GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

APLICACIÓN DE CONEXIÓN INALÁMBRICA Y ELABORACIÓN DE DATOS DE UN SENSOR ELECTROMIOGRÁFICO IMPLANTADO PARA CONTROL DE PRÓTESIS

Autor: Mariona Pros Fernández

Director: Romano Giannetti

Co-Director: José Daniel Muñoz Frías

Madrid

Declaración de originalidad

Declaro bajo mi responsabilidad que el Proyecto presentado con el título **Aplicación de conexión inalámbrica y elaboración de datos de un sensor electromiográfico implantado para control de prótesis** en la ETS de Ingeniería – ICAI de la Universidad Pontificia Comillas en el curso académico **4º de Ingeniería en Tecnologías de telecomunicaciones** es de mi autoría y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Uso de Inteligencia Artificial[1]

Declaro bajo mi responsabilidad que (indicar la opción correcta):

☐ No he utilizado Inteligencia Artificial en la elaboración del presente documento.

☒ He utilizado Inteligencia Artificial en la elaboración del presente documento y/o del Anexo B siempre en las condiciones permitidas por la Universidad Pontificia Comillas, es decir, aplicando el Nivel 2 de la Escala de Evaluación de Perkins et al. (2024): "*La IA puede utilizarse para actividades previas a la tarea, como la lluvia de ideas, la descripción y la investigación inicial. Este nivel se centra en el uso de la IA para la planificación, las síntesis y la generación de ideas, pero las evaluaciones deben hacer hincapié en la capacidad de desarrollar y refinar estas ideas de forma independiente*". En concreto, las Inteligencia Artificial ha sido empleada para:

(indicar aquí el uso concreto que se ha hecho de la Inteligencia Artificial) Entender algunos de los conceptos relacionados con el contexto del proyecto. Aprender algunas funcionalidades del uso de la aplicación usada para el desarrollo de mi aplicación, MIT App Inventor. Reescribir de manera más formal algunas frases de la memoria del TFG y del Anexo B.

Firmado (alumno): Mariona Pros Fernández
Fecha:29/05/2026

Autorización para la entrega del Proyecto

El Director del Proyecto	El co-Director del Proyecto (si aplica)
FIRMADO: GIANNETTI ROMANO - ****3425* el día 29/05/2026 con un certificado emitido por AC FNMT Usuarios	Firmado por MUÑOZ FRIAS JOSE DANIEL - ***7384** el día 31/05/2026 con un certificado emitido por AC FNMT Usuarios
Fdo:	Fdo:
Fecha:	Fecha:

[1] Esta declaración se refiere al uso de la Inteligencia Artificial generativa para realizar los documentos del Proyecto (Anexo B y Memoria). No aplica a Proyectos donde, por su naturaleza, deban emplear inteligencia artificial como parte de los mismos (aplicación de técnicas de aprendizaje automático, redes neuronales, análisis de datos…)

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

APLICACIÓN DE CONEXIÓN INALÁMBRICA Y ELABORACIÓN DE DATOS DE UN SENSOR ELECTROMIOGRÁFICO IMPLANTADO PARA CONTROL DE PRÓTESIS

Autor: Mariona Pros Fernández

Director: Romano Giannetti

Co-Director: José Daniel Muñoz Frías

Madrid

Agradecimientos

Quisiera expresar mi más sincero agradecimiento a todas las personas que han contribuido de una u otra manera a la realización de este Trabajo de Fin de Grado.

En primer lugar, me gustaría agradecer a mis tutores por su orientación, apoyo y disponibilidad durante todo el desarrollo del proyecto. Sus consejos y experiencia han sido fundamentales para poder llevar a cabo este trabajo.

También quiero agradecer al equipo de investigación y a todas las personas implicadas en el proyecto global del sensor implantable, por permitirme formar parte de esta iniciativa y por proporcionarme el contexto y los recursos necesarios para desarrollar este trabajo. Asimismo, deseo agradecer a mis compañeros y amigos por su apoyo y motivación a lo largo de esta etapa académica.

Por último, quiero dedicar un agradecimiento especial a mi familia por su apoyo incondicional, su paciencia y su confianza durante todos estos años de formación.

A todos ellos, muchas gracias.

APLICACIÓN DE CONEXIÓN INALÁMBRICA Y ELABORACIÓN DE DATOS DE UN SENSOR ELECTROMIOGRÁFICO IMPLANTADO PARA CONTROL DE PRÓTESIS

Autor: Pros Fernández, Mariona.
Director: Giannetti, Romano.
Co-Director: Muñoz Frías, José Daniel
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

El presente proyecto consiste en el desarrollo de una aplicación para dispositivos móviles capaz de recibir las señales enviadas por un sensor electromiográfico implantable, denominado BluEMG, y representarlas en tiempo real en pantalla. Además, la aplicación permite la grabación, el almacenamiento y la exportación de los datos obtenidos por la tarjeta para su posterior análisis y procesamiento.

Palabras clave: Bluetooth BLE, aplicación móvil, sensor implantable, MIT App Inventor, señales electromiográficas, BluEMG.

1. Introducción

En los últimos años el desarrollo de prótesis controladas directamente mediante señales fisiológicas del propio cuerpo ha experimentado un avance significativo. La obtención y tratamiento de señales electromiográficas (EMG), generadas por la actividad eléctrica de los músculos, resultan de gran importancia para su futura aplicación en el control de prótesis.

Tradicionalmente, la captación de las señales electromiográficas se realizaba mediante electrodos superficiales. Sin embargo, la señal final obtenida no resulta lo suficientemente limpia debido a la cantidad de tejidos que debe atravesar hasta llegar a la superficie. [1]

Con el objetivo de obtener señales electromiográficas de mayor calidad y reducir la presencia de ruido e interferencias, el Instituto de Investigación Tecnológica de la Universidad Pontificia de Comillas en colaboración con el hospital de Getafe, ha desarrollado un sistema de adquisición implantable denominado BluEMG. Este sistema está diseñado para captar las señales que fluyen a través de colgajos mediante un sensor implantable, permitiendo una monitorización precisa y fiable. [2]

El sistema de adquisición dispone de cuatro canales diferenciales con una ganancia fija de 100 en cada uno de ellos, además de incorporar un amplificador de ganancia programable (PGA) que permite ajustar la amplificación según las necesidades de adquisición. Asimismo, el rango dinámico del sistema viene determinado por una señal de salida comprendida entre 0 y 2,5 V, con una referencia de 1,1 V, lo que permite adaptar la sensibilidad del sistema en función de la ganancia total aplicada.

Con el fin de hacer posible la implantación del sensor, se hace uso de una técnica quirúrgica innovadora, desarrollada y actualmente en proceso de patente bajo el nombre de VMDT. Esta técnica consiste en conectar la parte del nervio que todavía conserva funcionalidad a un colgajo de músculo sano, permitiendo utilizar la señal mioeléctrica generada en dicho músculo para el control de una prótesis. De este modo, es posible aprovechar la actividad nerviosa residual del paciente y transformarla en señales útiles para el accionamiento y control protésico.

El presente Trabajo de Fin de Grado se enmarca en este ámbito médico-tecnológico y tiene como objetivo el desarrollo de una aplicación para móvil capaz de establecer conexión inalámbrica con el sensor implantable, mostrar por pantalla en tiempo real las señales obtenidas y exportarlas en un archivo CSV.

2. Definición del proyecto

En el contexto mencionado en la introducción, la aplicación para móvil desarrollada tiene como objetivo ser capaz de reconocer todos los dispositivos cercanos que hagan uso de la tecnología de Bluetooth Low Energy (BLE) y mostrarlos por pantalla para permitir la conexión entre el móvil y el sensor de interés, denominado BluEMG.

Una vez establecida la conexión entre ambos dispositivos, la aplicación recibe los datos captados por el sensor y muestra por pantalla las señales electromiográficas correspondientes a los canales de adquisición del dispositivo.

Mientras se visualizan las señales procedentes del sensor por pantalla, hay la opción de poner la aplicación a grabar, para posteriormente exportar el archivo grabado, el cual contiene los datos de los canales seleccionados durante la grabación, y compartirlo a través de cualquiera de las aplicaciones del dispositivo móvil sobre el que se trabaja.

3. Descripción del modelo/sistema/herramienta

El sensor utilizado en este proyecto dispone de cuatro canales independientes mediante los cuales es capaz de captar las señales eléctricas generadas en los colgajos musculares reconectados a los nervios y transmitirlas inalámbricamente a la aplicación móvil mediante Bluetooth Low Energy. Gracias a esta arquitectura multicanal, el sistema permite adquirir simultáneamente distintas señales EMG, facilitando así un análisis más completo y una representación en tiempo real de la actividad muscular registrada.

La aplicación móvil desarrollada para interactuar con el sensor se divide principalmente en dos pantallas diferenciadas, cada una de ellas orientada a una fase concreta del funcionamiento del sistema.

1. Primera pantalla

La primera pantalla de la aplicación está destinada principalmente a la conexión y configuración inicial del dispositivo BluEMG. En la parte superior de esta pantalla se encuentra un botón cuya función es iniciar la búsqueda de dispositivos Bluetooth Low Energy cercanos disponibles para la conexión. Una vez iniciada la búsqueda, se muestra automáticamente una lista con todos los dispositivos BLE detectados en las proximidades del teléfono móvil.

Entre los dispositivos encontrados, el usuario deberá seleccionar el sensor BluEMG correspondiente. Una vez establecida correctamente la conexión, se muestran en la parte inferior de la pantalla diversos datos relevantes relacionados con el dispositivo, como información del fabricante, modelo, versión de firmware o estado de batería. Estos parámetros permiten comprobar rápidamente que la conexión se ha realizado correctamente y conocer el estado general del sensor antes de comenzar a trabajar con él.

Finalmente, esta pantalla dispone de un botón situado en la zona inferior denominado visualizar señal. Al pulsarlo, el usuario es redirigido a la pantalla principal de la aplicación, donde se llevará a cabo tanto la adquisición como la representación gráfica de las señales EMG.

Imagen 1: 1º pantalla desconectada

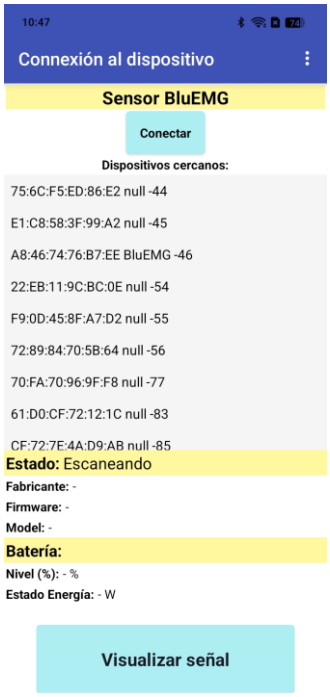


Imagen 2: 1º pantalla, escaneando

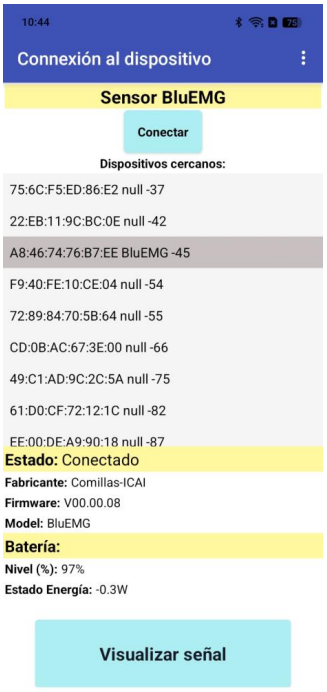


Imagen 3: 1º pantalla, conectada

2. Pantalla principal

La pantalla principal constituye el núcleo funcional de toda la aplicación, ya que en ella se concentra la mayor parte de la lógica desarrollada y es donde se realiza la visualización en tiempo real de las señales procedentes de los cuatro canales disponibles del sensor.

En la parte superior de la interfaz se encuentran 5 botones principales encargados de gestionar las distintas funcionalidades implementadas en la aplicación.

El primero de ellos es el botón *"Record"*, cuya función es iniciar la grabación de los datos adquiridos. Justo a su lado se encuentra el botón *"Stop"*, encargado de finalizar el proceso de grabación. A continuación, se sitúa el botón *"Gain"*, mediante el cual se despliega una lista con todas las opciones de ganancia disponibles para el sensor. El usuario puede seleccionar la configuración de ganancias que considere más adecuada en función de las características de la señal EMG que se está adquiriendo.

Debajo de estos 3 botones principales se encuentran los otros dos botones restantes: *"Export media"* y *"Share"*. Una vez finalizada la grabación, el usuario debe pulsar sobre *"Export media"* para generar automáticamente un archivo CSV con todos los datos registrados y almacenarlo en el dispositivo móvil. Posteriormente, mediante el botón *"Share",* es posible compartir dicho archivo utilizando cualquiera de las plataformas o aplicaciones disponibles en el sistema operativo del dispositivo, como aplicaciones de mensajería, correo electrónico o servicios de almacenamiento en la nube.

Además, la aplicación incorpora un campo de texto denominado *"Comment"*, que permite añadir comentarios o anotaciones personalizadas junto a los datos registrados. Esta funcionalidad resulta especialmente útil para identificar condiciones específicas de la prueba, observaciones clínicas o cualquier información relevante asociada a la sesión de adquisición.

Debajo de la zona correspondiente a la gestión de funcionalidades se encuentran los cuatro espacios gráficos destinados a la representación en tiempo real de las señales EMG, obtenidas de cada uno de los canales del sensor. Cada canal dispone de su propio Canvas, permitiendo visualizar de manera independiente la evolución temporal de las señales musculares adquiridas.

Por último, en la parte inferior de la pantalla principal se vuelven a mostrar algunos parámetros característicos del sensor, como el porcentaje de batería restante o información del dispositivo. Esto permite que el usuario pueda supervisar continuamente el estado del sensor mientras trabaja con la aplicación, garantizando así un control constante del funcionamiento del sistema.

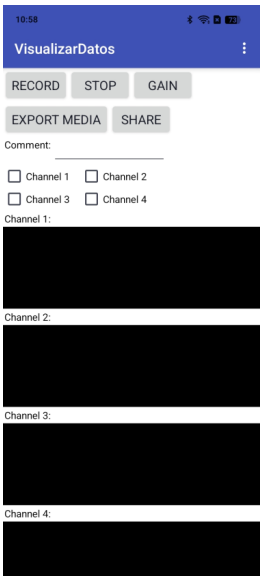


Imagen 4: Pantalla principal, parte superior

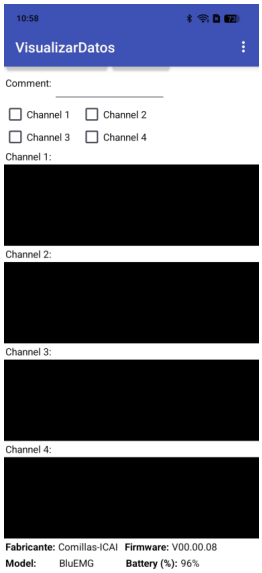


Imagen 5: Pantalla principal, parte inferior

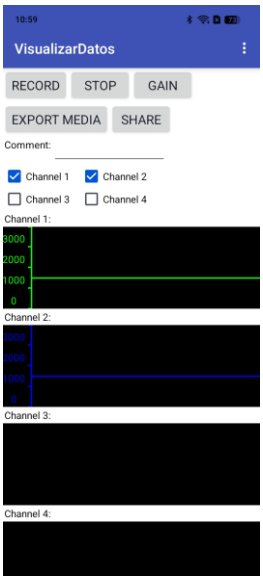


Imagen 6: Pantalla principal con 2 canales

Imagen 7: Desplegable opciones ganancia

4. Resultados

Una vez finalizado el desarrollo de la aplicación móvil, se realizaron distintas pruebas funcionales con el objetivo de comprobar el correcto funcionamiento de la comunicación entre el sensor BluEMG y el dispositivo móvil mediante Bluetooth Low Energy.

En primer lugar, se verificó el correcto funcionamiento del sistema de escaneo BLE. La aplicación fue capaz de detectar correctamente los dispositivos cercanos y permitir la conexión con el sensor blue MG punto. Además, cada dispositivo pudo diferenciarse mediante su dirección. Son m a c coma, facilitando la identificación cuando varios sensores se encontraban simultáneamente dentro del rango Bluetooth.

Una vez establecida la conexión, la aplicación recibió correctamente los datos enviados por el sensor y representó en tiempo real las señales electromiográficas correspondientes a los distintos canales de adquisición. Durante las pruebas realizadas se comprobó el funcionamiento simultáneo de los cuatro canales, observándose una representación gráfica fluida y estable de las señales EMG.

También se verificó el correcto funcionamiento del sistema de grabación y exportación de datos. La aplicación fue capaz de almacenar las muestras adquiridas y generar posteriormente archivos CSV compatibles con herramientas externas.

A continuación, se muestra un ejemplo simplificado del formato utilizado en los archivos CSV generados:

```
Sample,CH1,CH2,CH3,CH4
0,1520,1488,1512,1499
1,1531,1495,1508,1501
2,1540,1502,1515,1506
3,1535,1498,1510,1503
```

En general, los resultados obtenidos demuestran que la aplicación desarrollada cumple correctamente los objetivos planteados, permitiendo establecer comunicación inalámbrica con el sensor BluEMG, visualizar señales electromiográficas en tiempo real y exportar los datos para su posterior procesamiento.

5. Conclusiones

En conclusión, el proyecto ha permitido desarrollar una aplicación móvil capaz de adquirir, procesar, representar y almacenar señales electromiográficas procedentes de un sensor BluEMG mediante Bluetooth Low Energy. La aplicación no solo permite la visualización en tiempo real de las señales, sino que también incorpora funcionalidades como la configuración de canales y ganancia, la grabación de datos y la exportación y compartición de archivos CSV.

Además, el uso de MIT APP Inventor ha demostrado ser una herramienta adecuada para el desarrollo de aplicaciones biomédicas, permitiendo implementar de forma sencilla funcionalidades complejas relacionadas con la comunicación BLE el procesado de señales.

Finalmente, este trabajo demuestra la viabilidad de desarrollar soluciones móviles funcionales y accesibles para la monitorización de señales biomédicas, con posibles aplicaciones futuras en ámbitos como la rehabilitación, la investigación o el control protésico.

6. Referencias

[1] P. F. Pasquina et al., «First-in-man demonstration of a fully implanted myoelectric sensors system to control an advanced electromechanical prosthetic hand», J. Neurosci. Methods, vol. 244, pp. 85-93, abr. 2015, doi: 10.1016/j.jneumeth.2014.07.016.

[2] «IMPLANT DEVICE»

WIRELESS CONNECTION AND DATA PROCESSING APPLICATION FOR AN IMPLANTED ELECTROMYOGRAPHIC SENSOR FOR PROSTHETIC CONTROL

Author: Pros Fernández, Mariona.
Supervisor: Giannetti, Romano
Co-Supervisor: Muñoz Frías, José Daniel
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

This project consists of the development of a mobile application capable of receiving the signals sent by an implantable electromyographic sensor, called BluEMG, and displaying them on screen in real time. In addition, the application allows the recording, storage, and export of the data obtained from the board for subsequent analysis and processing.

Keywords: Bluetooth BLE, mobile application, implantable sensor, MIT App Inventor, electromyographic signals, BluEMG.

1. Introduction

In recent years, the development of prostheses directly controlled through psychological signals generated by the human body has experienced significant progress. The acquisition and processing of electromyographic signals generated by the electrical activity of muscles are of great importance for their future application in prosthetic control.

Traditionally, electromyographic signals were acquired using surface electrodes. However, the final signal obtained is not sufficiently clean due to the amount of tissue the signal must pass through before reaching the surface. [1]

With the aim of obtaining higher quality electromyographic signals and reducing the presence of noise and interference, the Instituto de Investigación Tecnológica of Universidad Pontificia Comillas, in collaboration with Getafe Hospital, has developed an implantable acquisition system called BluEMG. This system is designed to capture the signals flowing through muscle flaps by means of an implantable sensor, enabling precise and reliable monitoring. [2]

The acquisition system features four differential channels, each with a fixed gain of 100, and also incorporates a programmable gain amplifier (PGA) that allows the amplification to be adjusted according to the acquisition requirements. In addition, the dynamic range of the system is determined by an output signal ranging from 0 to 2.5 V, with a 1.1 V reference, making it possible to adapt the system sensitivity according to the total applied gain.

In order to make sensor implantation possible, an innovative surgical technique, developed and currently under patent application under the name VMDT, is employed. This technique consists of connecting the portion of the nerve that still retains functionality to a healthy muscle flap, allowing the myoelectric signal generated in that muscle to be used for prosthetic control. In this way, it is possible to take advantage of the patient's residual neural activity and transform it into useful signals for prosthetic actuation and control.

This Bachelor's thesis is framed within this technological field and aims to develop a mobile application capable of establishing a wireless connection with the implantable sensor, displaying the acquired signals on screen in real time, and exporting them into a CSV file.

2. Project Definition

Within the context described in the introduction, the objective of this mobile application is to recognize all nearby devices using Bluetooth Low Energy (BLE) technology and display them on screen in order to allow the connection between the mobile device and the sensor of interest, called BluEMG.

Once the connection between both devices has been established, the application receives the data acquired by the sensor and displays the electromyographic signals corresponding to the device's acquisition channels.

While the signals received from the sensor are displayed on screen, the application also allows the user to start a recording session in order to later export the recorded file, which

contains the data from the selected channels during the recording process, and share it through any application available on the mobile device being used.

3. Description of the Model/System/Tool

The sensor used in this project features four independent channels capable of capturing the electrical signals generated in muscle flaps reconnected to the nerves and transmitting them wirelessly to the mobile application via Bluetooth Low Energy. Thanks to this multi-channel architecture, the system can simultaneously acquire different EMG signals, enabling a more complete analysis and real-time representation of the recorded muscular activity.

The mobile application developed to interact with the sensor is mainly divided into two different screens, each one focused on a specific phase of the system's operation.

1) First Screen

   The first screen of the application is mainly intended for the initial connection and configuration of the BluEMG device. At the top of this screen, there is a button whose function is to start the search for nearby Bluetooth Low Energy devices available for connection. Once the search begins, a list containing all BLE devices detected near the mobile phone is automatically displayed.

   Among the detected devices, the user must select the corresponding BluEMG sensor. Once the connection has been successfully established, several relevant device parameters are displayed at the bottom of the screen, such as manufacturer information, model, firmware, or battery status. These parameters allow the user to quickly verify that the connection has been successfully established and to know the general status of the center before starting to work with it.

   Finally, this screen includes a button located at the bottom called *"Visualizar Señal"*. When pressed, the user is redirected to the main screen of the application, where both the acquisition and graphical representation of the EMG signals are carried out.

Image 1: First screen, disconnected

Image 2: First screen, scanning

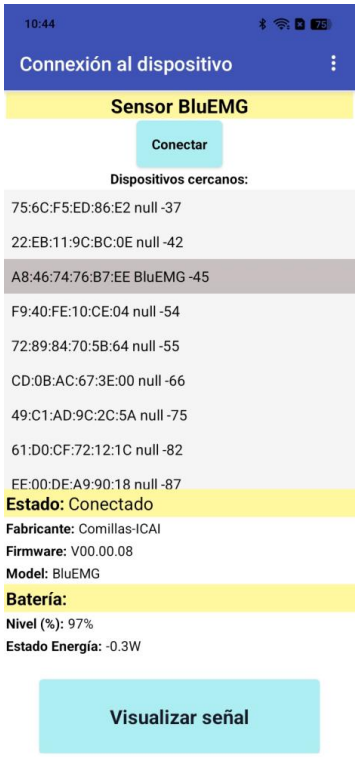


Image 3: First screen, connected

2) Main Screen

The main screen constitutes the functional core of the entire application, since it contains most of the developed logic and is where the real-time visualization of the signals coming from the four available channels of the sensor takes place.

At the top of the interface, there are five main buttons responsible for managing the different functionalities implemented in the application.

The first one is the *"Record"* button, whose function is to start recording the acquired data. Right next to it is the *"Stop"* button, responsible for ending the recording process. Next to it is the *"Gain"* button, through which a list containing all the available gain options for the sensor is displayed. The user can select the gain configuration considered most appropriate depending on the characteristics of the EMG signal being acquired.

Below these three main buttons are the remaining two buttons: *"Export media"* and *"Share"*. Once the recording has finished, the user must press "Export media" in order to automatically generate a CSV file containing all the recorded data and store it on the mobile device. Afterwards, using the share button, it is possible to share the generated file through any platform or application available on the device's operating system, such as messaging applications, e-mail, or cloud storage services.

