Hz a 16Hz para hallar los coeficientes del filtro IIR debido a que no se estaban realizando las diferencias de los valores de SSC adecuadas cuando el brazo está en flexión o extensión.

**Hacer un informe de funcionamiento de la propuesta donde se pueda verificar en 20 ensayos de flexión / extensión en dos sujetos diferentes para cada sujeto cuantas veces se detecta correctamente el movimiento de flexión y cuantas veces el de extensión**

Para realizar el informe de funcionamiento de la propuesta donde se pueda verificar en 20 ensayos de flexión/extensión en dos sujetos para detectar cuántas veces se detalla un correcto funcionamiento del código para la correcta detección del movimiento, se escogieron dos personas para realizar los 20 movimientos, se limpió la zona de los electrodos con alcohol y se ubican los electrodos como se puede detallar en la Imagen 1.



***Imagen 1.** Ubicación de los electrodos*

Los 2 sujetos realizaron 20 ensayos cada uno, alternando entre movimientos de flexión y extensión del brazo, durante cada ensayo se registró el movimiento realizado por cada sujeto. En el informe se determinó que el valor de extensión o flexión corresponde a 1 si el led tuvo un funcionamiento correcto y 0 si no se dio un correcto funcionamiento.

Los datos de los 20 ensayos realizados fueron recopilados en la Tabla 1 para el sujeto 1 y la Tabla 2 para el sujeto 2, como se muestra a continuación.

Intento	Extensión	Flexión	Intento	Extensión	Flexión
1	1	1	11	1	1
2	1	1	12	1	1
3	1	1	13	1	1
4	0	1	14	0	1
5	0	1	15	1	1
6	0	1	16	1	0

<b>7</b>	1	1	<b>17</b>	1	1
<b>8</b>	1	1	<b>18</b>	1	1
<b>9</b>	1	1	<b>19</b>	1	1
<b>10</b>	0	1	<b>20</b>	0	0

*Tabla 1. Valores ensayo persona 1*

En los resultados de la tabla 1 para los movimientos de flexión y extensión de la persona 1 se observó que se detallaron que se dieron 4 fallos de 20 ensayos para la detección del movimiento de flexión y 6 fallos de 20 ensayos para la detección del movimiento de extensión.

<b>Intento</b>	Extensión	Flexión	<b>Intento</b>	Extensión	Flexión
<b>1</b>	1	1	<b>11</b>	1	0
<b>2</b>	1	1	<b>12</b>	1	0
<b>3</b>	1	1	<b>13</b>	1	1
<b>4</b>	0	1	<b>14</b>	1	1
<b>5</b>	1	1	<b>15</b>	1	1
<b>6</b>	1	1	<b>16</b>	1	0
<b>7</b>	1	1	<b>17</b>	0	0
<b>8</b>	0	1	<b>18</b>	0	0
<b>9</b>	1	0	<b>19</b>	1	1
<b>10</b>	1	1	<b>20</b>	1	0

*Tabla 2. Valores ensayo persona 2*

En los resultados de la tabla 1 para los movimientos de flexión y extensión de la persona 1 se observó que se detallaron que se dieron 7 fallos de 20 ensayos para la detección del movimiento de flexión y 4 fallos de 20 ensayos para la detección del movimiento de extensión.



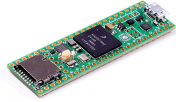

Para la persona 1, el sistema propuesto tuvo una tasa de acierto del 90% para la detección de flexión y del 70% para la detección de extensión, en el caso de la persona 2, la tasa de acierto fue del 65% para la detección de flexión y del 80% para la detección de extensión, las diferencias en el rendimiento entre los dos sujetos pueden deberse a varios factores, como la colocación de los electrodos, la intensidad de la señal EMG individual, o la variabilidad en los patrones de movimiento.

En conclusión la propuesta de detección de movimientos de flexión y extensión del brazo utilizando señales EMG tiene un buen rendimiento, con tasas de acierto altas en la mayoría de los casos evaluados, se pueden observar diferencias en el desempeño entre los sujetos, lo que sugiere la necesidad de ajustes adicionales y un mayor número



de pruebas para mejorar la precisión y consistencia del sistema, pero en general los resultados son prometedores y respaldan la viabilidad de la propuesta para su uso en aplicaciones prácticas

**Realizar un cuadro comparativo sobre microcontroladores comercialmente disponibles a nivel nacional teniendo en cuenta capacidades de memoria, velocidad de procesamiento, costo. De la tabla sustente cual podría ser la mejor opción para mejorar la solución desarrollada**