In addition, the application includes a text field called *"Comment",* which allows the user to add personalized comments or annotations together with the recorded data. This functionality is especially useful for identifying specific testing conditions, clinical observations, or any relevant information associated with the acquisition session.

Below the functionality management section are four graphical areas dedicated to the real-time representation of the EMG signals acquired from each sensor channel. Each channel has its own Canvas, allowing the independent visualization of the temporal evolution of the acquired muscular signals.

Finally, at the bottom of the main screen, some characteristic sensor parameters are displayed again, such as the remaining battery percentage or device information. This allows the user to continuously monitor the sensor status while working with the application, ensuring constant supervision of the system's operation.

Image 4: Main screen, top part

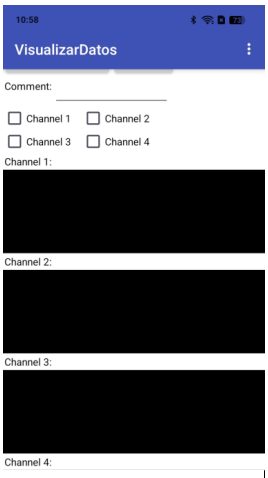


Image 5: Main screen, lower part

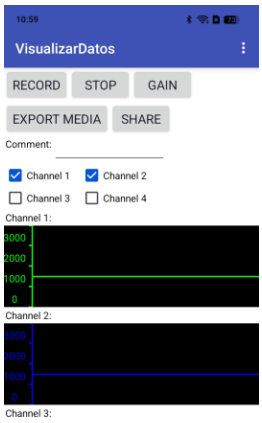


Image 6: Main screen, with 2 channels selected

Image 7: Gain options

4. Results

Once the development of the mobile application was completed, several functional tests were carried out in order to verify the correct operation of the communication between the BluEMG sensor and the mobile device through Bluetooth Low Energy.

First, the correct operation of the BLE scanning system was verified. The application was able to successfully detect nearby BLE devices and establish a connection with the BluEMG sensor. In addition, each device could be distinguished using its unique MAC address, making it possible to identify different sensors when multiple BluEMG devices were simultaneously within Bluetooth range.

After the connection was established, the application successfully received the data transmitted by the sensor and displayed the electromyographic signals from the different acquisition channels in real time. During the tests, simultaneous operation of up to four channels was verified, showing stable and smooth real-time signal visualization.

The correct operation of the different functionalities implemented in the application was also verified, including channel activation and deactivation, gain configuration, and ADC conversion control.

Furthermore, the recording and data export system was successfully tested. The application was able to store the acquired samples and generate CSV files compatible with external tools such as Excel or MATLAB for further analysis.

The following table shows a simplified example of the CSV format generated by the application:

Sample,CH1,CH2,CH3,CH4
0,1520,1488,1512,1499
1,1531,1495,1508,1501
2,1540,1502,1515,1506
3,1535,1498,1510,1503

Overall, the obtained results demonstrate that the developed application successfully fulfills the proposed objectives, allowing wireless communication with the BluEMG

sensor, real-time visualization of electromyographic signals, and exportation of the acquired data for further processing.

5. Conclusion

In conclusion, this project has enabled the development of a mobile application capable of acquiring, processing, displaying, and storing electromyographic signals from a BluEMG sensor using Bluetooth Low Energy technology. The application not only allows real-time signal visualization but also incorporates functionalities such as channel and gain configuration, data recording, and CSV file exporting and sharing.

Furthermore, the use of MIT App Inventor has proven to be a suitable tool for the development of biomedical applications, following the straightforward implementation of complex functionalities related to BLE communication and signal processing.

Finally, this work demonstrates the feasibility of developing functional and accessible mobile solutions for biomedical signal monitoring, with potential future applications in areas such as rehabilitation, research, and prosthetic control.

6. References

[1] P. F. Pasquina et al., «First-in-man demonstration of a fully implanted myoelectric sensors system to control an advanced electromechanical prosthetic hand», J. Neurosci. Methods, vol. 244, pp. 85-93, abr. 2015, doi: 10.1016/j.jneumeth.2014.07.016.

[2] «IMPLANT DEVICE»

Índice de la memoria

Índice de ilustraciones

Índice de tablas

Capítulo 1. INTRODUCCIÓN

El desarrollo de prótesis ha evolucionado considerablemente en los últimos años gracias al avance en su control mediante señales fisiológicas extraídas del propio cuerpo humano. Los progresos en la adquisición y en el procesamiento de señales biomédicas han tenido un impacto significativo en la mejora de estos dispositivos, orientados a aumentar la calidad de vida de aquellas personas que han perdido algún miembro a lo largo de su vida.

El contexto del proyecto se enmarca dentro del desarrollo de un sensor implantable diseñado para registrar la actividad eléctrica generada por los músculos. En particular, el sensor está orientado a la medición de señales electromiográficas que se producen en los colgajos musculares utilizados en determinados procedimientos quirúrgicos. Estas señales son captadas directamente en el tejido muscular mediante el sensor implantado, lo cual permite obtener información más precisa y limpia sobre la actividad en ese músculo. La principal ventaja de emplear un sensor implantable reside en la posibilidad de utilizar electrodos de aguja para la adquisición de las señales, ya que, en caso contrario, sería necesario recurrir a electrodos superficiales, los cuales presentan una mayor sensibilidad al ruido y a las interferencias externas.

Una vez registradas, las señales son transmitidas de forma inalámbrica mediante tecnología Bluetooth Low Energy (BLE) hacia dispositivos externos, visualizadas y posteriormente tratadas. Este sistema de comunicación permite establecer un enlace eficiente entre el sensor implantable y las herramientas de procesamiento de datos, facilitando el acceso a la información obtenida por el dispositivo.

El objetivo del sensor es obtener señales electromiográficas lo más limpias y fiables posibles, minimizando las interferencias y pérdidas de información que puedan aparecer al hacer uso de cualquier otra tecnología de captación de señales como pueden ser los electrodos superficiales. Estas señales pueden ser posteriormente procesadas y utilizadas como base

para sistemas de control de prótesis, contribuyendo así al desarrollo de interfaces más naturales entre el cuerpo humano y los dispositivos protésicos.

En este contexto, el presente Trabajo de Fin de Grado se centra en el desarrollo de la aplicación encargada de establecer la comunicación con el sensor implantado, permitir la visualización de las señales captadas por los cuatro canales de los cuales dispone el sensor y descargar los datos deseados en un formato adecuado para su posterior tratamiento.

1.1 MOTIVACIÓN DEL PROYECTO

En 2017, alrededor de 57,7 millones de personas en el mundo vivían sin algún miembro debido a traumatismos. Entre las causas más frecuentes de amputación se encuentran los accidentes de tráfico, las caídas, los accidentes relacionados con otros métodos de transporte y diferentes tipos de fuerzas mecánicas. A estos casos se deben añadir también aquellos en los que la pérdida de un miembro se produce como consecuencia de complicaciones derivadas de enfermedades, especialmente las relacionadas con el pie diabético. [1]

Según diversas estimaciones, en 2017 había 425 millones de personas con diabetes en el mundo, y se prevé que esta cifra continúe aumentando en los próximos años, pudiendo alcanzar valores cercanos a 629 millones de personas en 2045. En el caso de los pacientes diabéticos, la probabilidad de sufrir una amputación de miembro inferior se sitúa entre el 7,4% y el 41,3% dependiendo de distintos factores clínicos y del seguimiento médico del paciente. [1]

Además de los traumatismos y de las complicaciones asociadas a la diabetes, existen otras causas que pueden derivar en la pérdida de una extremidad, como pueden ser ciertas complicaciones derivadas de tumores malignos o determinadas malformaciones congénitas presentes desde el nacimiento. No obstante, estas causas presentan una incidencia significativamente menor y, por este motivo, no se incluyen en la estimación principal considerada en este trabajo. [1]

Si se toma el valor medio de la probabilidad de amputación en pacientes diabéticos y se considera el número total de personas que en 2017 habían perdido algún miembro debido a traumatismos, junto con la población mundial estimada de 7.645.617.954 personas, se puede estimar que al menos un 1,7 % de la población mundial vive con la ausencia de uno o más miembros.[1]

Una vez analizados estos datos, resulta más sencillo comprender la magnitud del problema y la necesidad de desarrollar tecnologías que permitan mejorar la calidad de vida de las personas afectadas. La pérdida de una extremidad supone una limitación significativa en la vida diaria de quienes la padecen, afectando no solo a su capacidad de movimiento o movilidad – especialmente en el caso de las extremidades inferiores -, sino también a su bienestar psicológico y emocional.

De hecho, diversos estudios indican que la pérdida de un miembro tiene un impacto importante en la autoestima de las personas, debido a la estrecha relación existente entre la imagen corporal y la percepción personal. Este impacto no solo afecta al propio individuo, sino también a su entorno familiar y social más cercano. [2]

Todas estas circunstancias contribuyen a una disminución significativa de la calidad de vida de las personas afectadas. Por este motivo, los avances tecnológicos orientados al desarrollo de soluciones que permitan mejorar la funcionalidad y la integración de prótesis tienen un impacto potencial muy relevante en millones de personas en todo el mundo.

En este contexto se sitúa el desarrollo del sensor electromiográfico implantable sobre el que se basa el presente Trabajo de Fin de Grado.

1.2 OBJETIVOS

El presente Trabajo de Fin de Grado se enmarca dentro de un proyecto de mayor envergadura desarrollado por el Instituto de Investigación Tecnológica ICAI junto con el hospital de Getafe, en el que participan diversos profesionales del ámbito de la ingeniería biomédica y la tecnología sanitaria. Tal y como se ha descrito previamente, dicho proyecto tiene como objetivo el desarrollo de un sensor implantable capaz de adquirir señales electromiográficas procedentes de tejidos musculares, concretamente de colgajos, y transmitirlas de forma inalámbrica mediante tecnología Bluetooth Low Energy (BLE).

Este sistema debe ser capaz de establecer comunicación con distintos dispositivos externos, como ordenadores personales y dispositivos móviles, permitiendo así la visualización, procesamiento y almacenamiento de los datos adquiridos. En este contexto, se identifica la necesidad de desarrollar una herramienta software específica que facilite la interacción entre el usuario y el sensor implantable.

El objetivo principal de este trabajo consiste en el diseño y desarrollo de una aplicación móvil capaz de conectarse de forma fiable con el sensor mediante BLE, permitiendo la recepción y visualización en tiempo real de las señales adquiridas. En concreto, la aplicación deberá ser capaz de representar de forma clara y simultánea las cuatro señales correspondientes a los cuatro canales de adquisición del dispositivo, garantizando una correcta interpretación por parte del usuario.

Adicionalmente, la aplicación deberá incorporar funcionalidades de almacenamiento y gestión de datos, permitiendo la generación de archivos en formato CSV que contengan la información recogida durante la adquisición. Estos archivos deberán poder ser exportados de forma sencilla a través de distintas plataformas de comunicación, como correo electrónico o aplicaciones de mensajería, facilitando así su posterior análisis en entornos externos.

Otro de los objetivos relevantes del proyecto es garantizar que la aplicación desarrollada sea intuitiva, eficiente y adaptable, teniendo en cuenta que se trata de un sistema orientado a un entorno biomédico, donde la usabilidad y la fiabilidad son aspectos críticos. Asimismo, se

pretende que la aplicación sea escalable, de manera que pueda adaptarse a futuras modificaciones del sensor o a la incorporación de nuevas funcionalidades.

Cabe destacar que este trabajo se desarrolla en el contexto de un proyecto en continua evolución, lo que implica que los requisitos y necesidades pueden variar a lo largo del tiempo. Por ello, uno de los objetivos implícitos es la capacidad de adaptación del desarrollo software a nuevas especificaciones, manteniendo en todo momento la coherencia con los objetivos globales del proyecto.

En definitiva, este Trabajo de Fin de Grado tiene como finalidad contribuir al desarrollo de un sistema completo de adquisición y visualización de señales electromiográficas implantables, centrando su aportación en la capa de software móvil, fundamental para la interacción entre el dispositivo y el usuario.

1.3 METODOLOGÍA DEL PROYECTO

La realización del presente proyecto se ha desarrollado siguiendo una metodología previamente definida en las etapas iniciales. No obstante, dicha metodología no ha sido rígida, sino que ha experimentado diversas adaptaciones a lo largo del proceso de desarrollo con el objetivo de ajustarse a las necesidades y dificultades que han ido surgiendo. Este enfoque flexible ha permitido optimizar el progreso del proyecto y facilitar la toma de decisiones en cada fase.

En una primera etapa, se llevó a cabo un proceso de investigación orientado a identificar los lenguajes y herramientas más adecuados para el desarrollo de una aplicación móvil. Inicialmente, y en base a los conocimientos adquiridos en asignaturas previas relacionadas con la programación web, se planteó la posibilidad de desarrollar la aplicación siguiendo una arquitectura clásica, cliente-servidor, diferenciando, entre frontend y backend. Para ello, se consideró el uso de tecnologías como Java, Javascript, así como lenguajes de marcado y estilo como HTML y CSS. Sin embargo, antes de iniciar el desarrollo, y tras evaluar esta aproximación, uno de los tutores del proyecto recomendó el uso de la herramienta MIT APP

Inventor, una plataforma específica para el desarrollo de aplicaciones móviles. Esta recomendación supuso un cambio en la estrategia inicial, optándose finalmente por esta herramienta debido a su adecuación al tipo de aplicación planteada.

La segunda fase del proyecto, de carácter formativo, consistió en el aprendizaje del funcionamiento de MIT App Inventor. Para ello, se recurrió principalmente a la documentación oficial disponible en su página web, así como a diversos tutoriales que explican tanto la estructura de la herramienta como la implementación de funcionalidades específicas. Esta etapa resultó fundamental, ya que permitió adquirir los conocimientos necesarios para abordar el desarrollo de la aplicación de manera eficiente.

Una vez superada la fase de aprendizaje, se inició el desarrollo de la aplicación propiamente dicho. En primer lugar, se diseñó la interfaz gráfica de la pantalla inicialmente destinada a la selección del dispositivo Bluetooth con el que se establecería la conexión. En esta pantalla también se incluyeron elementos informativos que permiten visualizar determinadas características del dispositivo una vez enlazado con el teléfono móvil, mejorando así la experiencia de usuario.

Posteriormente, se implementó la lógica asociada a esta primera pantalla. En esta fase se desarrollaron las funcionalidades necesarias para la detección de dispositivos Bluetooth cercanos, la obtención de la dirección MAC del sensor y la lectura de diversos parámetros relevantes mediante el uso de identificadores UID previamente proporcionados. Esta parte fue clave para garantizar una correcta comunicación entre la aplicación y el sensor.

A continuación, se abordó el desarrollo de la segunda pantalla, que constituye el núcleo principal del proyecto y en la que se ha concentrado la mayor parte del tiempo y esfuerzo invertidos. Siguiendo una metodología similar a la anterior, se comenzó con el diseño de la interfaz gráfica, definiendo la disposición de los cuatro canales de señal y los distintos controles asociados a las funcionalidades del sensor.

Finalmente, se implementó la lógica correspondiente a esta segunda pantalla, siendo esta la etapa más compleja del proyecto. En ella se desarrollaron funcionalidades avanzadas como

la adquisición de datos procedentes del conversor analógico-digital (ADC) del sensor, la representación gráfica en tiempo real de las señales obtenidas, así como la posibilidad de modificar parámetros como la ganancia del sistema. Además, se integraron otras funcionalidades complementarias que permiten una interacción completa con el dispositivo, consolidando así el correcto funcionamiento de la aplicación.

En conjunto, la metodología seguida ha permitido estructurar el desarrollo del proyecto de forma progresiva y adaptativa, facilitando la integración de conocimientos teóricos y prácticos en el ámbito del desarrollo de aplicaciones móviles.

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

El desarrollo de este proyecto se ha fundamentado principalmente en el uso de dos tecnologías clave, cuya integración resulta esencial para garantizar el funcionamiento del sistema en su conjunto.

La primera de ellas es Bluetooth Low Energy (BLE), una tecnología de comunicación inalámbrica diseñada específicamente para aplicaciones que requieren un consumo energético reducido. BLE permite la transmisión de datos de forma eficiente entre dispositivos, manteniendo un equilibrio adecuado entre consumo, alcance y velocidad de transmisión.

En el contexto de este proyecto, el uso de BLE resulta especialmente crítico, ya que el sistema se basa en un sensor implantable en un organismo vivo. En este tipo de aplicaciones, la eficiencia energética no es simplemente una ventaja, sino una necesidad fundamental. Un consumo elevado implicaría una reducción significativa de la autonomía del dispositivo debido al incremento de ciclos de carga y, consecuentemente, su vida útil se reduciría, lo que implicaría tener que operar al paciente más veces . Este hecho no solo incrementaría el riesgo para el paciente, sino que comprometería la viabilidad práctica del sistema. Por tanto, la elección de BLE como tecnología de comunicación responde a la necesidad de maximizar la duración del dispositivo implantable, minimizando al mismo tiempo el impacto sobre el usuario.

La segunda tecnología empleada, o más concretamente, herramienta de desarrollo, es MIT App Inventor, una plataforma diseñada para la creación de aplicaciones móviles de forma visual e intuitiva.

Cabe destacar que, a pesar de su gran potencial, MIT App Inventor sigue siendo una herramienta relativamente desconocida fuera de ciertos ámbitos académicos y de prototipado. Sin embargo, en el desarrollo de este proyecto ha demostrado ser especialmente

útil, permitiendo una rápida implementación de las funcionalidades requeridas y una adaptación ágil a los cambios en los requisitos del sistema.

Dado el papel central que esta herramienta desempeña en el desarrollo de la aplicación móvil asociada al sensor implantable, se considera oportuno profundizar en su funcionamiento, características y limitaciones en el siguiente apartado.

2.1 MIT APP INVENTOR

La realización del presente proyecto se ha basado íntegramente en el uso de la herramienta MIT App Inventor, desarrollada por el Massachusetts Institute of Technology. Esta plataforma de desarrollo visual permite la creación de aplicaciones móviles mediante un entorno basado en bloques, lo que simplifica notablemente la implementación de funcionalidades complejas sin necesidad de recurrir a la programación tradicional basada en código textual.

El principal objetivo de MIT App Inventor es facilitar el desarrollo de aplicaciones móviles, especialmente en contextos educativos y de prototipado rápido, permitiendo a los desarrolladores centrarse en la lógica de funcionamiento de la aplicación y en la integración de distintos módulos, como la comunicación inalámbrica o el procesamiento de datos. En el contexto de este proyecto, esta herramienta ha resultado especialmente útil para implementar de forma ágil la comunicación mediante Bluetooth Low Energy (BLE), así como para gestionar la visualización y el almacenamiento de las señales adquiridas por el sensor.

Uno de los aspectos relevantes de esta plataforma es su intención de abordar el problema de la compatibilidad entre los principales sistemas operativos móviles, como Android e iOS. Sin embargo, actualmente el soporte para dispositivos con el sistema operativo iOS se encuentra todavía en fase beta. Esto implica ciertas limitaciones en cuanto a funcionalidades disponibles, rendimiento y estabilidad en dispositivos de Apple, lo que debe tenerse en cuenta durante el desarrollo y las pruebas de la aplicación.

A pesar de estas limitaciones, MIT App Inventor constituye una herramienta muy adecuada para el desarrollo de prototipos funcionales en el ámbito académico, ya que permite una rápida iteración, facilita la depuración de errores y reduce significativamente la complejidad del proceso de desarrollo. En este proyecto, su utilización ha permitido implementar de forma eficiente las funcionalidades requeridas, adaptándose a los cambios y nuevas necesidades que han ido surgiendo a lo largo del desarrollo del sistema.

2.2 BLUETOOTH LOW ENERGY (BLE)

Bluetooth Low Energy (BLE) es una tecnología de comunicación inalámbrica diseñada para permitir el intercambio de datos entre dispositivos, consumiendo una cantidad de energía muy reducida. Esta tecnología forma parte del estándar Bluetooth y fue desarrollada específicamente para aplicaciones donde el bajo consumo energético y la transmisión periódica de pequeños volúmenes de datos resultan fundamentales, como ocurre en dispositivos biomédicos, sensores inalámbricos, wearables o sistemas IoT.

A diferencia del Bluetooth clásico, BLE no está pensado para transmitir grandes cantidades de información de manera continua, como audio o video, sino para enviar datos pequeños de forma rápida. Para ello, BLE utiliza un sistema basado en eventos, donde los dispositivos permanecen la mayor parte del tiempo en modo de bajo consumo y únicamente activan la comunicación cuando es necesario transmitir información. Gracias a esto, dispositivos alimentados por batería pueden funcionar durante largos periodos de tiempo con un consumo energético muy reducido.

La arquitectura de BLE se organiza en diferentes capas y utiliza el concepto de perfiles, servicios y características para estructurar la información intercambiada entre dispositivos. Un perfil BLE define el comportamiento general de la comunicación, mientras que los servicios agrupan funcionalidades relacionadas y las características representan los datos concretos que pueden leerse, escribirse o notificarse entre dispositivos. Esta organización modular facilita enormemente el desarrollo de aplicaciones y permite que diferentes dispositivos puedan comunicarse de forma estandarizada.[3]

En el proyecto desarrollado, BLE desempeña un papel fundamental, ya que toda la comunicación entre el sensor BluEMG y la aplicación móvil se realiza utilizando esta tecnología. La elección de BLE resulta especialmente adecuada debido a que el sistema requiere transmitir continuamente señales electromiográficas desde un dispositivo portátil alimentado por batería, manteniendo al mismo tiempo un consumo energético reducido y una comunicación inalámbrica estable.

Para la implementación de esta comunicación se ha diseñado un perfil BLE propietario específico para el sensor BluEMG y la aplicación móvil desarrollada. Este perfil incluye varios servicios encargados de gestionar diferentes funcionalidades del sistema.

El primero de ellos es el Device Information Service, unnservicio estándar de BLE utilizado para proporcionar información general sobre el dispositivo. Este servicio contiene las características *"Manufacturer Name", "Firmware Revisión"* y *"Model Number"*, inicializadas respectivamente con los valores "Comillas-ICAI". "V0.0" y "BluEMG". Gracias a este servicio, la aplicación móvil puede identificar correctamente el dispositivo conectado y mostrar información relevante al usuario.

El segundo servicio implementado es el Battery Service, también estándar dentro de BLE. Este servicio permite monitorizar el estado energético del sensor inalámbrico. Incluye la característica *"Battery Level*", que indica el porcentaje restante de batería, y la característica *"Battery Energy Status*", que contiene información sobre la potencia de carga o descarga de la batería. En concreto, se utiliza el campo *"Charge Rate",* expresado en vatios, donde valores positivos indican carga de batería y valores negativos indican descarga. Esta información resulta especialmente útil en dispositivos biomédicos portátiles, ya que permite supervisar el estado energético del sistema en tiempo real.

Además de los servicios estándar, se ha desarrollado un servicio propietario denominado ADC Service, diseñado específicamente para controlar el conversor analógico-digital del dispositivo y transmitir las señales electromiográficas capturadas. Este servicio constituye el núcleo principal de la comunicación BLE del proyecto.

Dentro de este servicio se encuentra la característica “Datos ADC”, encargada de transmitir las muestras adquiridas por el sensor. La información enviada consiste en una matriz 4x25 valores uint16, donde cada muestra contiene los datos de los cuatro canales. En total, se transmiten 200 bytes por paquete, tamaño perfectamente compatible con BLE, cuyo límite máximo para una característica es de 512 bytes. Esta característica permite lectura y notificaciones, lo que posibilita que la aplicación móvil reciba automáticamente los nuevos datos en tiempo real sin necesidad de realizar consultas continuas al dispositivo.

El servicio ADC también incorpora la característica “Conv On/Off”, utilizada para habilitar o deshabilitar las conversiones del ADC. Mediante un único byte de escritura, la aplicación puede iniciar las mediciones cuando desea adquirir señales o detenerlas cuando no son necesarias, reduciendo así el consumo energético del dispositivo.

Otra característica importante es “Ganancia”, que permite configurar la ganancia del PGA (*Programmable Gain Amplifier*) de la tarjeta electrónica. Esta configuración se realiza mediante un byte cuyos valores representan diferentes ganancias posibles, desde 1 hasta 200. Actualmente se aplica una única ganancia común a todos los canales, aunque la arquitectura diseñada permitiría implementar ganancias independientes por canal de futuras conversiones.

Además, el servicio ADC incorpora una característica adicional denominada “CanalesOnOff”, cuya función es permitir la selección dinámica de los canales que se desean muestrear. Esta característica está formada por un vector de 4 bytes, donde cada posición representa el estado de uno de los cuatro canales disponibles. Un valor igual a 1 indica que el canal correspondiente se encuentra habilitado para el muestreo, mientras que un valor igual a 0 indica que dicho canal permanece desactivado.

Por último, para identificar de forma única este servicio propietario y sus características, se han utilizado UUID de 128 bits generados específicamente para el proyecto mediante la herramienta Linux uuidgen. El uso de UUID personalizados es habitual en servicios BLE propietarios, ya que evita conflictos con servicios estándar definidos oficialmente por Bluetooth SIG.

Capítulo 3. ESTADO DE LA CUESTIÓN

El presente proyecto se enmarca dentro del ámbito de la ingeniería biomédica, un sector en constante evolución y desarrollo debido al creciente interés por mejorar la calidad de vida de los pacientes mediante el uso de tecnologías cada vez más avanzadas. En concreto, el estudio y desarrollo de dispositivos implantables constituye una de las áreas con mayor proyección dentro de este campo, ya que combina conocimiento de electrónica, telecomunicaciones, medicina y procesamiento de señales para crear soluciones capaces de monitorizar, estimular o asistir distintas funciones del organismo humano.

Debido a que estos dispositivos se integran parcialmente o totalmente en el interior del cuerpo humano, los requisitos de diseño y funcionamiento son especialmente exigentes. Aspectos como la biocompatibilidad, la fiabilidad, la seguridad, el consumo energético o la capacidad de comunicación inalámbrica adquieren una gran relevancia, puesto que cualquier mejora tecnológica puede traducirse directamente en un incremento del bienestar y la seguridad del paciente. Por este motivo, la investigación en este ámbito se encuentra en continuo crecimiento, impulsada tanto por los avances científicos como por la necesidad de desarrollar sistemas más eficientes, menos invasivos y con una mayor autonomía.

En los últimos años, la aparición de nuevas tecnologías relacionadas con sensores biomédicos, sistemas de adquisición de señales y circuitos implantables ha permitido ampliar considerablemente las posibilidades de estos dispositivos. Gracias a ello, actualmente es posible registrar y analizar diferentes tipos de señales fisiológicas en tiempo real, facilitando aplicaciones relacionadas con el diagnóstico médico, la rehabilitación, el control protésico o la monitorización continua de pacientes.

Con el objetivo de comprender mejor el contexto tecnológico y científico en el que se desarrolla este trabajo, en el presente apartado se realizará una revisión del estado actual del sector biomédico relacionado con los dispositivos implantables. Para ello, se describirán los principales tipos de sensores implantables existentes, las señales biomédicas más relevantes

que pueden adquirirse mediante estos sistemas y las tecnologías de alimentación más utilizadas, especialmente en lo referente a las baterías implantables. Todo ello permitirá obtener una visión global del funcionamiento y de los desafíos actuales asociados a este tipo de tecnologías.

3.1 SENSOR BLUEMG DEL IIT

El sensor sobre el que se fundamenta la totalidad del proyecto, y que constituye la principal motivación para el desarrollo de la aplicación móvil, es un sistema denominado BluEMG, desarrollado por el Instituto de Investigación Tecnológica de la Universidad Pontificia de Comillas. Este sistema surge de la necesidad de adquirir señales electromiográficas de alta calidad mediante el uso de electrodos de aguja, evitando así el empleo de electrodos de superficie, los cuales presentan una mayor susceptibilidad al ruido, a las interferencias externas y a la degradación de la señal obtenida.

Para hacer posible esta adquisición de señales, se ha trabajado sobre una técnica innovadora denominada VMDT, actualmente en proceso de patente. Esta técnica representa una evolución de la conocida RPMI (*Regenerative Peripheral Nerve Interface*), utilizada principalmente en el ámbito quirúrgico relacionado con amputaciones, rehabilitación avanzada y control de prótesis mioeléctricas. La técnica VMDT consiste en conectar la parte de un nervio que todavía conserva funcionalidad a un colgajo de músculo sano, permitiendo registrar la señal mioeléctrica generada en dicho músculo como consecuencia de la actividad nerviosa residual.[4]

La obtención de estas señales resulta especialmente relevante debido a su aplicación en el control de prótesis, ya que permite transformar la actividad nerviosa del paciente en señales útiles para el accionamiento y control protésico. Gracias a esta metodología, es posible obtener señales más limpias, precisas y fiables, mejorando significativamente la calidad de adquisición respecto a otros métodos convencionales.

El sistema de adquisición dispone de cuatro canales diferenciales con una ganancia fija de 100 en cada uno de ellos, además de incorporar un amplificador de ganancia programable (*Programmable Gain Amplifier,* PGA) que permite ajustar la amplificación según las necesidades de adquisición. Asimismo, el rango dinámico del sistema viene determinado por una señal de salida comprendida entre 0 y 2,5 V, con una referencia de 1,1 V, lo que permite adaptar la sensibilidad del sistema en función de la ganancia total aplicada.

3.2 SENSORES ELECTROMIOGRÁFICOS IMPLANTABLES

Durante décadas, el control de prótesis mioeléctricas se ha basado principalmente en el uso de electrodos superficiales, debido a su facilidad de aplicación y a su carácter no invasivo. Esta técnica ha permitido el desarrollo de los primeros sistemas funcionales de control protésico; sin embargo, presenta importantes limitaciones en la adquisición de señales, lo que repercute directamente en la precisión y fiabilidad del control de la prótesis.

Entre las principales limitaciones de los electrodos superficiales destacan las siguientes:

1) Susceptibilidad al ruido eléctrico generado por el entorno
2) El registro de la actividad eléctrica procedente de músculos adyacentes al electrodo (crosstalk), lo que puede provocar activaciones no deseadas en la prótesis.
3) El desplazamiento de los electrodos superficiales sobre la piel.
4) La sudoración de la piel altera la impedancia de contacto y afecta a la calidad de la señal.

[5]

Estas limitaciones hacen que las señales adquiridas sean menos selectivas, más ruidosas y, en consecuencia, menos adecuadas para sistemas de control avanzados con múltiples grados de libertad. Por ello, surge la necesidad de emplear electrodos implantables, capaces de registrar la señal mioeléctrica directamente en el músculo, obteniendo así señales de mayor calidad, estabilidad y especificidad.

No obstante, la implantación de electrodos implica, en la práctica, la introducción de un sistema completo dentro del organismo. Este hecho supone un reto tecnológico significativo, ya que la incorporación de dispositivos electrónicos en el cuerpo humano conlleva riesgos intrínsecos que deben minimizarse, como la respuesta inmunitaria, la degradación de materiales o la posible toxicidad. Por tanto, el diseño de sensores EMG implantables requiere un enfoque multidisciplinar que tenga en cuenta aspectos biomédicos, electrónicos y mecánicos.

Desde el punto de vista funcional, un sensor EMG implantable puede dividirse en tres bloques principales. En primer lugar, los electrodos, encargados de captar la señal bioeléctrica. En segundo lugar, el sistema electrónico, que incluye elementos como amplificadores de señal, convertidores analógico-digital (ADC), módulos de ganancia y sistemas de comunicación inalámbrica, como Bluetooth, entre otros. Por último, el sistema de alimentación, basado en baterías y cargadores inalámbricos, que proporciona la energía necesaria para el funcionamiento del dispositivo.

Existen distintos tipos de electrodos implantables en función de la aplicación y de las características de la señal que se desea obtener. Entre ellos destacan los siguientes:

1) Electrodo de lazo de alambre, formado por un hilo conductor con un extremo expuesto que permite el contacto eléctrico con el tejido.
2) Electrodo cortical, empleado en estudios de electrocorticografía (EcoG), no siendo relevante para aplicaciones de EMG en prótesis.
3) Electrodo multielemento, compuesto por un conjunto de finos hilos aislados agrupados, que permite el registro simultáneo de múltiples señales musculares.

De estos, los electrodos multielemento resultan especialmente adecuados para aplicaciones en control de prótesis avanzadas, ya que permiten obtener información de diferentes regiones musculares de forma simultánea, facilitando así un control más preciso y con mayor número de grados de libertad. [6]

Más allá de los electrodos, el resto de los módulos electrónicos del sensor debe adaptarse completamente al entorno biológico. Para garantizar la comodidad del paciente y el correcto funcionamiento del dispositivo, estos sistemas deben ser de tamaño reducido o presentar características mecánicas que les permitan integrarse de forma natural en el tejido. En este sentido, existen diversas estrategias tecnológicas.

El siguiente elemento que debe estar adaptado totalmente a las condiciones del interior del cuerpo es el resto de los módulos que contiene el sensor. Además, debe tratarse de un elemento muy pequeño o muy moldeable para poder asegurarnos de la comodidad del paciente. Hay muchas formas de hacerlo y muchas otras que todavía siguen en desarrollo, pero nos vamos a centrar en las 3 principales técnicas:

1) Encapsulado biocompatible: Consiste en alojar la electrónica en una carcasa fabricada con materiales biocompatibles y herméticos, capaces de impedir la entrada de fluidos y evitar la degradación del dispositivo. Además, el diseño geométrico del encapsulado debe minimizar el daño tisular y prevenir posibles lesiones.
2) Miniaturización extrema: Implica la integración de todos los componentes electrónicos en dispositivos de tamaño muy reducido, mediante tecnologías avanzadas como circuitos integrados específicos (ASIC) o sistemas en chip (SoC). Esta técnica permite disminuir el impacto físico del implante y mejorar la tolerancia por parte del organismo.
3) Electrónica flexible: Basada en el uso de materiales y estructuras flexibles que permiten al dispositivo adaptarse a la geometría y dinámica de los tejidos biológicos, reduciendo el estrés mecánico y mejorando la estabilidad a largo plazo.

Finalmente, uno de los aspectos más críticos en el diseño de sensores EMG implantables es el sistema de alimentación. El uso de baterías biocompatibles es fundamental para el suministro seguro y fiable de energía dentro del cuerpo humano. Sin embargo, las químicas convencionales y las estructuras rígidas hacen muy difícil la adaptación al entorno biológico. En este contexto, las baterías biocompatibles son clave, ya que deben integrarse en tejidos

blandos y dinámicos sin comprometer ni la funcionalidad de los dispositivos ni la seguridad del paciente.[7]

A diferencia de las baterías convencionales, los sistemas implantables requieren características específicas, entre las que destacan la biocompatibilidad, una adecuada densidad energética, estabilidad de voltaje y compatibilidad mecánica con los tejidos. Estas exigencias hacen necesario el uso de materiales avanzados, tanto orgánicos como inorgánicos, así como el desarrollo de electrodos nanoestructurados y arquitecturas flexibles o incluso biodegradables, capaces de adaptarse al entorno fisiológico. [7]

En la actualidad, dispositivos implantables como los marcapasos, cuyas necesidades de biocompatibilidad son similares a las requeridas por el sensor electromiográfico desarrollado en este proyecto, emplean baterías convencionales de litio de larga duración, con una vida aproximada de entre cinco y diez años. Sin embargo, el principal inconveniente de este tipo de baterías reside en su agotamiento energético. Cuando la batería alcanza el final de su vida útil, resulta necesario someter al paciente a una nueva intervención quirúrgica para proceder a su reemplazo.[8]

Teniendo en cuenta los riesgos clínicos, el coste asociado y el impacto que supone para el paciente la realización de cirugías adicionales, existe una creciente necesidad de desarrollar nuevas soluciones energéticas que permitan evitar o reducir este tipo de intervenciones. Entre las líneas de investigación actuales destacan las técnicas de carga inalámbrica para dispositivos implantables y, de forma aún más innovadora, el desarrollo de sistemas capaces de obtener energía directamente del propio organismo.

En este contexto, diversos grupos de investigación internacionales están trabajando en alternativas prometedoras. En China, por ejemplo, se está estudiando el desarrollo de baterías capaces de utilizar el oxígeno presente en el cuerpo humano como fuente de energía, con el objetivo de proporcionar un funcionamiento potencialmente continuo durante toda la vida del paciente. Aunque los resultados obtenidos hasta el momento son esperanzadores, esta tecnología todavía se encuentra en fases preliminares de investigación y no dispone aún de una base suficientemente sólida para su aplicación clínica.

Por otro lado, investigaciones desarrolladas en Suiza han explorado el uso de células solares implantables bajo la piel para alimentar dispositivos médicos implantados. A pesar del gran número de estudios y avances que se están llevando a cabo de forma simultánea en este ámbito, en la actualidad la mayoría de los dispositivos implantables comerciales, como los marcapasos, continúan dependiendo de baterías de litio convencionales que requieren reemplazo quirúrgico una vez agotadas.[9]

En los sensores EMG implantables, la miniaturización es un aspecto crítico. Por ello, se emplean técnicas de fabricación avanzadas como la deposición en fase de vapor, la fotolitografía o el estampado elastomérico, que permiten integrar baterías de pequeño tamaño directamente en el sistema electrónico del sensor. Además, el diseño de las interfaces entre la batería y el tejido biológico resulta fundamental, ya que problemas como las corrientes de fuga, la respuesta inflamatoria o el desajuste mecánico pueden comprometer tanto la medición de la señal EMG como la estabilidad del implante. [7]

3.3 TIPOS DE SEÑALES ELÉCTRICAS EN EL CUERPO HUMANO

En el cuerpo humano se generan diferentes tipos de señales eléctricas que desempeñan un papel fundamental en el funcionamiento de los distintos sistemas del organismo. Estas señales permiten la comunicación entre células y tejidos, coordinando múltiples procesos fisiológicos necesarios para el correcto funcionamiento del cuerpo.

Entre las señales eléctricas más relevantes del organismo se encuentran las señales electrocardiográficas (ECG), las señales electromiográficas (EMG) y las señales electroencefalográficas (EEG). Las señales electrocardiográficas están asociadas a la actividad eléctrica del corazón, que permite la contracción del músculo cardíaco y, por tanto, el bombeo de la sangre a través del sistema circulatorio. Por su parte, las señales electromiográficas reflejan la actividad eléctrica generada por los músculos durante su contracción, proporcionando información sobre el funcionamiento de las fibras musculares y su activación.

Las señales electroencefalográficas, en cambio, están relacionadas con la actividad eléctrica del cerebro y se originan a partir de la comunicación entre las neuronas. Estas señales permiten estudiar diferentes procesos cerebrales, como la actividad neuronal asociada al pensamiento, la percepción o el control de movimiento.

Para analizar el movimiento muscular podría recurrirse tanto al estudio de las señales electroencefalográficas como al de las señales electromiográficas. Sin embargo, debido a la mayor complejidad que presenta la adquisición y el procesamiento de señales EEG, en este trabajo se opta por utilizar señales electromiográficas, ya que estas se generan directamente en los músculos responsables del movimiento y resultan más accesibles para su medición.

Las señales electromiográficas presentan amplitudes relativamente bajas. Cuando son captadas mediante electrodos de superficie, su rango típico de voltaje se sitúa aproximadamente entre 0 y 5 mV. Sin embargo, cuando estas señales se registran mediante sensores intramusculares o dispositivos implantables, el rango puede ampliarse aproximadamente entre 1 mV y 10 mV, debido a la mayor proximidad del sensor a las fibras musculares activas y a la reducción de interferencias externas.

Las señales electromiográficas (EMG) se obtienen a partir del registro de la actividad eléctrica generada por las unidades motoras durante la contracción muscular. En particular, estas señales están compuestas por la suma de los potenciales de acción de unidad motora (PAUMs), que representan la respuesta eléctrica producida por la activación simultánea de las fibras musculares pertenecientes a una misma unidad motora.

La adquisición de este tipo de señales puede realizarse mediante diferentes técnicas, siendo una de las más utilizadas la electromiografía intramuscular, que emplea electrodos de aguja insertados directamente en el tejido muscular. Este método permite registrar con mayor precisión la actividad eléctrica local, ya que capta señales procedentes de un número reducido de fibras musculares, mejorando así la reducción espacial y la calidad de la señal obtenida.

El potencial de acción de unidad motora (PAUM) se define como el registro eléctrico resultante de las descargas de las fibras musculares (FM) que componen una unidad motora (UM). Desde el punto de vista fisiológico, una unidad motora está formada por una motoneurona y el conjunto de fibras musculares que esta inerva, actuando como la unidad funcional básica del sistema neuromuscular.

En cuanto a sus características, los PAUMs presentan una amplitud media del orden de 0,5mV y una duración comprendida generalmente entre 8 y 14 ms. No obstante, tanto la forma como la amplitud y la duración de estos potenciales pueden variar en función de diversos factores, entre los que destacan las propiedades estructurales de la unidad motora (como el número y la distribución de fibras musculares) y sus características funcionales (como el tipo de fibra o el patrón de activación).

De este modo, el análisis de los PAUMs proporciona información relevante sobre el estado y el funcionamiento del sistema neuromuscular, siendo una herramienta fundamental tanto en el ámbito clínico como en aplicaciones de ingeniería biomédica, como el desarrollo de sistemas de control basados en señales electromiográficas.

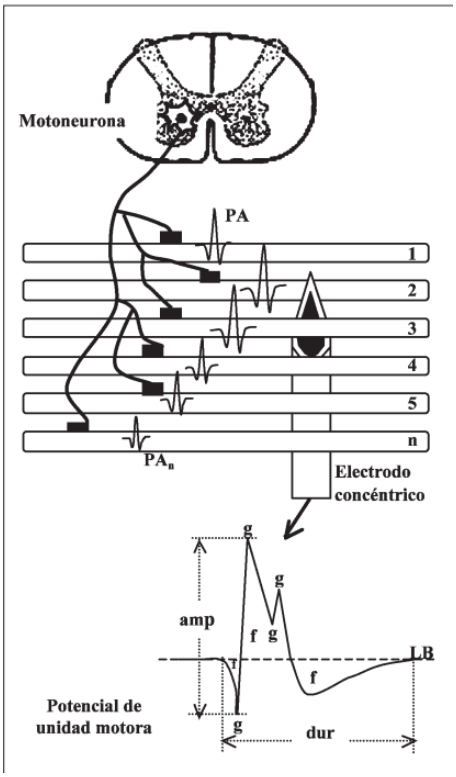


Ilustración 1:Potencial de acción de Unidad Motora [10]

Las señales electromiográficas presentan amplitudes relativamente bajas, lo que hace necesario el uso de sistemas de amplificación adecuados para poder procesarlas correctamente. En función del tipo de adquisición y del sistema empleado, estas señales suelen amplificarse en un rango que puede variar aproximadamente entre 5 y 250.000 veces, con el objetivo de llevarlas a niveles de tensión compatibles con los sistemas electrónicos de adquisición y procesamiento, típicamente del orden de 1 V a 10 V. [10]

No obstante, la amplificación de la señal no solo incrementa la componente útil, sino que también amplifica el ruido presente en la señal, lo que supone uno de los principales desafíos en el tratamiento de señales electromiográficas. Este ruido puede tener un origen tanto biológico como técnico. Entre las fuentes de ruido biológico se incluyen la actividad de otros músculos cercanos (crosstalk) o variaciones fisiológicas, mientras que el ruido técnico puede deberse a interferencias electromiográficas externas, al propio sistema de adquisición o al movimiento de los electrodos. [10]

Debido a la baja amplitud de las señales EMG, es frecuente que el ruido presente en la señal sea comparable o incluso superior a la señal de interés en determinados casos, lo que dificulta su análisis y procesamiento. Por este motivo, resulta imprescindible aplicar técnicas de filtrado y acondicionamiento de señal, que permitan mejorar la relación señal-ruido y extraer información relevante a partir de los datos adquiridos. [10]

En consecuencia, el diseño de sistemas de adquisición de señales electromiográficas debe prestar especial atención tanto a la etapa de amplificación como al tratamiento posterior de la señal, con el fin de garantizar la calidad de la información utilizada en aplicaciones biomédicas, como el control de prótesis. [10]

3.4 TIPOS DE PRÓTESIS

Los primeros escritos sobre prótesis para extremidades superiores fueron en Plinio el Viejo en el año 77 d.C. En esos escritos se hablaba de una prótesis de hierro que se hizo el general Sergio para poder continuar con su carrera militar. Aunque no fue hasta el siglo XVI cuando cirujanos y científicos describieron con éxito una prótesis de extremidad superior capaz de restaurar parcialmente la función de la mano. Tras la mutilación de un caballero alemán llamado Gottfried "Götz", se fabricó una novedosa mano de hierro con dedos que podían moverse pasivamente en las articulaciones metacarpofalángicas, interfalángicas proximales e interfalángicas distales.[11]

Sin embargo, no resultó ser demasiado útil debido a su peso, volumen y función limitada de restaurar la movilidad del miembro. Las prótesis seguían siendo más asistenciales que restauradoras.

Después de siglos de avances estancados en las prótesis de miembro superior, la evolución de las prótesis cambió drásticamente tras el final de la Segunda Guerra Mundial. Durante las décadas después de la fundación de la American Society for Surgery of the Hand y el National Research Council, se llegó al auge de las prótesis modernas de mano parcial.

En este contexto, la inversión en el desarrollo de prótesis fue aumentando y gracias a la mejora en las tecnologías de cada época, se fueron desarrollando distintos tipos de prótesis con objetivos únicos. [11]

En la siguiente tabla se muestra un resumen de los distintos tipos de prótesis que existen hoy en día:

Tipo	**Funcionales pasivas**	**Accionadas por el cuerpo**	**Accionadas externamente**	**Específica para la actividad**
Descripción	Prótesis estáticas que se utilizan para restaurar la forma y la función de la mano. Existen dos clases de dispositivos funcionales pasivos: - **Prótesis de silicona**: Fabricadas con silicona, que restauran la longitud y la forma de la mano. Las prótesis de silicona de alta definición pueden reproducir la mano no afectada con gran precisión - **Articuladas pasivas**: Restauran la longitud de la mano y utilizan un mecanismo de trinquete para soportar altas cargas de fuerza durante la flexión. Deben liberarse manualmente, lo que generalmente se realiza con la mano contralateral.	Restauran el movimiento activo de la mano aprovechando las articulaciones restantes del amputado. No dependen de fuentes de energía externas. Estos dispositivos también restauran la forma de la mano, aunque tienen una apariencia más mecánica y se asocian con un aumento moderado del volumen.	Restauran el movimiento de la mano mediante fuentes de energía externas (por ejemplo, baterías). - **Prótesis mioeléctricas**: Estos dispositivos utilizan señales de electromiografía procedentes de la contracción muscular para generar movimiento. - **Híbridas (accionadas por el cuerpo/mioeléctricas)**: Combinan tecnología mioeléctrica y accionada por el cuerpo para crear una prótesis con múltiples capacidades.	Dispositivos asistivos no restaurativos diseñados para ayudar a los amputados a realizar tareas específicas (por ejemplo, gancho, adaptadores para utensilios de cocina, etc.).

Tabla 1: Tipos de prótesis [11]

A continuación, se muestra una imagen de cada uno de los tipos de prótesis mencionados anteriormente en el contexto de prótesis para manos.

Funcionales pasivas:

Ilustración 2: Persona con amputación parcial de mano con prótesis de silicona de alta definición en el dedo corazón [11]

Ilustración 3:Agricultor que sufrió una amputación parcial de la mano a nivel de las articulaciones metacarpofalángicas de los dedos 2–5. Este dispositivo pasivo articulado le permitió retomar sus responsabilidades laborales [11]

Accionadas por el cuerpo:

Ilustración 4:Imagen que muestra una prótesis accionada por el propio cuerpo. [11]

Accionadas externamente:

Ilustración 5:Correspondiente a un paciente con una prótesis mioeléctrica, un tipo de prótesis accionada externamente que utiliza electromiografía para restaurar el movimiento activo de la mano. [11]

Específicas para alguna actividad:

Ilustración 6:Hombre de 40 años que sufrió una amputación parcial traumática de la mano a nivel transmetacarpiano proximal. Con el objetivo de que pudiera volver a practicar motocross, se diseñó una prótesis específica para esta actividad. [11]

Este proyecto se enfoca en el uso de las señales electromiográficas para aplicar al control de prótesis del tipo *"accionadas externamente"* de entre las presentadas anteriormente.

Capítulo 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

Tras analizar el estado actual de las tecnologías biomédicas implantables y los distintos avances desarrollados en este ámbito, resulta necesario realizar una valoración crítica de las soluciones existentes con el fin de identificar sus limitaciones, necesidades actuales y posibles líneas de mejora. Aunque en los últimos años se han producido importantes avances en sensores biomédicos, sistemas e adquisición de señales y comunicaciones inalámbricas, todavía existen numerosos desafíos relacionados con la accesibilidad, la eficiencia energética, la facilidad de uso y la integración de estos dispositivos con plataformas móviles capaces de procesar y representar la información obtenida de forma sencilla e intuitiva.