	ATmega328P (Arduino) [2]	ESP32 [3]	Teensy 4.1 [4]	PIC18F4525 [5]
<b>Imagen</b>				
<b>Memoria RAM</b>	2 kB	520 kB	512 kB	3.8kB
<b>Canales ADC</b>	8	18	42	12
<b>Memoria Flash</b>	32 kB	520 kB	256 kB	40 kB
<b>Velocidad de procesamiento</b>	16 MHz	240 MHz	600 MHz	200 MHz
<b>Costo</b>	25.000 COP	27.000 COP	30.00 COP	60.000 COP
<b>Voltaje de funcionamiento</b>	5v	5v	3.3v	5v
<b>Ventajas</b>	Fácil de usar, amplia gama de shields disponibles, ideal para proyectos de electrónica básica	Wi-Fi y Bluetooth, gran cantidad de canales ADC, alta velocidad de procesamiento	Alta velocidad de procesamiento, gran cantidad de RAM y soporte para SD card	Bajo costo, fácil de usar
<b>Desventajas</b>	memoria limitada y baja velocidad de procesamiento	Mayor complejidad de uso	Mayor complejidad y costo	Menor cantidad de canales ADC, menor velocidad de procesamiento y mayor costo

*Tabla 3. Comparación de diferentes microcontroladores*

El microcontrolador más adecuado para mejorar la solución desarrollada del proyecto EMG sería el Teensy 4.1. Este cuenta con alta velocidad de procesamiento, teniendo en cuenta que la electromiografía genera señales de alta frecuencia que requieren un procesamiento rápido, 600 MHz es el más rápido de los cuatro microcontroladores comparados, lo que lo convierte en la mejor opción para capturar, acondicionar y procesar estas señales de manera eficiente, el EMG también genera grandes cantidades de datos y la memoria RAM del Teensy es una de las más grandes, adicionalmente este microcontrolador admite tarjetas SD, lo que le permite ampliar su capacidad de almacenamiento y es compatible con el IDE de Arduino, lo que facilita el desarrollo de software para el proyecto.

**Hacer un pitch (video) , presentación de 3 min indicando la propuesta de valor desde la ingeniería, de la solución desarrollada y subirla a youtube.**

[https://youtu.be/G5rNnqqm0\\_c](https://youtu.be/G5rNnqqm0_c)

## **Conclusiones**

- El algoritmo Slope Sign Changes implementado en el microcontrolador mostró resultados prometedores para discriminar de forma fiable entre movimientos de flexión y extensión a partir de las señales EMG adquiridas. El algoritmo actualmente no tiene en cuenta la duración de los cambios de pendiente, lo que podría hacer que sea menos robusto y estable frente a señales EMG complejas o con mucho ruido.
- Comparando diferentes microcontroladores disponibles en el mercado, el Teensy 4.1 emerge como un firme candidato para mejorar señales de este tipo de proyectos, su almacenamiento y velocidad lo hacen idóneo para manejar con más eficacia las exigencias de adquisición y procesamiento de datos EMG.
- Para el proyecto la implementación de una calibración más avanzada en el futuro podría ser crucial para evitar problemas como los glitches en el comportamiento del LED, los cuales están actualmente afectando su precisión. Esta mejora podría lograrse mediante un ajuste más preciso del filtro, la ganancia del amplificador y otros parámetros relevantes, con el fin de optimizar la calidad de la señal adquirida.

## **Referencias**

- [1] Kasun Samarawickrama, Sadun Ranasinghe, Yasoja Wickramasinghe. Electromyography (EMG) signal acquisition and processing by using surface electrodes. (2019) Obtenido de:  
[https://www.researchgate.net/publication/333118571\\_Electromyography\\_EMG\\_signal\\_acquisition\\_and\\_processing\\_by\\_using\\_surface\\_electrodes](https://www.researchgate.net/publication/333118571_Electromyography_EMG_signal_acquisition_and_processing_by_using_surface_electrodes)
- [2] ATMEGA328P-PU SPDIP-28. Aylamp mechatronics. Obteniendo de:  
<https://naylorlampmechatronics.com/microcontroladores/111-atmega328p-pu-spdip-28.html>
- [3] ESP32. Aylamp mechatronics. Obteniendo de:  
<https://naylorlampmechatronics.com/espressif-esp/1250-nodemcu-32-30-pin-esp32-wifi-usb-c.html>
- [4] TEENSY 4.1. Aylamp mechatronics. Obteniendo de:  
<https://naylorlampmechatronics.com/ardusystem-tarjetas/860-teensy-41.html>
- [5] PIC18F4525-I/P. DigiKey. Obteniendo de:  
<https://www.digikey.es/es/products/detail/microchip-technology/PIC18F4525-I-P/613247>