En este contexto, las tecnologías orientadas al ámbito *e-health* han adquirido una gran relevancia debido al impacto que pueden generar tanto en la calidad de vida de los pacientes como en la mejora de los procesos de monitorización y diagnóstico médico. La posibilidad de registrar señales biomédicas en tiempo real y transmitirlas de manera inalámbrica hacia dispositivos móviles abre la puerta al desarrollo de sistemas más portátiles, accesibles y adaptables a diferentes aplicaciones médicas y de investigación.

Sin embargo, pese a la existencia de múltiples soluciones comerciales y proyectos de investigación relacionados con sensores biomédicos y sistemas de monitorización, muchas de estas alternativas presentan limitaciones importantes. En algunos casos, los sistemas disponibles son cerrados y poco flexibles para el desarrollo de nuevas funcionalidades; en otros, el acceso y procesamiento de las señales biomédicas resulta complejo o requiere equipamiento especializado. Además, no todas las soluciones existentes permiten una integración sencilla entre dispositivos biomédicos inalámbricos y aplicaciones móviles capaces de visualizar, procesar y almacenar los datos adquiridos de forma eficiente.

A partir de esta necesidad surge la motivación técnica del presente proyecto, cuyo objetivo es desarrollar una aplicación móvil capaz de comunicarse con un sensor biomédico mediante Bluetooth Low Energy, adquirir señales electromiográficas en tiempo real, representarlas gráficamente y permitir tanto su almacenamiento como su posterior exportación para análisis externos. De esta forma, se busca proporcionar una solución funcional, accesible y fácilmente adaptable a futuras ampliaciones, contribuyendo al desarrollo de herramientas biomédicas móviles orientadas tanto a entornos de investigación como a posibles aplicaciones clínicas o de rehabilitación.

4.1.1 LÓGICA DE LA PANTALLA DE CONEXIÓN BLE

A continuación, se procede a describir de manera detallada el desarrollo e implementación de la primera de las pantallas de la aplicación. Esta pantalla constituye el punto de partida de la interacción con el sistema, ya que su funcionalidad principal es permitir la detección y conexión con un dispositivo Bluetooth Low Energy (BLE).

Asimismo, esta interfaz no solo se encarga de gestionar el proceso de conexión, sino que también permite la visualización de la información básica del dispositivo una vez establecida dicha conexión. Entre estos datos se incluyen parámetros relevantes como la identificación del dispositivo, su fabricante, modelo o versión de firmware, los cuales resultan esenciales para verificar el correcto funcionamiento del sistema y garantizar la adecuada comunicación entre la aplicación móvil y el sensor, además de que permiten comprobar que la aplicación se haya conectado al dispositivo correcto.

De este modo, esta primera pantalla desempeña un papel fundamental dentro del proyecto, al actuar como enlace inicial entre el usuario y el dispositivo, y asegurando una conexión fiable y proporcionando la información necesaria para el posterior tratamiento y análisis de los datos.

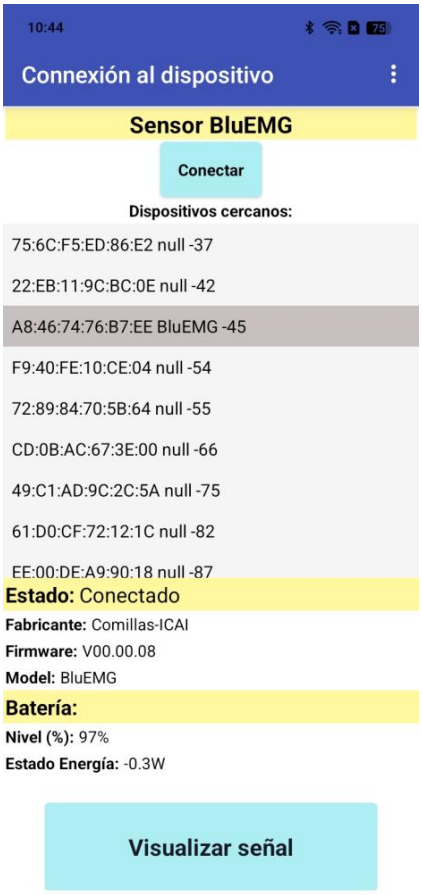


Ilustración 7: 1º pantalla, conexión con sensor

Ilustración 8: Variables globales contenedoras de UUIDs

En la ilustración 8 se definen una serie de variables globales, cada una de ellas asociada a una característica específica del sensor. Estas variables almacenan los identificadores únicos

universales (UUID) correspondientes a cada una de las funcionalidades implementadas en el dispositivo desarrollado por el IIT.

El uso de variables globales para almacenar dichos UUID resulta fundamental, ya que permite centralizar y organizar el acceso a las distintas características del sensor. Facilitando así su utilización a lo largo de toda la aplicación. De este modo, se simplifica la lectura y mantenimiento del código, evitando la repetición de valores y reduciendo la probabilidad de errores.

Además, estos UUID actúan como identificadores clave dentro del protocolo Bluetooth Low Energy, permitiendo a la aplicación móvil acceder de forma precisa a los diferentes servicios y características del dispositivo, tales como la lectura de datos, la configuración de parámetros o la obtención de información del sistema.

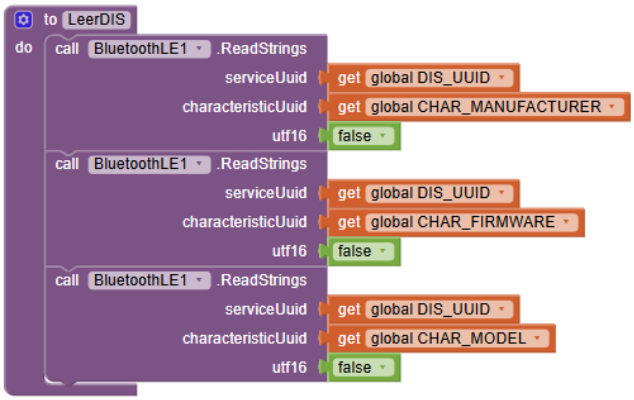


Ilustración 9: Procedimiento LeerDIS

El procedimiento denominado *leerDIS* es el encargado de gestionar la lectura de información del dispositivo a través del servicio estándar *Device Information Service* (DIS) de *Bluetooth Low Energy*. Este servicio está específicamente diseñado para proporcionar datos descriptivos del dispositivo, lo que permite identificar sus características básicas y verificar su correcto funcionamiento.

Dentro de este bloque se realizan diversas llamadas de lectura orientadas a obtener información clave del sensor. En concreto, se accede a 3 características fundamentales del dispositivo:

1) Fabricante del dispositivo, que permite identificar la entidad responsable de su desarrollo.
2) Versión del firmware, útil para conocer la versión del software interno del dispositivo y asegurar su compatibilidad con la aplicación.
3) Modelo del dispositivo, que facilita la identificación del tipo concreto de hardware utilizado.

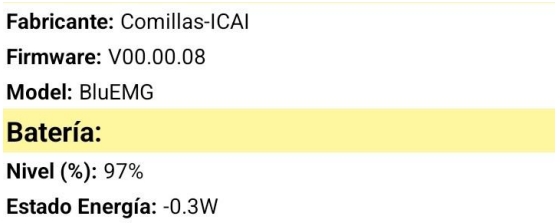


Ilustración 10: Características del sensor BluEMG

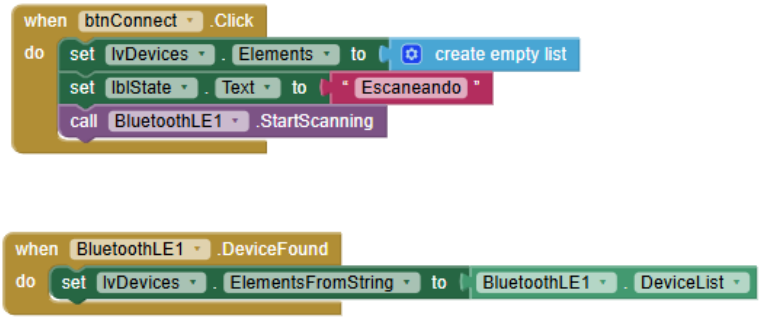


Ilustración 11: Lógica asociada al botón "Conectar" y a dispositivo encontrado por parte de BLE

En los dos bloques mostrados anteriormente se implementa la lógica asociada, por un lado, a la interacción del usuario con el botón *"Conectar"*, y por otro, al comportamiento de la aplicación durante el proceso de detección de dispositivos *Bluetooth Low Energy* (BLE).

En el primer bloque, correspondiente al evento de pulsación del botón, se define la secuencia de acciones que se ejecutan cuando el usuario inicia el proceso de conexión. En primer lugar, se limpia la lista de dispositivos disponibles, reiniciándola para evitar la presencia de datos residuales de búsquedas anteriores. A continuación, se actualiza el texto de la etiqueta de estado mostrando el mensaje *"Escaneando"*. Con el objetivo de proporcionar información visual al usuario sobre el estado actual del sistema. Finalmente, se inicia el proceso de escaneo BLE mediante la llamada correspondiente, lo que permite a la aplicación comenzar la búsqueda de dispositivos cercanos.

El segundo bloque se activa automáticamente cada vez que se detecta un nuevo dispositivo BLE durante el escaneo. En este caso, la aplicación obtiene la lista actualizada de dispositivos encontrados y la asigna al componente *ListView*, encargado de mostrar dicha información en la interfaz. De este modo, el usuario puede visualizar en tiempo real los dispositivos disponibles en su entorno y seleccionar aquel con el que desea establecer la conexión.

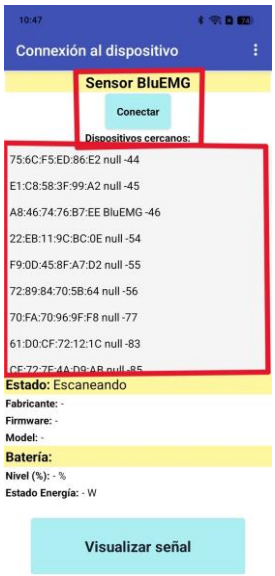


Ilustración 12: 1º pantalla, botón Conectar y ListViewer

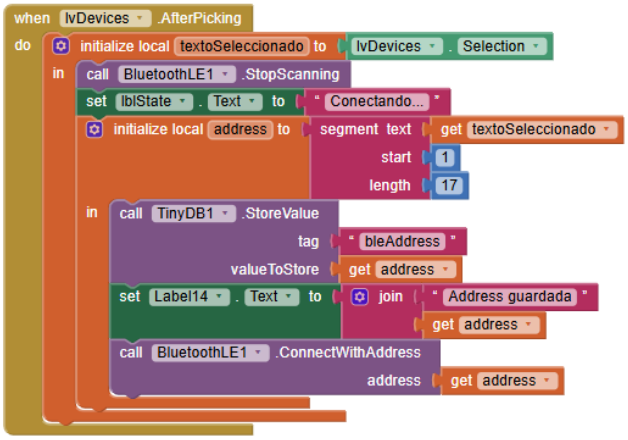


Ilustración 13: Lógica asociada a la selección del dispositivo deseado

Una vez que el usuario selecciona, dentro de la lista mostrada, el dispositivo al que desea conectarse, se ejecuta la lógica correspondiente al bloque previamente descrito. En primer lugar, se almacena en una variable local el texto asociado al dispositivo seleccionado, lo que permite trabajar posteriormente con dicha información de forma estructurada.

A continuación, se detiene el proceso de escaneo Bluetooth, evitando así la detección de nuevos dispositivos mientras se establece la conexión. Seguidamente, se actualiza el estado de la aplicación mediante el cambio del texto informativo a *"Conectando…"*, proporcionando al usuario una indicación clara del proceso en curso.

Posteriormente, a partir del texto previamente almacenado, se extrae la porción correspondiente a la dirección MAC del dispositivo, la cual se guarda en una nueva variable local. Este paso resulta fundamental, ya que la dirección MAC actúa como identificador único del dispositivo dentro de la comunicación Bluetooth.

Una vez obtenida dicha dirección y con el fin de permitir el intercambio de información entre ambas pantallas, se utiliza una base de datos local denominada *TinyDB*. En ella se almacenan los datos obtenidos en esta primera pantalla, de forma que posteriormente puedan recuperarse desde la pantalla principal y mostrarse correctamente al usuario.

Finalmente, se inicia la conexión entre el dispositivo móvil y el sensor utilizando la dirección MAC almacenada, estableciendo así el enlace necesario para la posterior comunicación de datos.

Además, la dirección MAC del dispositivo conectado se muestra por pantalla justo debajo del resto de características del sensor. Esta funcionalidad resulta especialmente útil para diferenciar e identificar correctamente distintos sensores BluEMG cuando existen varios dispositivos disponibles simultáneamente.

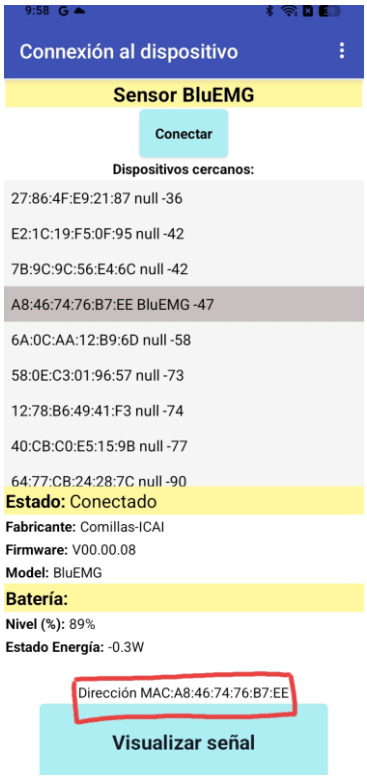


Ilustración 14: 1º pantalla, dirección MAC dispositivo conectado

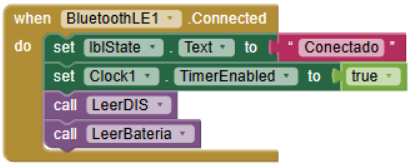


Ilustración 15: Lógica asociada a la conexión BLE, 1º pantalla

El bloque anterior se ejecuta en el momento en que la conexión entre el dispositivo móvil y el sensor Bluetooth Low Energy ha sido establecida de forma satisfactoria. Este evento marca el inicio de la fase de comunicación activa entre ambos dispositivos.

En primer lugar, se actualiza el estado de la interfaz de usuario mediante la modificación del texto informativo a *"Conectado"*, con el objetivo de indicar de forma clara que el proceso de enlace se ha completado correctamente. A continuación, se activa un reloj interno de la aplicación, el cual permite la ejecución periódica de tareas, como la lectura continua de datos o la actualización de la información mostrada en pantalla.

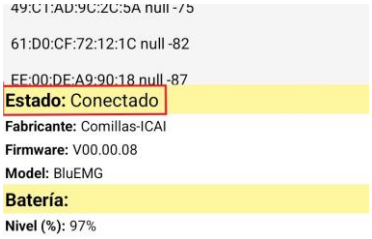


Ilustración 16: 1º pantalla, estado conectado

Finalmente, se realizan llamadas a dos procedimientos previamente definidos: leerDIS, encargado de obtener la información general del dispositivo a través del Device Information Service, tal y como se explicó anteriormente, y leerBatería, que permite acceder a los datos relacionados con el estado de la batería del sensor. De este modo, se inicia la recopilación de información relevante que será utilizada en las siguientes fases de la aplicación.

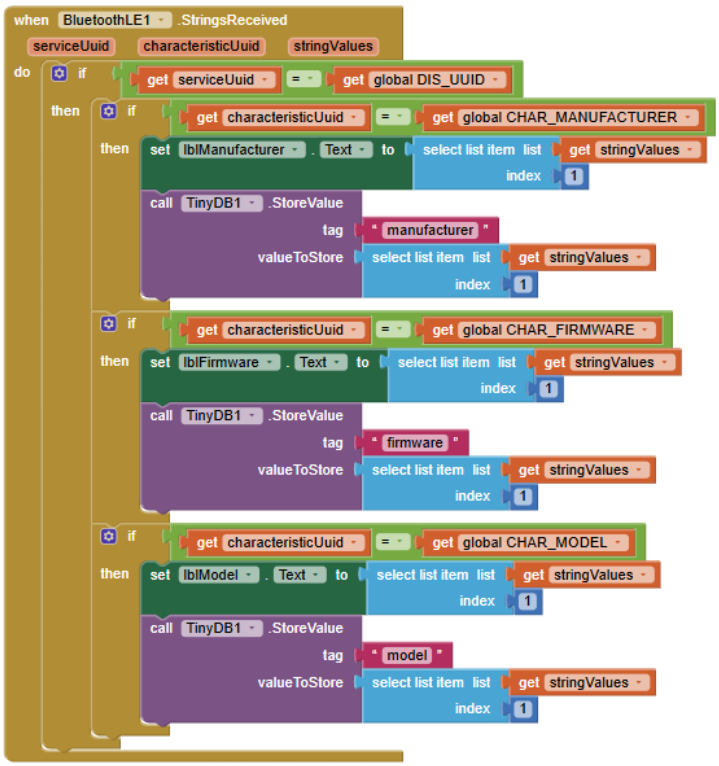


Ilustración 17: Lógica asociada a la recepción de características de BluEMG

En este bloque se lleva a cabo la recepción y el procesamiento de los datos que previamente han sido solicitados al dispositivo mediante las operaciones de lectura. Dicho bloque se ejecuta de forma automática cuando el dispositivo Bluetooth Low Energy (BLE) envía una respuesta, constituyendo así un elemento fundamental dentro del flujo de comunicación entre la aplicación y el sensor.

Tal y como se observa en la cabecera del bloque, se reciben 3 parámetros principales: *serviceUuid*, que identifica el servicio al que pertenece la información recibida; *characteristicsUuid*, que especifica la característica concreta dentro de dicho servicio; y *stringValues*, que contiene los datos transmitidos en formato de lista de cadenas de texto.

En primer lugar, se realiza un filtrado en función del servicio, procesándose únicamente aquellos datos que pertenecen al *Device Information Service*. Este paso resulta esencial para

evitar interferencias con información procedente de otros servicios y garantizar que únicamente se traten los datos relevantes en este contexto.

A continuación, se lleva a cabo un segundo filtrado basado en el tipo de característica recibida. En función de esta, se identifica si el dato corresponde al fabricante, a la versión del firmware o al modelo del dispositivo. Una vez determinado el tipo de información, el valor se muestra en la interfaz de usuario mediante las etiquetas correspondientes y, adicionalmente, se almacena en la memoria interna del dispositivo móvil. Esta persistencia permite reutilizar dichos datos en otras partes de la aplicación, como en la segunda pantalla, sin necesidad de realizar nuevas lecturas.

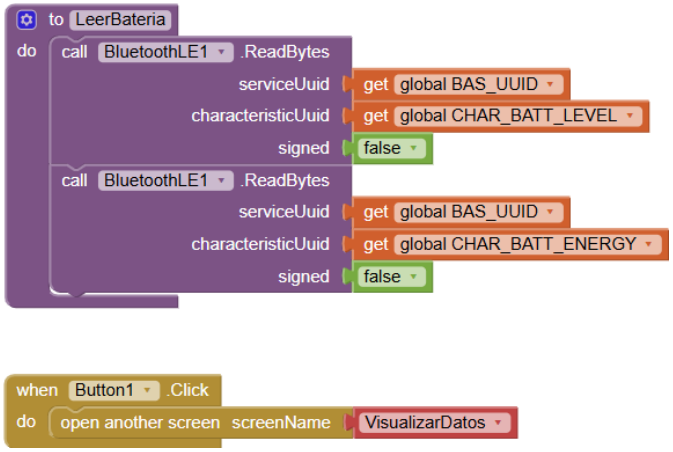


Ilustración 18: Procedimiento LeerBatería y lógica asociada al botón "Visualizar señal"

En la figura mostrada se presenta el procedimiento denominado leerBatería, cuya finalidad es obtener información relativa al estado energético del dispositivo mediante el uso del servicio estándar *Battery Service* de BLE. A través de este procedimiento se realiza, en primer lugar, la lectura del nivel de batería, expresado generalmente en porcentaje, lo que permite conocer el estado de carga del dispositivo en tiempo real.

Adicionalmente, se lleva a cabo la lectura de un segundo parámetro relacionado con la potencia instantánea de la batería, es decir, la energía con la que esta se está cargando o descargando en cada momento, expresada en vatios. Este dato resulta especialmente relevante para analizar el comportamiento energético del sistema y evaluar su consumo durante el funcionamiento.

Por otro lado, el segundo bloque representado en la imagen corresponde a la lógica asociada a un elemento de interacción de la interfaz, concretamente el botón *"Visualizar señal"*. Cuando el usuario pulsa dicho botón, se ejecuta la acción de abrir una nueva pantalla dentro de la aplicación. Esta segunda pantalla está destinada a la visualización de los datos adquiridos, permitiendo al usuario interactuar con la información recogida por el sensor de una manera más detallada.

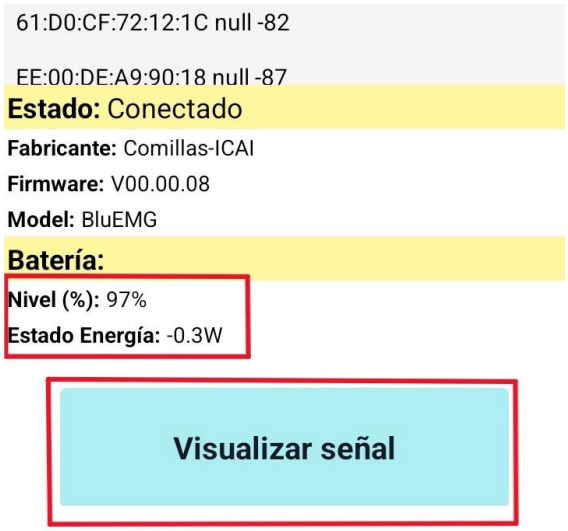


Ilustración 19: 1º pantalla, datos batería y boton Visualizar Señal

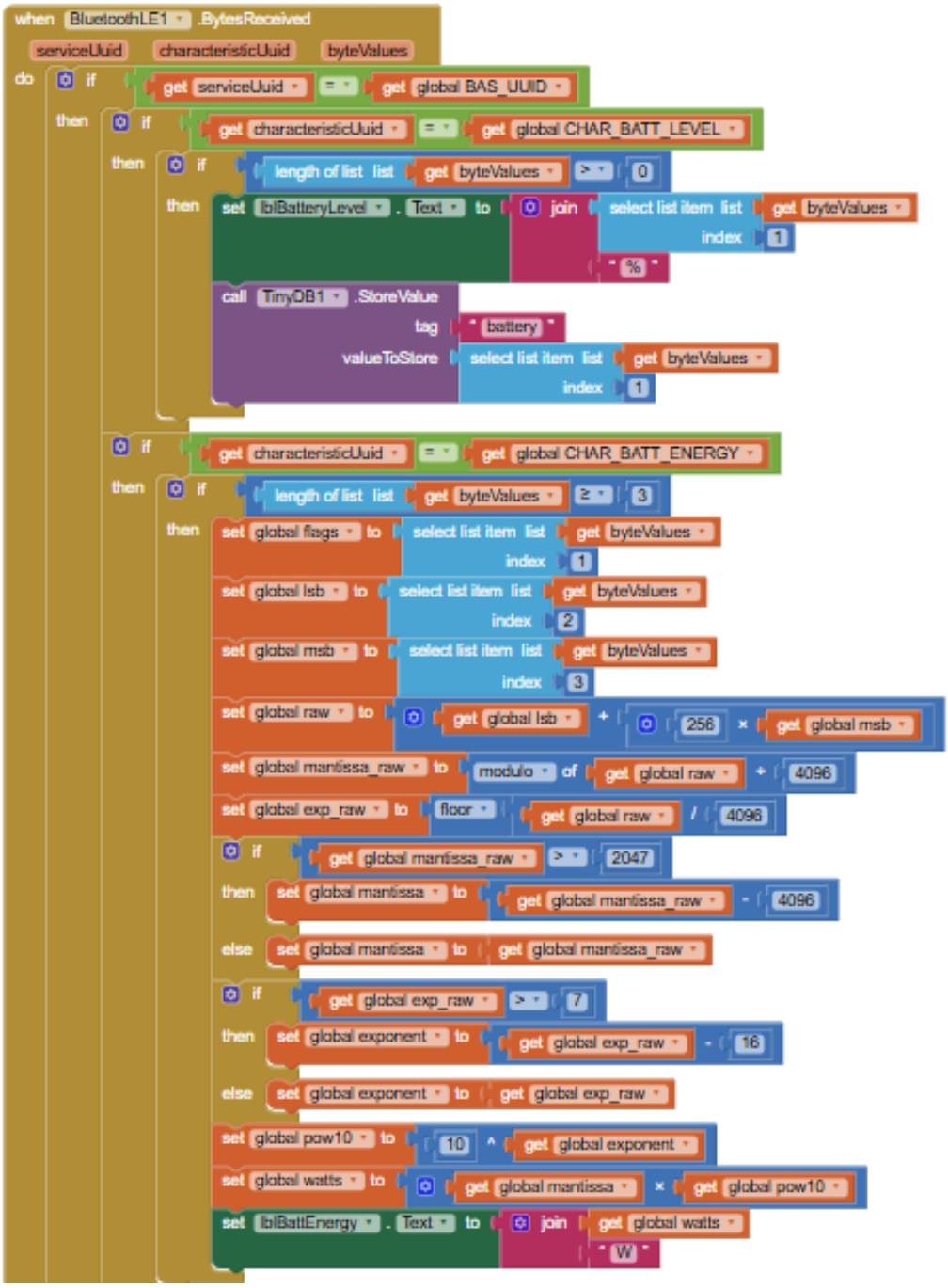


Ilustración 20: Lógica asociada a la recepción de bytes y estado de batería, 1º pantalla

Este bloque se ejecuta automáticamente cuando se reciben los datos pedidos al llamar a LeerBatería. Y, es donde se lleva a cabo el procesamiento de los datos en formato binario

(bytes) obtenidos del dispositivo, concretamente aquellos relacionados con el estado de la batería. Dado que estos datos no son directamente interpretables, es necesario transformarlos en un formato comprensible antes de poder mostrarlos en la interfaz de usuario.

En primer lugar, se realiza un filtrado en función de la característica correspondiente al nivel de batería. Una vez identificada, se extrae el valor recibido, que representa el porcentaje de carga, y se formatea adecuadamente añadiendo el símbolo "%". Para su correcta visualización. Asimismo, este valor se almacena en la memoria local del dispositivo.

A continuación, se procesa la característica asociada a la potencia de carga o descarga de la batería, expresada en vatios. A diferencia del nivel de batería, este valor no se recibe de forma directa, sino codificado en varios bytes, lo que requiere un tratamiento más complejo. En primer lugar, se separan los datos en sus componentes más básicos: el byte menos significativo (LSB) y el byte más significativo (MSB), que representan respectivamente la parte baja y la parte alta del valor. Una vez realizada esta separación, se reconstruye el valor bruto combinando ambos bytes.

Posteriormente, dicho valor se descompone en dos partes: la mantisa y el exponente, siguiendo el formato sfloat de 16 bits en formato IEEE-1107. A continuación, se aplican los ajustes necesarios para interpretar correctamente el signo de ambos componentes, así como el valor del exponente. Finalmente, se calcula el valor real mediante la combinación de la mantisa y el exponente, obteniendo así la potencia en vatios. Este valor ya procesado se presenta al usuario de forma clara y comprensible en la interfaz de la aplicación.

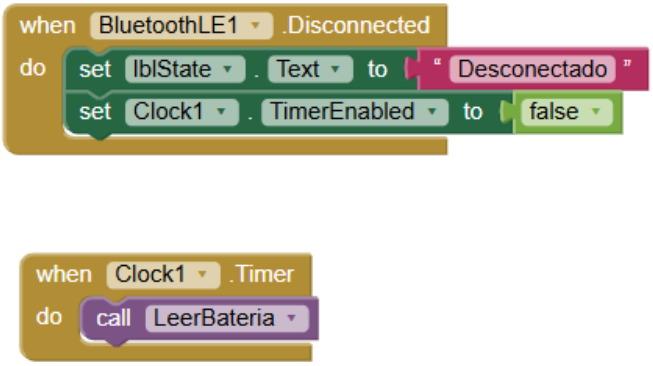


Ilustración 21: Lógica desconexión de BLE y finalización Clock1, 1º pantalla

Los dos últimos bloques correspondientes a la lógica de esta primera pantalla se muestran en la figura anterior y están relacionados con la gestión del estado de la conexión y la actualización periódica de la información del dispositivo.

En el primero de ellos se define el comportamiento de la aplicación cuando se produce la desconexión del dispositivo Bluetooth. En este caso, se actualiza el estado mostrado en la interfaz de usuario a *"Desconectado"*, con el fin de informar al usuario de la pérdida de conexión. Asimismo, se desactiva el reloj interno de la aplicación, estableciendo su valor en falso, lo que implica la detención de todas aquellas tareas periódicas que dependían de dicho temporizador, como la lectura periódica de datos.

Por otro lado, el segundo bloque se encarga de la actualización periódica del estado de la batería. Para ello, se realiza una llamada al procedimiento LeerBatería dentro de un evento controlado por el reloj interno. De este modo, se consigue que la lectura del nivel de batería se ejecute de forma recurrente a intervalos regulares, en este caso cada 5 minutos. Esta estrategia permite mantener la información actualizada en tiempo real, mejorando la monitorización del dispositivo sin necesidad de intervención por parte del usuario.

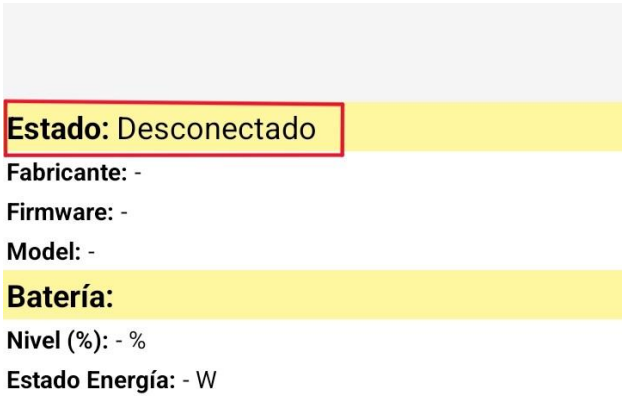


Ilustración 22: 1º pantalla, estado desconectado

Toda la lógica implementada en esta primera pantalla ha sido diseñada para establecer comunicación únicamente con dispositivos que dispongan de los servicios y características BLE identificados mediante los UUID específicos designados para el sensor BluEMG. De este modo, aunque durante el escaneo puedan detectarse múltiples dispositivos Bluetooth Low Energy cercanos, la aplicación solamente podrá establecer una comunicación funcional con aquellos dispositivos compatibles con el sistema desarrollado.

En caso de intentar conectarse a un dispositivo BLE diferente a BluEMG, la aplicación no será capaz de localizar las características necesarias para la recepción de los datos electromiográficos, debido a que los UUID asociados a dichos servicios serán distintos a los esperados. Como consecuencia, el estado de la conexión nunca pasará a "Conectado" y, por tanto, no se mostrará ningún dato en pantalla correspondiente a ese dispositivo.

Asimismo, si el usuario intenta acceder a la funcionalidad "Visualizar señal" sin que exista previamente una conexión válida con un sensor compatible, la aplicación abrirá la segunda pantalla, pero mostrará un mensaje de error indicando que no se encuentra ningún dispositivo Bluetooth Low Energy conectado.

4.1.2 LÓGICA DE LA PANTALLA PRINCIPAL

Ilustración 23: Pantalla principal con canales 1 y 3 seleccionados

La siguiente interfaz que se va a analizar y explicar detalladamente corresponde a la pantalla principal de nuestra aplicación. En esta pantalla se concentran las funcionalidades más importantes de la aplicación, ya que es el punto desde el cual se realiza la adquisición, visualización y gestión de los datos obtenidos por el sensor BluEMG. Debido a su relevancia dentro del funcionamiento global del proyecto, resulta fundamental comprender la estructura y lógica implementadas en cada uno de sus apartados.

La pantalla principal se encuentra organizada en cuatro bloques principales, cada uno de ellos encargado de gestionar una funcionalidad específica dentro de la aplicación. Esta

división modular permite mantener una estructura clara, mejorar la organización del código y facilitar tanto el mantenimiento como la escalabilidad futura del sistema.

El primer bloque es el más importante de toda la aplicación, ya que contiene la lógica encargada de la recepción y procesamiento de los datos enviados por el dispositivo BluEMG mediante tecnología Bluetooth Low Energy (BLE). Este bloque se ocupa de establecer la comunicación con el sensor, recibir los datos en tiempo real, ordenarlos correctamente y procesarlos para su posterior representación gráfica en pantalla. Además, cada canal de adquisición dispone de su propio elemento gráfico tipo *Canvas*, donde se dibuja y actualiza continuamente la señal electromiográfica obtenida a través de dicho canal. Gracias a este sistema, el usuario puede visualizar de manera dinámica y precisa la evolución de las señales capturadas por el sensor durante el proceso de adquisición.

El segundo bloque está orientado a una de las funcionalidades que contiene el sensor: la modificación de la ganancia de la señal. Este apartado permite ajustar la amplificación de la señal enviada por el dispositivo con el objetivo de adaptarla a diferentes necesidades que puedan surgir. Las opciones de ganancia disponibles son x1, x2, x5, x10, x20, x50, x100 y x200.

El tercer bloque engloba toda la lógica relacionada con la exportación y compartición de los datos generados por la aplicación. Este apartado es especialmente importante, ya que permite completar el flujo de trabajo para el cual ha sido diseñada la aplicación. Gracias a esta funcionalidad, el usuario puede enviar la información recopilada a través de diferentes medios disponibles en el dispositivo móvil, como correo electrónico, Bluetooth o cualquier otra aplicación compatible instalada en el teléfono. De este modo, se facilita el intercambio, almacenamiento y posterior análisis de los datos adquiridos.

Por último, y aunque se trate en parte de una funcionalidad ya presente en otra sección de la aplicación, se ha considerado interesante mantenerla también accesible desde esta segunda pantalla para facilitar la consulta al usuario. Esta última funcionalidad muestra en la parte inferior de la interfaz, concretamente debajo de la cuarta señal, algunas características relevantes sobre el sensor que pueden resultar útiles mientras se trabaja con este. Estas

características son: el modelo del dispositivo, la versión del firmware, el porcentaje de batería restante y la potencia de carga o descarga del dispositivo inalámbrico.

4.1.2.1 Adquisición y representación de señales

Esta parte de la aplicación constituye el bloque más extenso y, al mismo tiempo, el más relevante de todo el sistema, ya que sobre ella se sustenta el resto de las funcionalidades y servicios implementados en la aplicación. Es en esta sección donde se llevan a cabo tanto la recepción y el procesamiento de las señales EMG enviadas por el sensor BluEMG como su posterior representación gráfica en tiempo real.

La lógica implementada en esta parte puede dividirse principalmente en dos grandes bloques funcionales:

1) Adquisición de las señales

   Este bloque se encarga de recibir y reorganizar los bytes enviados por el sensor a través de Bluetooth Low Energy con el objetivo de reconstruir correctamente las señales correspondientes a cada uno de los canales activos. Debido a que los datos enviados por el dispositivo llegan mezclados según el número de canales habilitados, resulta necesario implementar una lógica capaz de separar y ordenar cada muestra en su canal correspondiente.

   Además, cabe destacar que la frecuencia de muestreo efectiva depende directamente del número de canales seleccionados. Cuantos más canales se encuentren activos simultáneamente, menor será el número de muestras recibidas por canal dentro de cada paquete de datos enviado por el sensor. Por esta razón, la lógica de adquisición debe adaptarse dinámicamente a cada una de las posibles configuraciones de canales, garantizando siempre la correcta reconstrucción de las señales independientemente del número de canales activos.

Una vez organizados y procesados, los datos obtenidos se almacenan tanto para su representación gráfica en tiempo real como para su posterior exportación en formato CSV.

2) Representación de las señales

El segundo gran bloque corresponde a la representación gráfica de las señales EMG obtenidas. Para ello, se utilizan los cuatro Canvas disponibles en la interfaz principal de la aplicación, uno para cada canal posible.

Con el objetivo de simplificar la estructura del programa y mantener una lógica organizada, la representación gráfica se ha dividido en cuatro procedimientos independientes denominados drawSignal1, drawSignal2, drawSignal3 y drawSignal4. Aunque cada uno de estos procedimientos trabaja con los datos correspondientes a un canal distinto, todos comparten exactamente la misma base lógica y el mismo sistema de representación gráfica.

Asimismo, los ejes de coordenadas utilizados son idénticos en los cuatro Canvas, garantizando una representación uniforme de las señales y facilitando así la comparación visual entre los distintos canales adquiridos simultáneamente.

Empezaremos con la lógica relacionada con la adquisición de los datos:

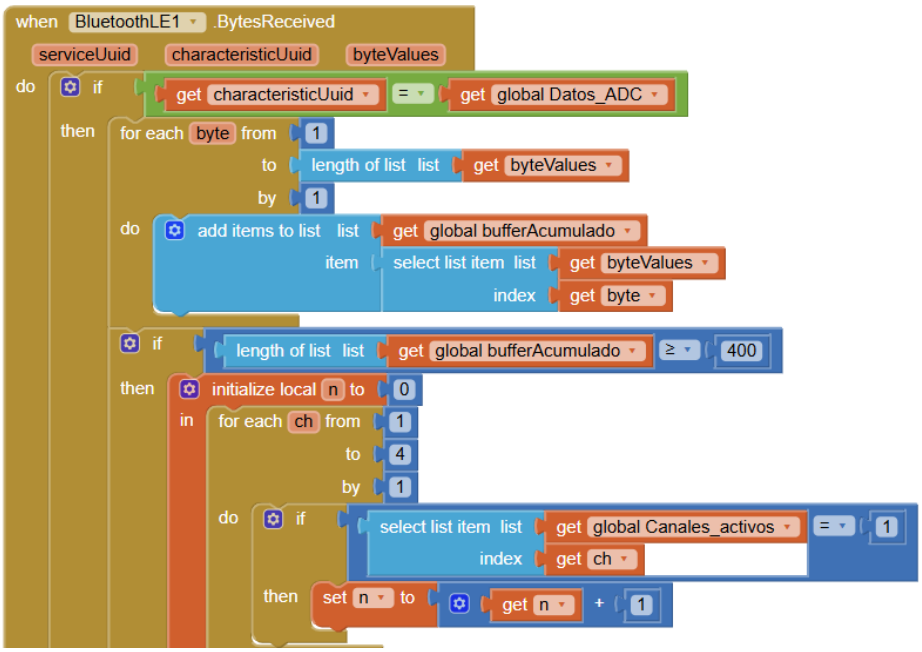


Ilustración 24: Lógica recepción de bytes para reconstrucción de señal, parte 1

En primer lugar, el bloque comienza comprobando mediante una estructura condicional if…then que la característica BLE desde la cual se están recibiendo los bytes corresponde efectivamente al servicio esperado (ADC Service). Esta verificación resulta necesaria para evitar errores de procesamiento y garantizar que únicamente se trabajará con los datos procedentes de la característica (Datos ADC) encargada de transmitir las señales EMG.

Seguidamente, se implementa un bucle for each byte que recorre todos los elementos de la lista byteValues, desde la posición 1 hasta la longitud total de la lista recibida. Durante cada iteración, el byte correspondiente se añade a una variable global denominada *"bufferAcumulado"*.

La finalidad de este procedimiento es asegurar que el procesamiento de los datos se realice siempre sobre bloques completos de 400 bytes. Debido a la posibilidad de mandar 1, 2, 3 o 4 canales, en caso de seleccionar únicamente 3 canales, los 400 bytes no se pueden dividir en 3 de forma entera, así que se descartan los últimos 4 bytes, y por tanto, solamente se usan 396 bytes. Aunque el sensor está diseñado para enviar paquetes de este tamaño, en ocasiones

la transmisión BLE puede fragmentar los datos y hacer que los bytes lleguen en varios paquetes más pequeños. Por este motivo, resulta más seguro almacenar temporalmente todos los bytes recibidos dentro de *"bufferAcumulado"* hasta reunir la cantidad completa necesaria para procesar correctamente las señales.

Una vez almacenados los datos, se introduce un nuevo bloque condicional if…then encargado de comprobar si la longitud de la lista *"bufferAcumulado"* es igual o superior a 400 bytes. Cuando esta condición se cumple, significa que ya se dispone de un bloque completo de datos válido y puede comenzar el procesamiento de las señales.

Dentro del bloque then, se inicializa una nueva variable local denominada *"n"*. Esta variable será utilizada para almacenar el número de canales activos simultáneamente en ese instante. Aunque la lista *"Canales_activos"* contiene el estado de los canales, dicha lista mantiene siempre una longitud fija de cuatro posiciones, independientemente del número real de canales habilitados. Por ello, resulta necesario calcular dinámicamente cuántos canales se encuentran activos realmente.

Para realizar este cálculo, se implementa un bucle for each ch que recorre los valores desde 1 hasta 4, correspondientes a los cuatro canales disponibles del sistema. Durante cada iteración, se comprueba si el valor almacenado en la posición correspondiente de la lista *"Canales_activos"* es igual a 1. En caso afirmativo, significa que dicho canal está activo y, por tanto, se incrementa en una unidad el valor actual de la variable *"n"*.

Al finalizar este proceso, la variable *"n"* contendrá el número exacto de canales activos, información que resultará fundamental para interpretar correctamente la distribución de los datos dentro de los 400 bytes recibidos y reconstruir adecuadamente las señales correspondientes a cada canal.

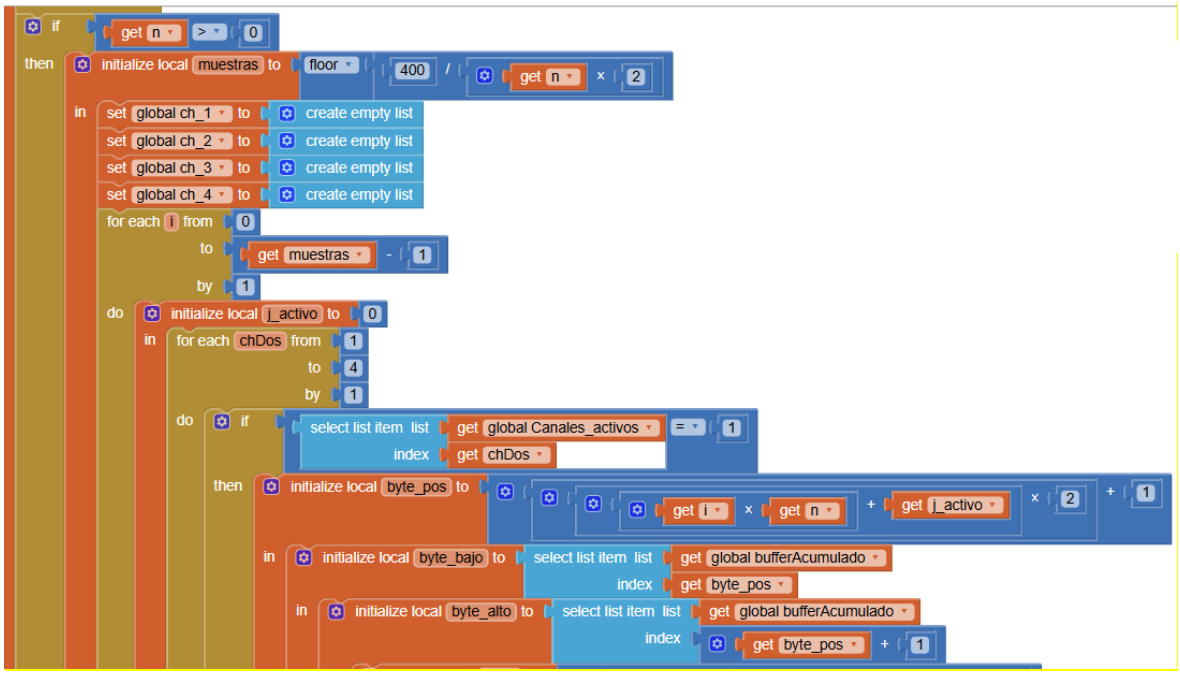


Ilustración 25:Lógica recepción de bytes para reconstrucción de señal, parte 2

Una vez calculado el valor de *"n"*, correspondiente al número de canales activos simultáneamente, se inicializa una nueva variable local denominada *"muestras"*. Esta variable se calcula como el valor entero inferior (*floor*) de la operación 400/ (n*2). El motivo de este cálculo es que cada muestra adquirida ocupa 2 bytes, por lo que el número total de muestras disponibles por canal depende directamente del número de canales activos dentro de los 400 bytes recibidos.

Seguidamente, se asegura de que las listas ch1, ch2, ch3 y ch4, utilizadas para almacenar temporalmente las muestras correspondientes a cada canal, se encuentren vacías antes de comenzar el procesamiento. Este paso resulta fundamental para evitar la mezcla de datos pertenecientes a paquetes anteriores, lo que produciría errores tanto en la representación gráfica como en el almacenamiento posterior de las señales.

Una vez inicializadas correctamente las listas, se implementa un nuevo bucle for each i, que recorre los valores desde 0 hasta muestras-1, avanzando de uno en uno. Este bucle será el

encargado de procesar individualmente cada muestra contenida dentro del bloque de datos recibido.

Dentro de este bucle se inicializa una variable local denominada *"j_activo"* con valor 0. La función de esta variable es actuar como contenedor auxiliar de los canales activos encontrados durante el procesamiento de cada muestra. A medida que se recorren los distintos canales y se detectan canales habilitados, el valor de *"j_activo"* irá incrementándose para permitir calcular correctamente la posición de los bytes correspondientes a cada canal dentro del paquete de datos recibido.

Posteriormente, se introduce un segundo bucle for each chDos, que recorre los valores desde 1 hasta 4, correspondientes a los cuatro canales disponibles en el sistema. Dentro de este bucle se comprueba si el valor almacenado en la posición correspondiente de la lista global *"canales_activos"* es igual a 1. Si la condición se cumple, significa que dicho canal se encuentra habilitado y, por tanto, se continúa con el procesamiento de la muestra correspondiente.

A continuación, se inicializa una nueva variable local denominada *"byte_pos"*. Esta variable representa la posición exacta del byte dentro de la lista "*bufferAcumulado"* que contiene todos los bytes recibidos desde el sensor. Su cálculo se realiza mediante la expresión:

$$byte_{pos} = \Big(\big((i * n) + j_{activo}\big) * 2\Big) + 1$$

Esta fórmula permite localizar correctamente el inicio de cada muestra dentro del flujo de datos, teniendo en cuenta tanto el número de muestra actual (i) como el número de canales activos (n) y el índice del canal activo procesado *("j_activo")*.

Una vez calculada la posición correspondiente, se obtiene el valor de *"byte_bajo"* tomando el byte almacenado en la posición *"byte_pos"* de la lista *"bufferAcumulado"*. Del mismo modo, se inicializa la variable local *"byte_alto"* utilizando el byte almacenado en la posición *"byte_pos"* + 1. Estos dos bytes representan conjuntamente una única muestra digitalizada enviada por el sensor, codificada en formato Little-endian.

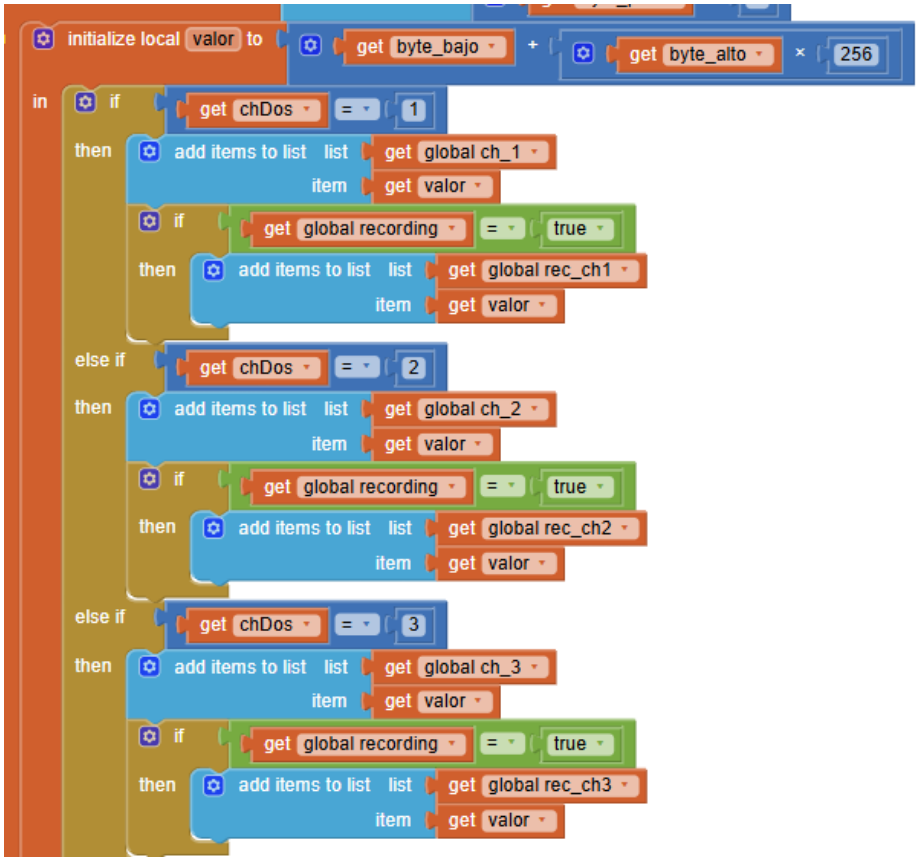


Ilustración 26: Lógica recepción de bytes para reconstrucción de señal, parte 3

Una vez obtenidas las partes inferior y superior de cada una de las muestras digitales, resulta necesario combinar ambos bytes para reconstruir el valor completo de interés correspondiente a la señal EMG adquirida.

Para ello, se define una nueva variable local denominada *"valor"*, la cual se inicializa mediante la suma del byte menos significativo *("byte_bajo")* y el byte más significativo *("byte_alto")* multiplicado por 256. El cálculo realizado es el siguiente:

$$valor = byte_{bajo} + byte_{alto} * 256$$

Esta operación permite reconstruir correctamente el valor numérico original enviado por el sensor a partir de los dos bytes recibidos en formato little-endian.

Una vez obtenido el valor completo de la muestra, el siguiente paso consiste en almacenarlo en la lista correspondiente según el canal del que provenga. Para ello, se implementa una estructura condicional compuesta por varios bloques if…then…else if…then, en los cuales se comprueba el valor de la variable *"chDos"*.

Cada condición verifica si chDos corresponde a uno de los cuatro canales posibles (1,2,3 o 4). Dependiendo del canal identificado, el valor calculado se añade a la lista correspondiente (ch1, ch2, ch3 o ch4). Por ejemplo, si *chDos*=3, el valor obtenido se añadirá a la lista ch3, encargada de almacenar temporalmente las muestras pertenecientes al tercer canal.

Además, dentro de cada bloque then se incorpora una nueva condición if…then adicional encargada de comprobar el estado de la variable global *"recording"*. Esta variable actúa como identificador de si la grabación de datos se encuentra activa o no. Si su valor es true, el dato procesado no solo se almacena en la lista temporal utilizada para la representación gráfica, sino que también se añade a la lista global de grabación correspondiente (rec_ch1, rec_ch2, rec_ch3, rec_ch4).

Gracias a esta lógica, la aplicación puede simultáneamente representar las señales en tiempo real y almacenar los datos adquiridos para su posterior exportación en formato CSV cuando la función de grabación se encuentra habilitada.

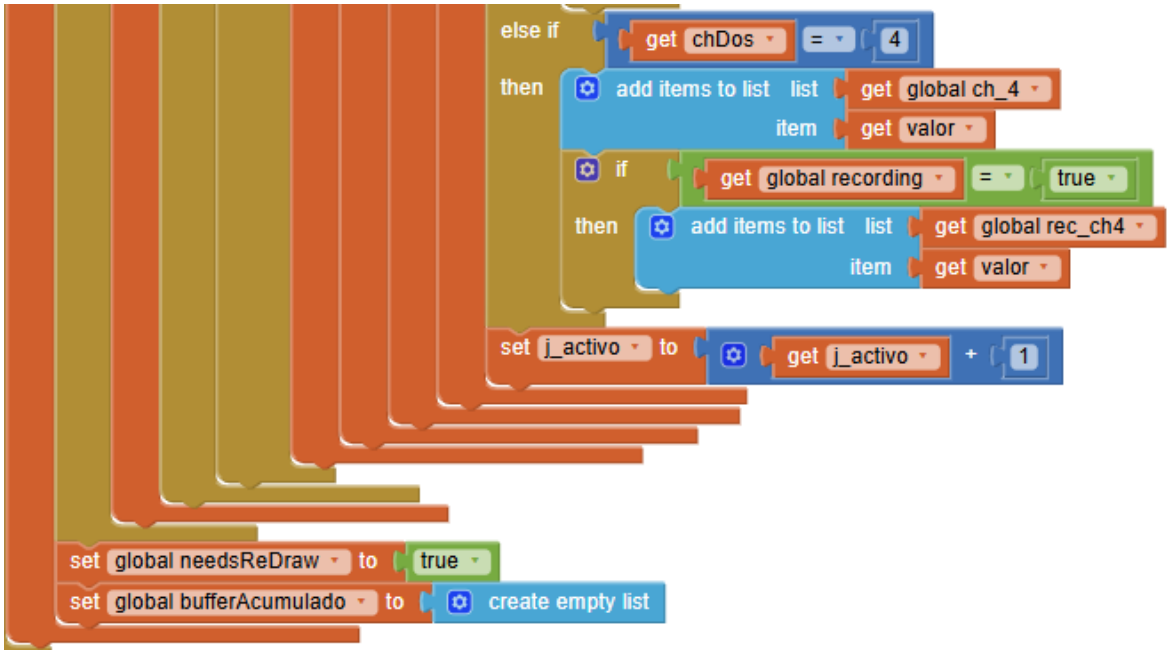


Ilustración 27: Lógica recepción de bytes para reconstrucción de señal, parte 4

Por último, fuera de todos los bloques condicionales if…then anteriores, se actualiza el valor de la variable *"j_activo"*, configurándola como la suma de su valor actual más uno. Esta operación se realiza para avanzar al siguiente canal activo dentro del paquete de datos recibido, permitiendo que el cálculo de posiciones de los bytes continúe realizándose correctamente durante las siguientes iteraciones del bucle. Gracias a este incremento progresivo, la aplicación puede identificar adecuadamente qué muestras pertenecen a cada canal activo y reconstruir las señales en el orden correcto.

Posteriormente, fuera del bucle for each i from 0 to muestras-1, pero todavía dentro del bloque condicional que comprueba que "n" sea mayor que 0, se configura la variable de control *"needsReDraw"* con valor true. Este *flag* actúa como un indicador para la aplicación, señalando que ya se dispone de nuevos datos correctamente procesados y que, por tanto, las señales pueden representarse gráficamente en los distintos Canvas correspondientes a cada canal.

Finalmente, se vacía el contenido de la lista *"bufferAcumulado"*. Este paso resulta esencial para evitar que bytes pertenecientes al bloque de datos previamente procesado permanezcan almacenados y puedan mezclarse con los nuevos datos recibidos en futuras iteraciones. De esta manera, cada ciclo de procesamiento comienza siempre con un buffer limpio, garantizando así la correcta adquisición y reconstrucción de las señales EMG.

Se prosigue con la lógica correspondiente al procedimiento drawSignal1. Puesto que los cuatro procedimientos son prácticamente iguales, se describirá únicamente el primer bloque de forma detallada.

La representación gráfica de las señales electromiográficas se realiza mediante el redibujado periódico del contenido del Canvas conforme se reciben nuevas muestras desde el sensor BluEMG. Para ello, la aplicación trabaja con listas dinámicas que almacenan temporalmente los datos recibidos de cada canal y posteriormente son recorridas para dibujar la señal punto a punto mediante segmentos de línea consecutivos.

Antes de representar una nueva ventana de datos, el Canvas se limpia completamente y se vuelve a dibujar tanto la rejilla de referencia como la señal electromiográfica correspondiente. El desplazamiento horizontal de la señal (*scrolling*) se consigue, por tanto, mediante el refresco continuo de toda la ventana gráfica en tiempo real, en lugar de desplazar físicamente la imagen ya dibujada.

La aplicación no implementa actualmente una estructura específica de tipo *ring buffer* o buffer circular. En su lugar, se utilizan listas dinámicas donde se almacenan temporalmente las muestras necesarias para la representación gráfica de cada canal. A partir de dichas listas, se calcula automáticamente la separación horizontal entre puntos en función del ancho disponible en el Canvas y del número de muestras recibidas en cada actualización.

Asimismo, la representación gráfica incorpora un escalado vertical configurable mediante valores mínimos y máximos globales, permitiendo adaptar la visualización de la amplitud de la señal. Sin embargo, la densidad horizontal de representación permanece fija y no

dispone actualmente de funcionalidades de zoom temporal o modificación dinámica de la ventana visible de datos.

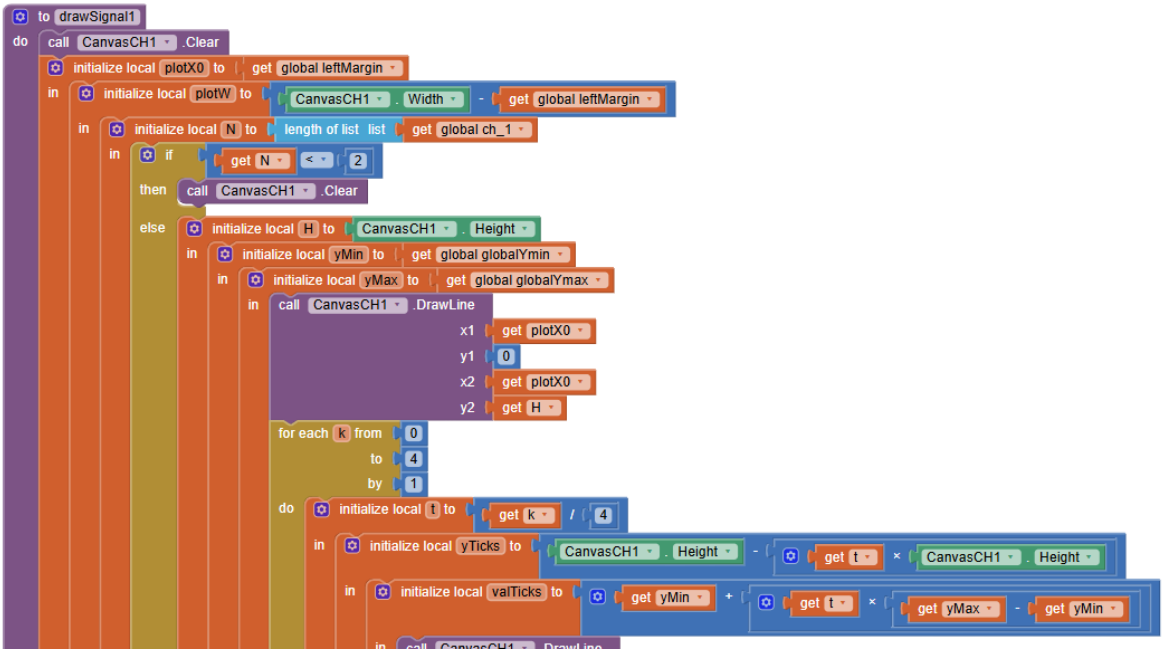


Ilustración 28: Procedimiento drawSignal1 , parte 1

Este procedimiento comienza limpiando el Canvas, sobre el cual se representan las señales EMG. Esta operación resulta necesaria para evitar que las nuevas señales se dibujen sobre representaciones anteriores, lo que dificultaría la visualización correcta de los datos y produciría una representación gráfica confusa.

A continuación, se inicializa una variable local denominada *"plotX0"*, asignándole el valor de la variable global *"leftMargin"*. Esta variable representa el margen izquierdo utilizado para comenzar la zona de dibujo de la gráfica dentro del Canvas, dejando espacio suficiente para representar los ejes y las posibles referencias visuales.

Seguidamente, se inicializa la variable local *"plotW"*, cuyo valor corresponde al total del Canvas menos el margen izquierdo *"leftMargin"*. Esta variable define el ancho útil disponible para representar gráficamente las señales adquiridas.

Posteriormente, se crea una nueva variable local llamada *"N",* la cual almacena la longitud de la lista global *"ch_1"*. Este valor representa el número total de muestras disponibles para el canal correspondiente y será utilizado posteriormente para calcular la distribución de los puntos sobre el eje horizontal de la gráfica.

Una vez inicializadas todas estas variables, se implementa un bloque condicional if…then…else. En la condición if se comprueba si el valor de *"N"* es menor que 2. Esta verificación se realiza porque, para poder dibujar una señal correctamente, resulta necesario disponer de al menos dos puntos consecutivos entre los cuales trazar una línea. Si la condición se cumple, es decir, si no existen suficientes muestras disponibles, simplemente se limpia el Canvas.

En caso contrario, el procedimiento continúa con la lógica encargada de representar la señal gráficamente. Para ello, se inicializan tres nuevas variables locales:

- *"H",* que almacena la altura del Canvas.
- *"yMin",* inicializada con el valor de la variable global "*Ymin"*, correspondiente al límite inferior del eje vertical.
- *"yMax",* inicializada con el valor de la variable global *"Ymax"*, correspondiente al límite superior del eje vertical.

Una vez calculados todos los parámetros relacionados con márgenes, alturas y dimensiones de la gráfica, se llama al procedimiento interno DrawLine del componente Canvas. En esta llamada se utilizan los siguientes parámetros:

- x1= plotX0
- y1=0
- x2=plotX0
- y2=H

Esta línea vertical representa el eje principal de coordenadas de la gráfica.

Llegados a este punto, ya se dispone de los ejes básicos necesarios para comenzar la representación visual de la señal. A continuación, se implementa un bucle for each k, que recorre los valores desde 0 hasta 4, avanzando de uno en uno. La finalidad de este bucle es generar las divisiones o marcas del eje vertical de la gráfica.

Dentro del bucle se inicializa la variable local t, calculada como:

$$t = k/4$$

Esta variable representa una proporción normalizada entre 0 y 1, utilizada para distribuir uniformemente las divisiones a lo largo del eje vertical.

Posteriormente, se inicializan dos nuevas variables locales denominadas *"yTicks"* y *"valTicks":*

$$yTicks = H - t * H$$

$$valTicks = yMin + t * (yMax - yMin)$$

La variable "*yTicks"* representa la posición vertical exacta dentro del Canvas donde se dibujará cada marca o división del eje Y. Por otro lado, la variable *"valTicks"* representa el valor numérico asociado a dicha división, es decir, el valor real de amplitud de señal correspondiente a esa posición del eje vertical.

Gracias a estas variables, la aplicación puede generar automáticamente una escala visual uniforme sobre el eje Y, facilitando la interpretación de la amplitud de las señales EMG representadas en pantalla.

Seguidamente, como se muestra en la ilustración 29, se llama nuevamente al procedimiento DrawLine utilizando los siguientes parámetros:

- x1=plotX0 -5
- y1= yTicks
- x2=plotX0

- y2= yTicks

Además, también se llama al procedimiento DrawText introduciéndole los parámetros siguientes:

- text= round (valTicks)
- x=15
- y= yTicks-6

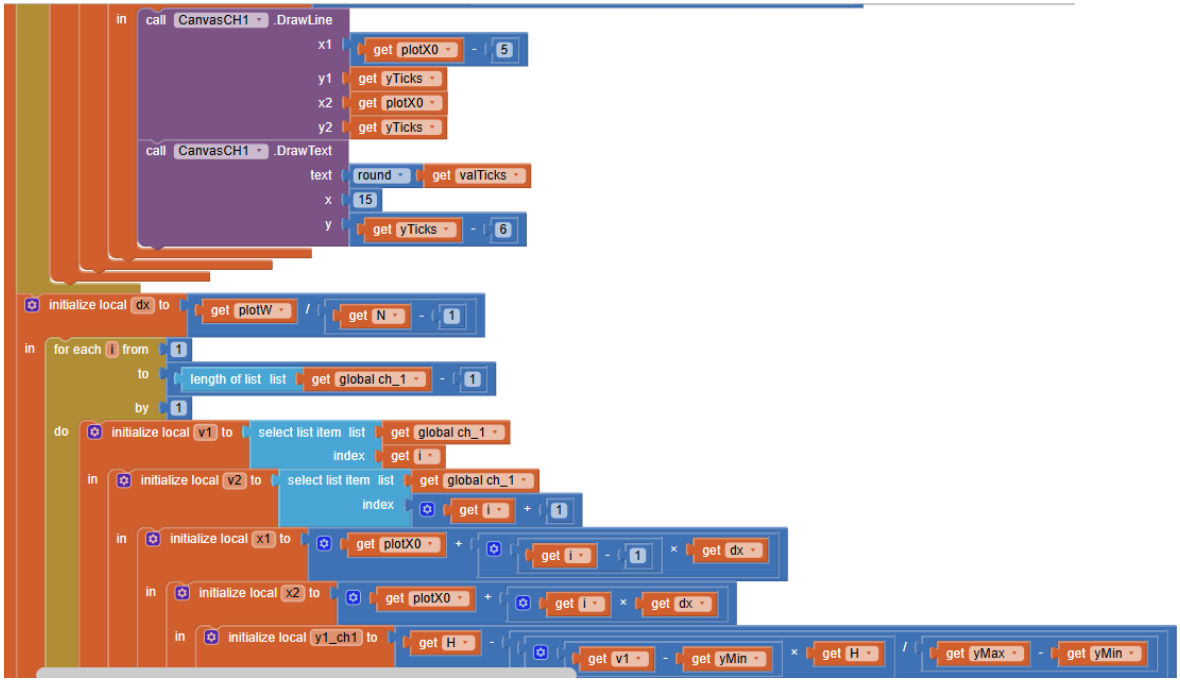


Ilustración 29:Procedimiento drawSignal1 , parte 2

Seguidamente, fuera del bucle for each k, el procedimiento continúa con la lógica encargada de dibujar la señal propiamente dicha. Una vez definidos los ejes y las divisiones de referencia, se procede a representar gráficamente los valores almacenados en la lista correspondiente al canal.

Para ello, se inicializa una nueva variable denominada dx, calculada a partir del ancho útil de la gráfica *("plotW")* y del número total de muestras disponibles *("N")*. Esta variable

representa la separación horizontal entre dos muestras consecutivas, permitiendo distribuir todos los puntos de la señal de forma uniforme a lo largo del eje X.

A continuación, se introduce un nuevo bucle for each i, que recorre los valores desde 0 hasta la longitud de la lista ch1-1, avanzando de uno en uno. Este bucle permite tomar pares de muestras consecutivas para unirlas mediante líneas y formar así la representación continua de la señal.

Dentro del bucle se inicializan varias variables locales:

- v1= contiene el valor del elemento i de la lista ch_1
- v2= contiene el valor del elemento i+1 de la lista ch_1
- x1= plotX0 + (i-1) * dx . Representa la posición horizontal del primer punto.
- x2=plotX0 + (i*dx) . Representa la posición horizontal del segundo punto.
- y1_ch1= H – (((v1 - yMin)* H)/ (yMax - yMin)) . Representa la posición vertical del primer punto dentro del Canvas.

El valor de y1_ch1 se calcula escalando la muestra v1 respecto a los límites verticales definidos por *"yMin"* y *"yMax"*. Además, se resta el resultado a la altura total del Canvas, ya que en App Inventor el eje vertical crece hacia abajo. De esta forma, los valores altos de la señal se representan en la parte superior del gráfico y los valores bajos en la parte inferior.

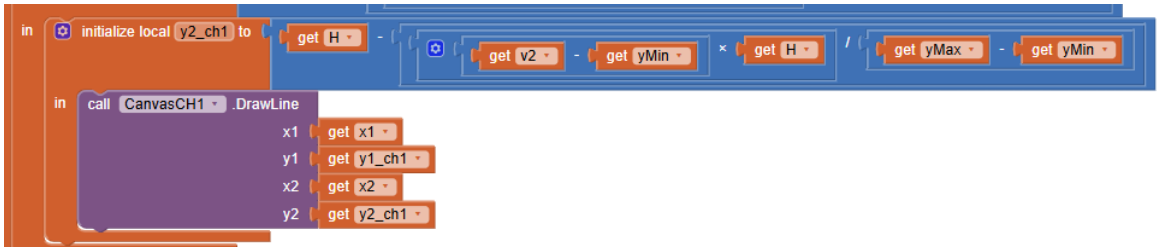


Ilustración 30:Procedimiento drawSignal1 ,parte 3

Por último, se define una variable local más *"y2_ch1"*, la cual representa la posición vertical del segundo punto de la señal dentro del Canvas. Esta variable se utiliza para determinar la

coordenada Y correspondiente a la segunda muestra (v2) que será utilizada al dibujar el segmento de línea entre dos puntos consecutivos de la señal

El valor de *"y2_ch1"* se calcula mediante la siguiente expresión:

$$y2_{ch1} = H - (((v2 - yMin) * H)/(yMax - yMin))$$

Esta operación realiza un escalado del valor de la muestra v2 respecto a los límites verticales definidos por *"yMin"* y *"yMax"*, adaptándolo a las dimensiones reales del Canvas. Del mismo modo que ocurría con *"y1_ch1",* el valor obtenido se resta de la altura total del Canvas debido a que el sistema de coordenadas de App Inventor utiliza un eje vertical invertido, donde los valores mayores de *"Y*" se representan más abajo en pantalla.

Una vez calculadas todas las coordenadas necesarias, se llama al procedimiento interno DrawLine del componente Canvas, encargado de dibujar el segmento de línea entre las dos muestras consecutivas de la señal. Para ello, se utilizan los siguientes parámetros:

- x1= x1
- y1= y1_ch1
- x2= x2
- y2= y2_ch1

Toda la lógica descrita anteriormente, relacionada con la representación gráfica de las señales en los distintos Canvas, se ejecuta de forma periódica utilizando el temporizador asociado a Clock1. El período configurado en este temporizador determina la frecuencia con la que la aplicación actualiza visualmente las señales EMG mostradas en pantalla, la cual es de 100 Hz actualmente, pero puede modificarse fácilmente si fuera necesario.

Gracias a este mecanismo, la representación gráfica se realiza de manera continua y en tiempo real, permitiendo que las nuevas muestras procesadas se dibujen automáticamente sobre el Canvas correspondiente a cada canal activo. De esta forma, el usuario puede visualizar la evolución de las señales mientras el sensor continúa enviando datos mediante Bluetooth Low Energy.

A continuación, se muestra el momento exacto en el que se invoca cada uno de los procedimientos encargados de dibujar las señales correspondientes a los distintos canales disponibles en la aplicación.

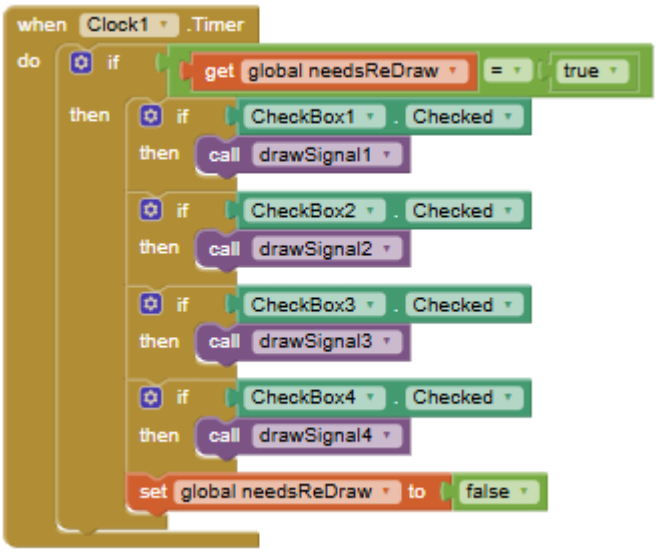


Ilustración 31: Lógica finalización Clock1 pantalla principal

Como se muestra en la imagen anterior, una vez transcurrido el tiempo configurado en Clock1, se ejecuta automáticamente una serie de acciones relacionadas con la actualización gráfica de las señales EMG representadas en pantalla.

En primer lugar, se comprueba si el valor de la variable *"needsReDraw"* es igual a true. Este flag actúa como un indicador que informa a la aplicación si existen nuevos datos procesados pendientes de ser representados gráficamente. Si la condición se cumple, significa que es necesario actualizar los gráficos de los distintos canales.

A continuación, se implementan cuatro bloques condicionales if…then, uno para cada uno de los posibles canales disponibles en la aplicación. En cada bloque se comprueba si el CheckBox correspondiente al canal se encuentra seleccionado. Si un canal está activo, se llama al procedimiento encargado de dibujar la señal asociada a dicho canal en su correspondiente Canvas.

Finalmente, una vez completada la actualización gráfica de todos los canales activos, se configura nuevamente la variable *"needsReDraw"* con el valor false. De esta manera, la aplicación indica que los datos ya han sido representados correctamente y evita realizar redibujados innecesarios hasta que se reciban y procesen nuevas muestras desde el sensor.

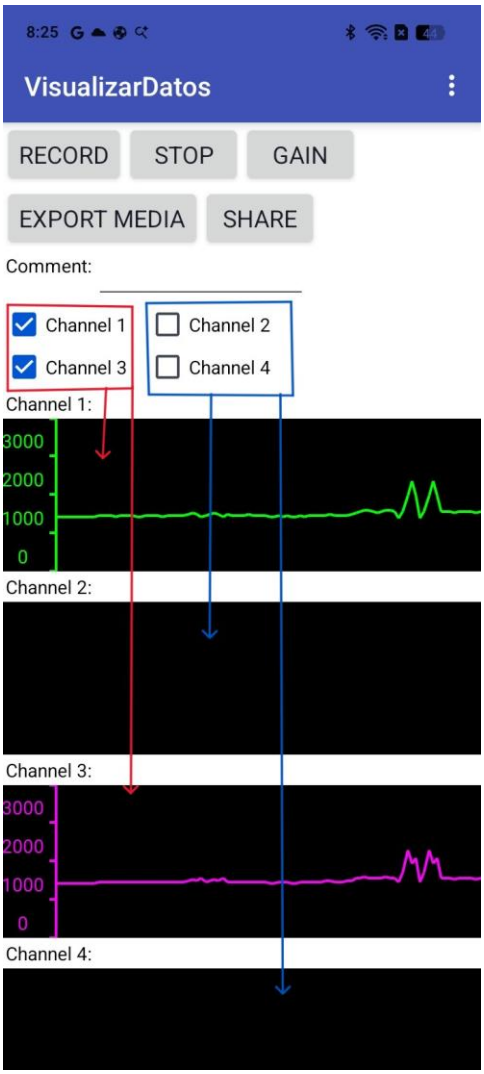


Ilustración 32: Pantalla principal, canales 1 y 3 seleccionados

4.1.2.2 Otras funcionalidades

El conversor analógico-digital integrado en el sensor BluEMG dispone de diversos servicios BLE tanto de escritura como de lectura, además del servicio principal encargado de la

transmisión de los datos EMG adquiridos. Estos servicios permiten configurar distintos parámetros de funcionamiento del dispositivo de manera remota desde la aplicación móvil, proporcionando así un mayor control sobre el comportamiento del sensor y optimizando tanto el consumo energético como la adquisición de señales.

Los principales servicios implementados son los siguientes:

1) Conv On/Off

   Este servicio está formado por un único byte cuya función es habilitar o deshabilitar las conversiones analógico-digitales realizadas por el sensor. Cuando el valor enviado es 1, las conversiones se activan y el dispositivo comienza a realizar mediciones y transmitir datos. Por el contrario, cuando el valor es 0, las conversiones se deshabilitan, permitiendo reducir significativamente el consumo energético del sistema cuando no se requiere adquirir señales.

   Se trata de una característica exclusivamente de escritura, ya que la aplicación únicamente necesita enviar órdenes al dispositivo para controlar el inicio o la detención de las mediciones.

2) Ganancia

   Este servicio consiste también en un único byte utilizado para configurar la ganancia aplicada por el PGA (*Programmable Gain Amplifier*) integrado en la tarjeta electrónica del sensor. La ganancia determina el nivel de amplificación aplicado a la señal analógica antes de ser digitalizada, permitiendo adaptar la amplitud de la señal EMG a las características de adquisición del conversor analógico-digital.

   Los valores admitidos actualmente son los siguientes:

   - 0 → Ganancia x1
   - 1 → Ganancia x2
   - 2 → Ganancia x5
   - 3 → Ganancia x10
   - 4 → Ganancia x20
   - 5 → Ganancia x50
   - 6 → Ganancia x100

- 7 → Ganancia x200

En la implementación actual, la misma configuración de ganancia se aplica simultáneamente a todos los canales disponibles. Sin embargo, el sistema podría ampliarse para permitir una configuración independiente de la ganancia en cada canal. En ese caso, serían necesarios 12 bits en total, utilizando 3 bits para almacenar la configuración de cada uno de los cuatro canales.

Al igual que el servicio anterior, esta característica es exclusivamente de escritura.

3) Canales On/Off

Este servicio se implementa mediante un vector de 4 bytes cuya función es indicar qué canales deben ser muestreados por el sistema. Cada byte del vector representa el estado de un canal concreto:

- El byte 0 indica si el canal 1 está activo (1) o desactivado (0).
- El byte 1 controla el estado del canal 2.
- El byte 2 controla el estado del canal 3.
- El byte 3 controla el estado del canal 4.

Gracias a este sistema, es posible habilitar únicamente los canales necesarios en cada momento, consiguiendo una mayor resolución, ya que la frecuencia de muestreo total es de 10 kHz y ha de compartirse entre los 4 canales.

A continuación, se describirá detalladamente cómo se ha implementado cada uno de estos servicios dentro de los bloques de programación de la aplicación.

1) Conversor On/Off

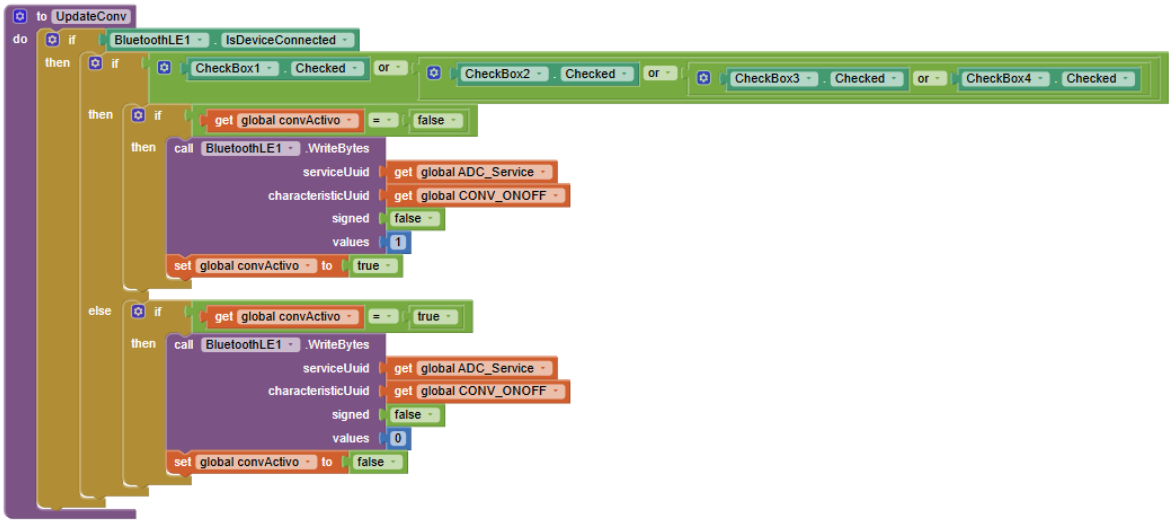


Ilustración 33: Procedimiento UpdateConv

En este bloque de código se implementa la lógica encargada de controlar el servicio Conv On/Off del sensor BluEMG, cuya función es habilitar o deshabilitar las conversiones analógico-digitales en función del estado de los canales seleccionados por el usuario.

En primer lugar, se introduce un bloque condicional if…then en el que se comprueba si existe algún dispositivo conectado mediante Bluetooth Low Energy. Esta verificación resulta necesaria para garantizar que únicamente se intenten enviar comandos al sensor cuando la conexión BLE se encuentre correctamente establecida. En caso de existir un dispositivo conectado, la ejecución continúa con la lógica incluida dentro del bloque then.

Dentro de este bloque se implementa una segunda estructura condicional if…then…else. En ella se comprueba si existe al menos un CheckBox seleccionado, es decir, si algún canal de adquisición se encuentra activo. Si se detecta que hay uno o más canales seleccionados, la ejecución continúa con la lógica definida dentro del bloque then. En caso contrario, se ejecuta la lógica contenida en el bloque else.

La lógica implementada en el bloque else tiene como objetivo detener las conversiones cuando no existe ningún canal activo. Para ello, se comprueba si la variable global *"convActivo"* se encuentra con el valor true, indicando que las conversiones estaban previamente activadas. Si esta condición se cumple, se envía mediante BLE un byte con valor 0 utilizando el UUID correspondiente al servicio Conv On/Off. Este valor indica al sensor que debe detener las conversiones analógico-digitales para reducir el consumo energético del dispositivo. Finalmente, se actualiza la variable global *"convActivo",* estableciéndola en false para reflejar el nuevo estado del sistema.

Por otro lado, cuando existe al menos un canal seleccionado, se comprueba si la variable *"convActivo"* se encuentra en false. En caso afirmativo, se envía a través del UUID correspondiente un byte con valor 1, indicando al sensor que debe iniciar las conversiones y comenzar la adquisición de datos. Tras realizar esta operación, la variable *"convActivo"* se actualiza a true, indicando que el sistema de conversión se encuentra activo.

Por defecto, al abrir la pantalla principal y establecerse la conexión con el dispositivo BLE, el valor del servicio Conv On/Off se configura inicialmente a 0. Esta decisión permite mantener deshabilitadas las conversiones mientras no exista ningún canal seleccionado, contribuyendo así al ahorro energético del sensor.

Además, el procedimiento descrito anteriormente se ejecuta automáticamente cada vez que se modifica el estado de cualquiera de los CheckBox asociados a los canales. De esta manera, la aplicación comprueba dinámicamente si las conversiones deben activarse o desactivarse en función de los canales seleccionados en cada momento.

2) Ganancia

La lógica relacionada con la ganancia se desarrolla en los siguientes bloques:

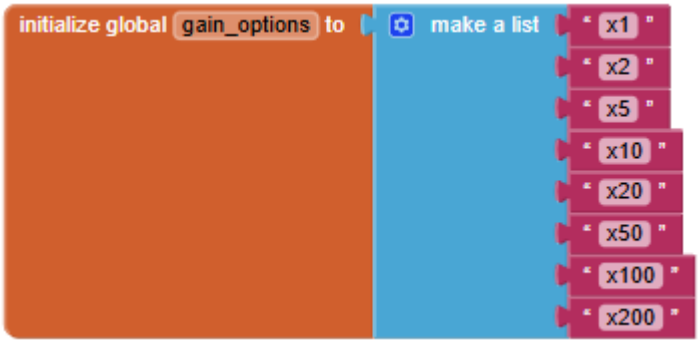


Ilustración 34: Variable global que contiene las opciones de ganancia

En primer lugar, se crea una variable global denominada *"gain_options"*. Esta variable consiste en una lista que almacena todas las configuraciones de ganancia compatibles con el sensor BluEMG.

La finalidad de esta lista es centralizar en una única estructura todas las opciones de ganancia disponibles, facilitando así tanto su acceso como su gestión desde los distintos bloques de programación de aplicación. Cada elemento de la lista representa uno de los niveles de ganancia admitidos por el PGA del sensor, permitiendo al usuario seleccionar el nivel de amplificación más adecuado según las características de la señal EMG que se desee adquirir.

Ilustración 35: Lógica asociada al botón "Gain"

Posteriormente, se implementa un bloque asociado al botón *"Gain"* de la pantalla principal, situado aproximadamente en la zona superior central de la interfaz, junto al botón *"Stop"*. La función de este botón es permitir al usuario acceder a la configuración de ganancia del sensor de forma sencilla e intuitiva.

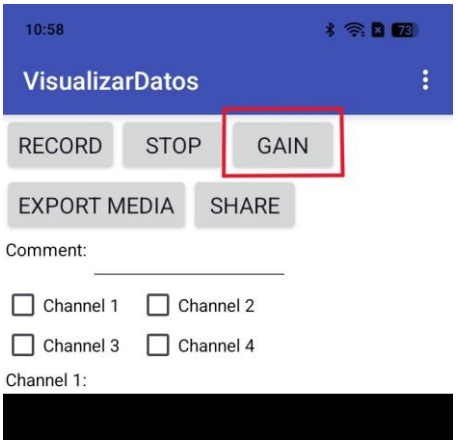


Ilustración 36: Pantalla principal, boton ganancia

Una vez abierto el ListPicker1 y seleccionada una de las opciones de ganancia, la aplicación continúa con la lógica implementada en el siguiente bloque, encargado de procesar la opción elegida y aplicar la nueva configuración al dispositivo.

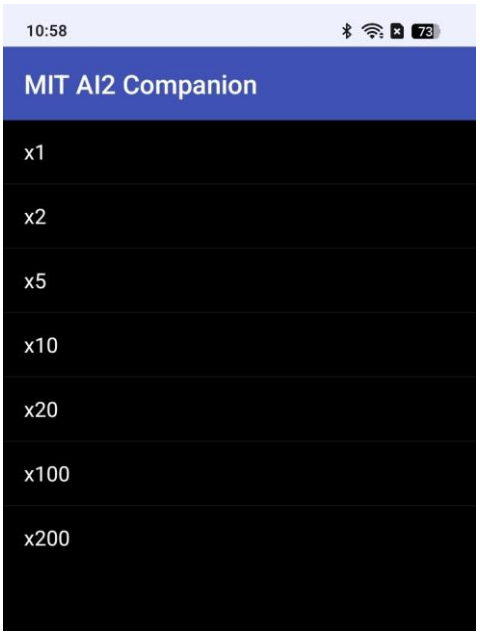


Ilustración 37: Pantalla principal, listPicker ganancia

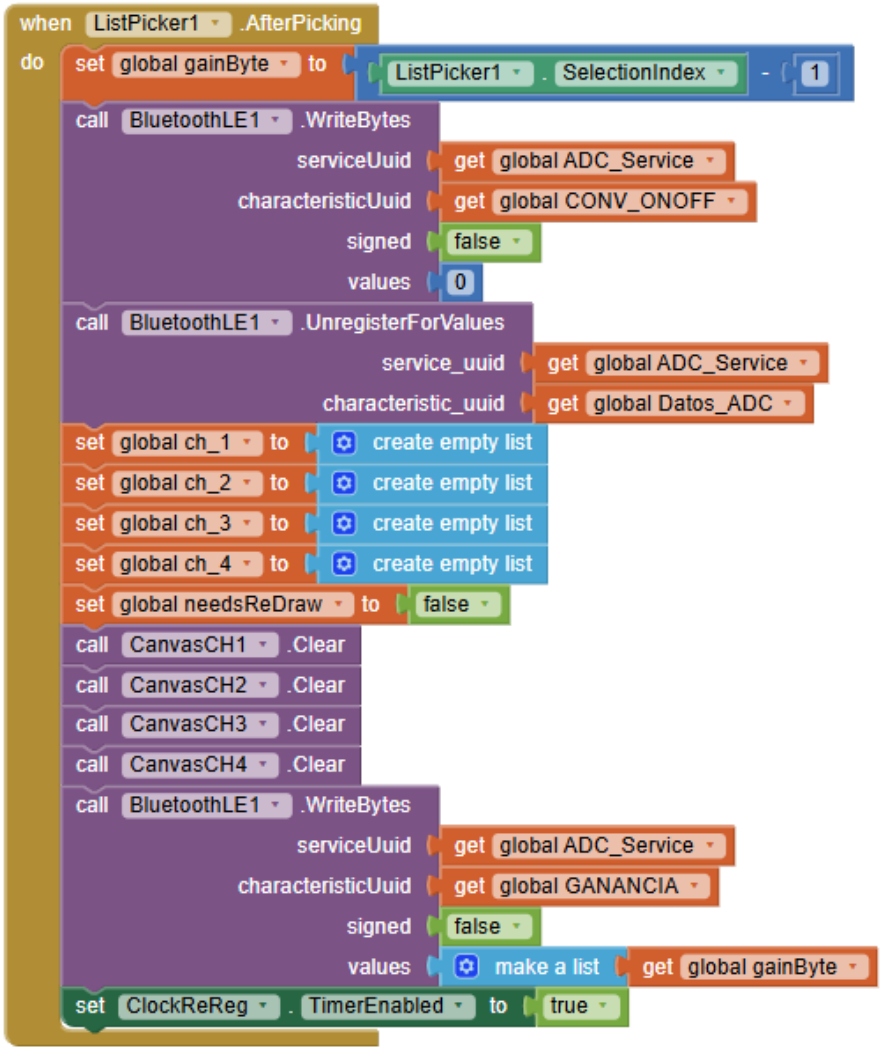


Ilustración 38: Lógica asociada a la selección de una opción de ganancia

Al inicio de este bloque y, como primer paso, se configura la variable global *"gainByte"* con el valor correspondiente al índice de la opción de ganancia seleccionada en ListPicker1, restándole posteriormente una unidad. Esta corrección resulta necesaria debido a que MIT App Inventor indexa las listas empezando desde 1, mientras que el sensor BluEMG interpreta los valores de ganancia comenzando desde 0.

A continuación, se envía mediante BLE un byte con valor 0 al servicio Conv On/Off, con el objetivo de detener temporalmente las conversiones analógico-digitales del sensor. Seguidamente, mediante el procedimiento UnregisterForValues, se cancela la

suscripción a las notificaciones BLE. Esto implica que el dispositivo deja momentáneamente de enviar datos a la aplicación, permitiendo realizar el cambio de configuración de forma segura y evitando inconsistencias durante el proceso.

Una vez detenidas todas las conversiones y comunicaciones relacionadas con la adquisición de datos, se reinicializan las listas asociadas al almacenamiento de las señales de cada uno de los canales. El objetivo de esta operación es garantizar que no se mezclen datos obtenidos con configuraciones de ganancia distintas. Aunque este reinicio no sería estrictamente imprescindible para el funcionamiento del sistema, sí contribuye a mantener una mayor coherencia en los datos mostrados y mejora el comportamiento general de la aplicación.

Posteriormente, la variable *"needsRedraw"* se configura con el valor false, deteniendo temporalmente el redibujado de las señales en pantalla mientras se realiza el cambio de ganancia. Además, se eliminan todas las señales que se encuentren dibujadas en ese instante sobre los distintos Canvas asociados a cada canal. De esta manera, la representación gráfica puede reiniciarse desde cero utilizando los nuevos valores de ganancia una vez aplicada la nueva configuración.

Después de preparar el sistema, se envía mediante el UUID correspondiente al servicio de ganancia una lista que contiene el valor almacenado en la variable *"gainByte"*. Con ello, el sensor recibe la nueva configuración de amplificación que deberá aplicar a las señales EMG adquiridas.

Finalmente, se activa un temporizador denominado ClockReReg, creado específicamente para introducir un pequeño retardo temporal en el sistema. Este retraso permite proporcionar al dispositivo el tiempo necesario para aplicar correctamente la nueva configuración de ganancia y estabilizar nuevamente la comunicación BLE antes de reanudar la recepción y representación de datos.

A continuación, se describe el comportamiento que tendrá lugar una vez finalice el temporizador asociado a ClockReReg.

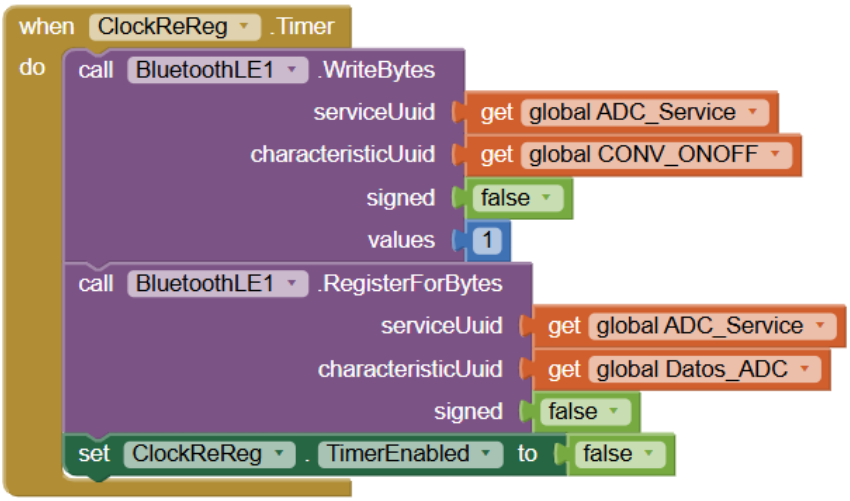


Ilustración 39: Lógica asociada a finalización ClockReReg

Una vez finalizado el tiempo asociado al temporizador ClockReReg, se procede a restaurar el funcionamiento normal del sistema tras la reconfiguración de la ganancia.

En primer lugar, se vuelve a enviar mediante BLE un byte con valor 1 al servicio Conv On/Off, indicando al sensor que debe reactivar las conversiones.

A continuación, se vuelve a habilitar el procesado de los datos recibidos mediante BLE, permitiendo nuevamente la recepción, interpretación y representación de estos.

Finalmente, el temporizador ClockReReg se desactiva estableciendo su estado en false, evitando así nuevas ejecuciones innecesarias del bloque asociado al temporizador hasta que vuelva a producirse un nuevo cambio de ganancia.

3) Canales On/Off

La lógica necesaria para implementar la transmisión y el procesamiento de los datos correspondientes a cada uno de los canales, en función del número de canales activos en cada momento, se divide en varios bloques funcionales que se explicarán detalladamente a continuación.

Esta organización permite adaptar el comportamiento de la aplicación a las distintas configuraciones posibles de adquisición, ya que el sensor puede enviar datos de uno o varios canales de forma simultánea. Por ello, resulta necesario identificar previamente qué canales se encuentran habilitados y distribuir correctamente las muestras recibidas, asegurando que cada valor se almacene y represente en el canal correspondiente.

Ilustración 40: Variable global que contiene el estado de los canales disponibles

En primer lugar, se inicializa una variable global denominada *"Canales_activos",* la cual consiste en una lista formada por cuatro posiciones, todas ellas inicializadas con el valor 0. Cada una de estas posiciones representa el estado de uno de los cuatro canales disponibles en el sistema.

El valor almacenado en cada posición indica si el canal correspondiente se encuentra activo o no. Un valor 1 representa que el canal está habilitado para la adquisición y transmisión de datos, mientras que un valor 0 indica que dicho canal permanece desactivado. De esta manera, la lista *"Canales_activos"* actúa como una estructura de control que permite conocer en todo momento qué canales están siendo utilizados por la aplicación.

A continuación, se desarrolla el procedimiento *"actualizarCanalesActivos",* encargado de actualizar el contenido de esta lista en función de los canales seleccionados por el usuario.

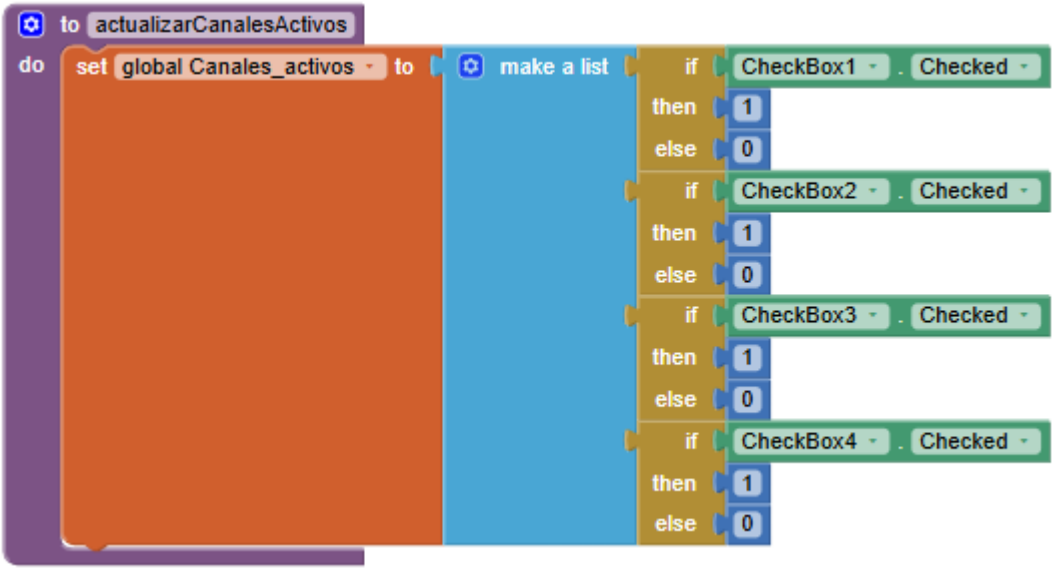


Ilustración 41: Procedimiento actualizarCanalesActivos

La función principal de este procedimiento es actualizar dinámicamente los valores almacenados en cada una de las posiciones de la lista *"Canales_activos"*, permitiendo reflejar en todo momento qué canales se encuentran habilitados para la adquisición de datos.

El procedimiento modifica el valor correspondiente a cada canal de 0 a 1 cuando dicho canal es seleccionado por el usuario mediante su correspondiente CheckBox. Del mismo modo, si un canal deja de estar seleccionado, el valor asociado vuelve a configurarse en 0. Gracias a este mecanismo, la aplicación mantiene constantemente sincronizado el estado interno de los canales activos con la interfaz gráfica mostrada al usuario.

La implementación del procedimiento consiste en reconstruir la lista *"Canales_activos"* utilizando el estado de cada uno de los CheckBox disponibles. Para cada canal, se comprueba el valor de CheckBoxX.Checked. Si el resultado es verdadero (true), indicando que el canal se encuentra seleccionado, la posición correspondiente de la lista se configura con el valor 1. En caso contrario, el valor almacenado permanece en 0.

Por último, se desarrolla el último procedimiento necesario para proporcionar la funcionalidad deseada a la aplicación. Este procedimiento recibe el nombre de

"enviarCanalesOnOff" y su principal objetivo es comunicar al sensor BluEMG qué canales deben permanecer activos para la adquisición y transmisión de datos.

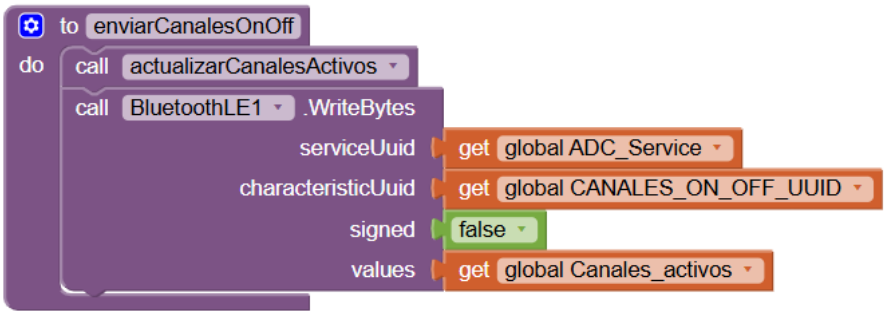


Ilustración 42: Procedimiento enviarCanalesOnOff

Este procedimiento consiste en dos partes:

1) Actualización del estado de los canales activos

   En primer lugar, se realiza una llamada al procedimiento *"actualizarCanalesActivos",* implementado anteriormente. La finalidad de esta llamada es garantizar que los valores almacenados en la lista *"canales_activos"* se encuentren completamente actualizados y correspondan exactamente con los canales seleccionados por el usuario en ese instante. De esta forma, se evita enviar información desactualizada o incoherente al sensor antes de realizar la configuración definitiva.

2) Envío de la configuración al sensor mediante BLE

   Una vez actualizada la lista *"canales_activos",* se realiza una llamada al procedimiento interno de Bluetooth Low Energy denominado *WriteBytes*. Este procedimiento es el encargado de transmitir al sensor la lista completa de estados de los canales utilizando el UUID correspondiente al servicio Canales On/Off. Gracias a ello, el dispositivo recibe información necesaria para habilitar o deshabilitar el muestreo de cada canal según la configuración seleccionada desde la aplicación.

   Además, dentro de este procedimiento se especifica que los valores enviados deben interpretarse como datos sin signo (*unsigned values*). Esta configuración resulta importante para asegurar que los bytes transmitidos sean interpretados correctamente

por el sensor, evitando posibles errores durante el proceso de comunicación entre la aplicación móvil y el dispositivo BluEMG.

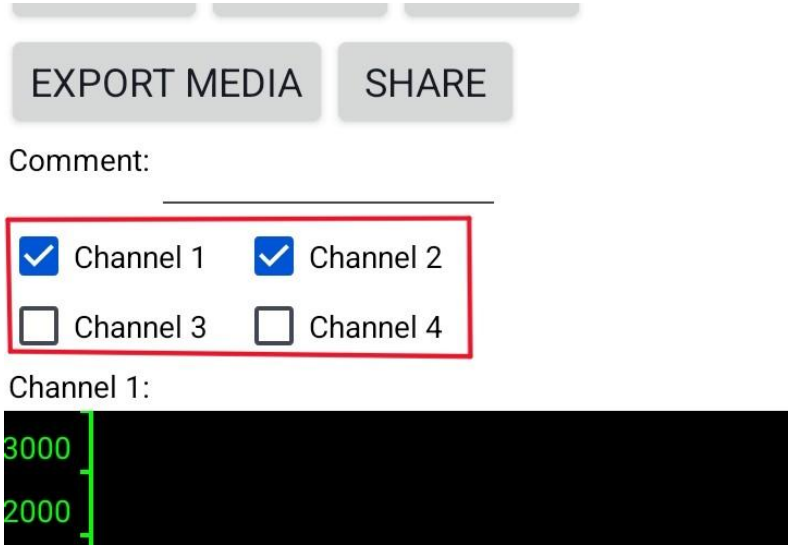


Ilustración 43:Pantalla principal, CheckBoxes

Por último, aunque este apartado no corresponda directamente al desarrollo de ningún procedimiento específico relacionado con los servicios descritos anteriormente, resulta interesante mostrar el punto exacto de la implementación donde se utiliza y se configura cada uno de dichos servicios dentro de la aplicación.

De esta manera, es posible comprender con mayor claridad cómo se integran los distintos servicios BLE del sensor BluEMG dentro de la lógica general del sistema. Además, esta representación permite identificar de forma más estructurada el flujo completo de funcionamiento de la aplicación, desde la selección de parámetros por parte del usuario hasta el envío de comandos y la recepción de datos desde el sensor.

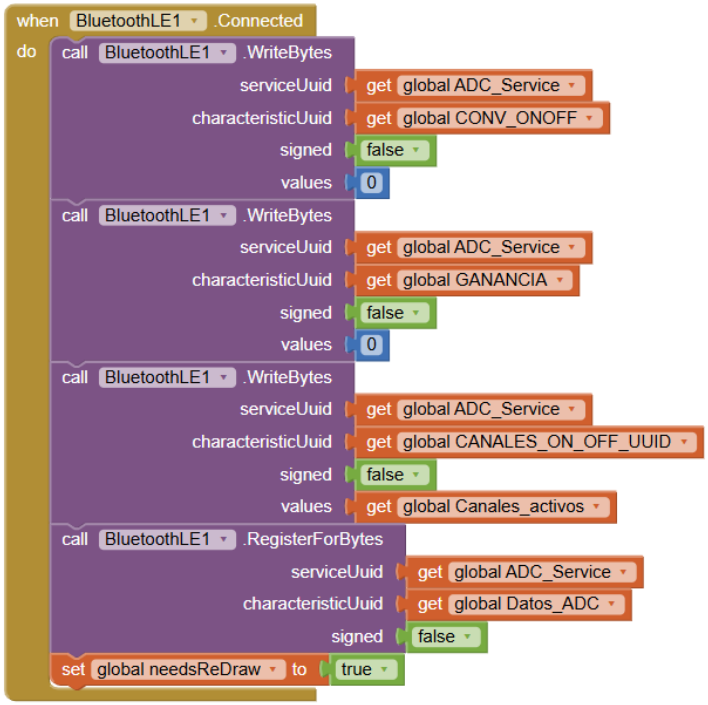


Ilustración 44: Lógica asociada al establecimiento de la conexión BLE

En primer lugar, al abrir por primera vez la pantalla principal de la aplicación y una vez establecida correctamente la conexión con el dispositivo mediante Bluetooth Low Energy, se configuran automáticamente los distintos servicios descritos anteriormente utilizando sus valores por defecto. Esta inicialización resulta fundamental para garantizar que cada vez que se inicia la aplicación, el sistema parte siempre de una configuración conocida y controlada, evitando posibles inconsistencias derivadas de configuraciones previas almacenadas en el sensor.

El primer servicio que se configura es el correspondiente al Conversor On/Off . Inicialmente, este servicio se establece con el valor 0, lo que implica que las conversiones analógico-digitales permanecen desactivadas. Esta decisión permite reducir el consumo energético del dispositivo mientras no exista ningún canal seleccionado para la adquisición de datos. Posteriormente, cuando el usuario active alguno de los canales mediante los

correspondientes CheckBox, el valor del servicio pasará a 1, habilitando así el inicio de las conversiones y la transmisión de señales EMG.

El siguiente servicio configurado es el relacionado con la ganancia de las señales adquiridas. Aunque el sensor BluEMG dispone por defecto de una ganancia inicial de valor 1, esta configuración se vuelve a enviar explícitamente desde la aplicación como medida de seguridad. De esta manera, se garantiza que el sistema opere siempre con una configuración conocida incluso en situaciones donde el sensor pudiera haber mantenido valores diferentes procedentes de sesiones anteriores. Para realizar esta operación, se envía el valor correspondiente (0) mediante el procedimiento *WriteBytes* proporcionado por el componente Bluetooth Low Energy.

El tercer y último servicio configurado corresponde al servicio Canales On/Off. Para realizar esta configuración, se envía mediante el procedimiento WriteBytes la lista *"canales_activos",* descrita anteriormente. Como ya se ha explicado, esta lista se inicializa por defecto con todos sus valores configurados a 0, indicando que ninguno de los cuatro canales disponibles se encuentra habilitado inicialmente.

A continuación, se ejecuta el procedimiento *RegisterForBytes* de BLE. Este procedimiento permite registrar la aplicación para recibir notificaciones cuando el sensor envíe nuevos bytes de datos. Gracias a esta suscripción, y debido a que la característica utilizada es del tipo *notify*, la aplicación permanece constantemente a la espera de nuevos paquetes de información transmitidos por el dispositivo, pudiendo así recibir, procesar y representar correctamente las señales EMG en tiempo real.

Por último, pero no menos importante, se configura la variable de control *"needsReDraw"* con el valor true. Esta variable actúa como un *flag* encargado de indicar que la representación gráfica de las señales está habilitada. De esta forma, cuando el sensor comience a transmitir datos, la aplicación sabrá que puede dibujar las señales recibidas en el Canvas correspondiente a cada canal activo.

En la imagen mostrada a continuación pueden observarse los procedimientos que se ejecutan automáticamente cada vez que se modifica el estado de alguno de los CheckBox asociados a los canales de adquisición. Estos procedimientos se activan tanto cuando un nuevo canal es seleccionado como cuando uno de los canales previamente activos es deseleccionado.

Lo primero que ocurre al modificarse el estado de un CheckBox es la llamada al procedimiento *"actualizarCanalesActivos"*. Mediante esta llamada, se actualiza el contenido de la lista *"canales_activos"*, modificando el valor correspondiente al canal afectado para reflejar correctamente si este se encuentra activo (1) o inactivo (0).

Seguidamente, se llama al procedimiento *"UpdateConv"*, cuya función principal es gestionar el estado del servicio Conv On/Off del sensor. Este procedimiento se encarga de comprobar si existe al menos un canal seleccionado y, en caso afirmativo, enviar el valor 1 al byte de conversión para habilitar la adquisición de datos.

A continuación, se activa un temporizador asociado al componente Clock3. La finalidad de este temporizador es introducir un pequeño retardo temporal en el flujo de ejecución, permitiendo que el sensor disponga de tiempo suficiente para procesar correctamente las escrituras realizadas sobre los distintos servicios BLE sin llegar a saturarse o producir conflictos en la comunicación. Para esta función se ha establecido un período de 1 segundo.

Por último, se implementa un bloque condicional if…then en el que se comprueba si el CheckBox correspondiente al canal modificado se encuentra en estado false, es decir, deseleccionado. En caso de cumplirse esta condición, se ejecuta el procedimiento encargado de borrar el contenido gráfico del Canvas asociado a dicho canal. De esta manera, las señales correspondientes a canales desactivados desaparecen inmediatamente de la interfaz gráfica, manteniendo la representación visual sincronizada con la configuración actual del sistema.

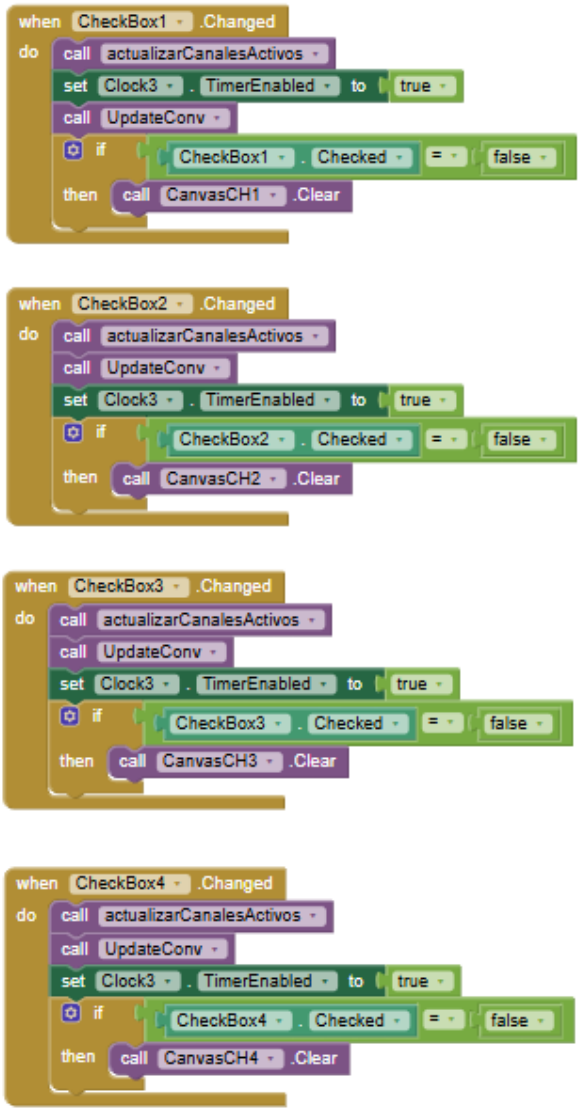


Ilustración 45: Lógica asociada al cambio de estado de los *CheckBoxes* de los 4 canales disponibles

Exportación y compartición de datos

La realización de esta parte del proyecto se basa en la implementación de cuatro botones principales, cada uno de ellos asociado a una función específica dentro del proceso de compartición de los datos registrados por la aplicación. Estos botones son: *"Record"* (Grabar), encargado de iniciar la adquisición y almacenamiento de los datos; *"Stop"*

(Detener), cuya función es finalizar el proceso de grabación en curso; *"Export Media"* (Exportar datos), utilizado para generar y almacenar un archivo con la información registrada; y, finalmente, *"Share"* (Compartir), que permite enviar o compartir los datos exportados mediante diferentes aplicaciones o servicios disponibles en el dispositivo.

Cada uno de estos botones desempeña un papel fundamental dentro del flujo de trabajo diseñado para la captura, almacenamiento y posterior distribución de la información obtenida. De esta manera, el usuario puede controlar de forma sencilla e intuitiva todas las etapas del proceso, desde el inicio de la grabación hasta la exportación y compartición final de los datos.

Antes de implementar la lógica asociada a cada una de estas funcionalidades, es necesario definir una serie de variables globales que serán utilizadas a lo largo de todo el programa. Estas variables permiten almacenar información relevante, como el estado de la grabación, los datos capturados, los nombres de los archivos generados o las rutas de almacenamiento, facilitando así la comunicación y el intercambio de información entre los distintos bloques funcionales de la aplicación.

Ilustración 46: Definición variables globales necesarias para la exportación CSV

Seguidamente, se desarrolla la lógica asociada al botón *"Record"*, situado en la parte superior izquierda de la pantalla principal de la aplicación. Este botón constituye el punto de inicio del proceso de adquisición de datos, ya que su activación permite comenzar la grabación y el almacenamiento de la información procedente de los distintos canales seleccionados por el usuario.

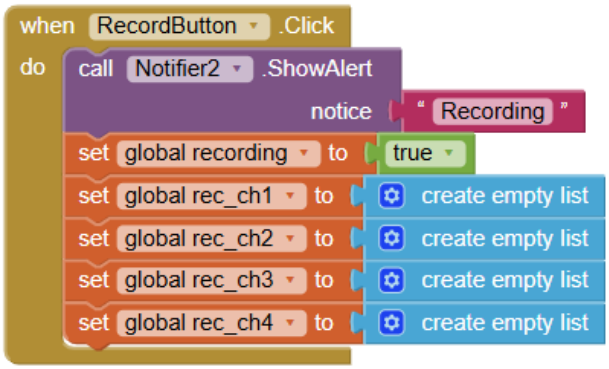


Ilustración 47: Lógica asociada al botón "Record"

La lógica implementada para el botón *"Record"* se basa en una serie de pasos secuenciales cuyo objetivo es preparar la aplicación para el inicio de la grabación y el almacenamiento de los datos recibidos. El funcionamiento de esta lógica puede resumirse en los siguientes puntos:

1) Notificación visual al usuario

   En primer lugar, se muestra una alerta en pantalla con el mensaje *"Recording"*. Esta notificación tiene como finalidad informar al usuario de que, a partir de ese momento, todos los datos que aparezcan en pantalla y sean recibidos por la aplicación comenzarán a ser registrados y almacenados.

2) Activación de la variable de control de grabación

   A continuación, se modifica el valor de una variable global de tipo booleano, estableciéndola en true. Esta variable actúa como un *flag* o indicador de estado dentro de la aplicación, permitiendo que el bloque encargado de recibir los datos detecte que la grabación se encuentra activa. Como consecuencia, los datos recibidos pasan a

almacenarse en un buffer o estructura de memoria independiente, destinado específicamente a conservar la información que posteriormente podrá ser exportada.

3) Inicialización y limpieza de las estructuras de almacenamiento

   Posteriormente, se vacían las listas utilizadas para almacenar los datos grabados. Este paso resulta fundamental para garantizar que cada nueva sesión de grabación comience desde un estado limpio, evitando tanto la sobreescritura de información como la inclusión de datos pertenecientes a grabaciones anteriores.

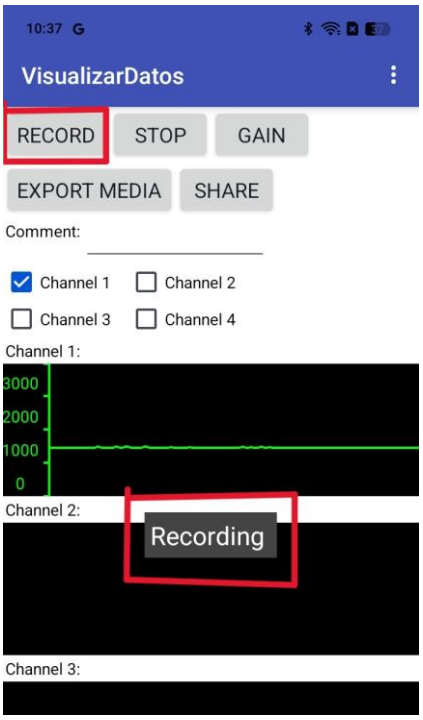


Ilustración 48: Pantalla principal, botón Record

Seguidamente, se presenta la lógica asociada al botón *"Stop"*, cuya función principal consiste en detener el proceso de grabación y finalizar el almacenamiento de los datos registrados durante la sesión.

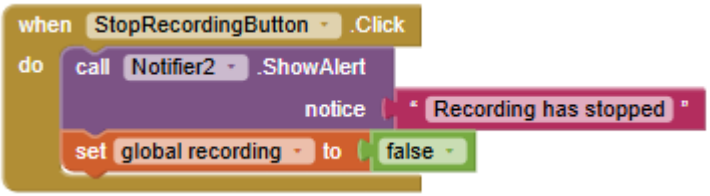


Ilustración 49: Lógica asociada al botón "Stop"

En este bloque de código se implementa la lógica asociada al botón *"Stop",* situado a la derecha del botón *"Record"* en la interfaz principal de la aplicación. La finalidad de este botón es permitir al usuario detener el proceso de grabación de datos de manera inmediata y controlada.

Al pulsar dicho botón, se muestra en pantalla una alerta informativa destinada a notificar al usuario de que la grabación ha finalizado correctamente. De esta forma, se indica que, a partir de ese instante, los nuevos datos que continúen apareciendo o siendo recibidos por la aplicación dejarán de almacenarse en memoria y, por tanto, no formarán parte del archivo exportable generado posteriormente.

Tras mostrar la notificación correspondiente, la lógica del programa modifica el valor de la variable global booleana *"Recording",* estableciéndola en false. Esta variable actúa como un indicador de estado o *flag* dentro del sistema, permitiendo que el bloque encargado de recibir y procesar los datos detecte que la grabación ya no se encuentra activa. Como consecuencia, los datos recibidos después de pulsar el botón *"Stop"* se siguen mostrando en la interfaz de usuario, pero dejan de almacenarse en las listas o buffers destinados a la exportación.

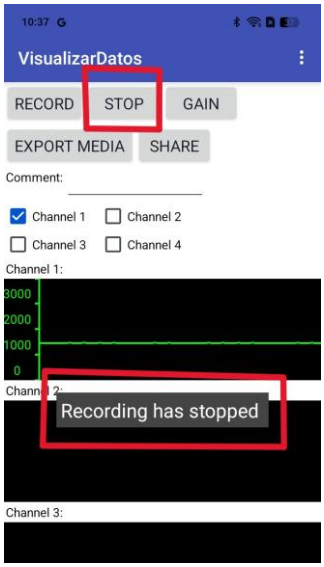


Ilustración 50: Pantalla principal, botón Stop

El siguiente elemento funcional que se va a describir es el botón *"Export media"*, ubicado justo debajo del botón *"Record"*, cuya función principal es generar y exportar un archivo con todos los datos previamente grabados durante la sesión.

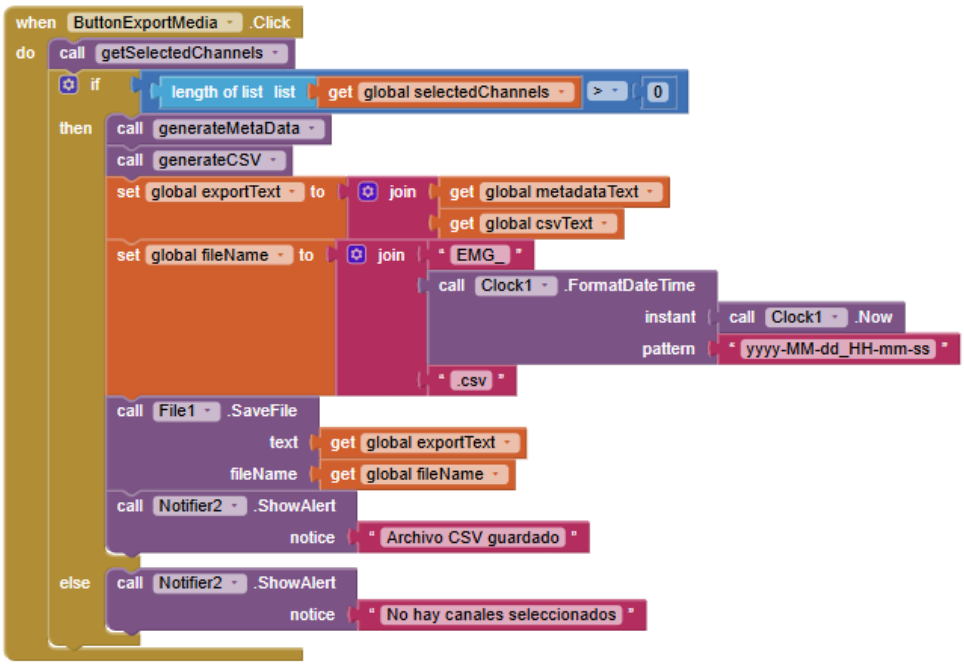


Ilustración 51: Lógica asociada al botón "Export Media"

La lógica descrita a continuación se ejecuta al pulsar el botón *"Export media"*, acción que debe realizarse una vez detenida la grabación mediante el botón *"Stop"*.

En primer lugar, se llama al procedimiento *"getSelectedChannels"*, cuyo funcionamiento se explicará con mayor detalle más adelante. De forma resumida, este procedimiento se encarga de identificar qué canales se encuentran seleccionados, con el objetivo de conocer el origen de los datos registrados y poder generar correctamente tanto los metadatos como el archivo CSV final.

A continuación, se incluye una estructura condicional if…then…else para comprobar que el número de canal seleccionado sea distinto de cero. Esta verificación es necesaria para evitar que se ejecute el proceso de exportación cuando no existe ningún canal activo, ya que en ese caso no habría datos válidos que guardar y el flujo de trabajo no sería correcto. Si al menos un canal ha sido seleccionado, se ejecutan los procedimientos *"generateMetadata"* y *"generateCSV"*, encargados de generar la información descriptiva del registro y el contenido principal del archivo CSV, respectivamente.

En caso de que el usuario pulse el botón *"Export media"* sin haber seleccionado ningún canal, la aplicación mostrará una alerta con el mensaje *"No channels selected"*, informando de que no es posible continuar con la exportación.

Una vez generados los metadatos y el contenido CV, se configura la variable global *"exportText"*, que contiene el texto final que será exportado. Esta variable se obtiene mediante la unión de los metadatos generados por *"generateMetadata"* y el texto CSV generado por *"generateCSV"*. Además, se define la variable global *"fileName"*, encargada de asignar un nombre específico al archivo creado. Dicho nombre se construye uniendo el prefijo *"EMG_"*, la fecha y hora actuales obtenidas a partir del reloj principal con el formato yyyy-MM-dd_HH-mm-ss, y la extensión *".csv"*. De esta forma, cada archivo generado cuenta con un nombre único y fácilmente identificable.

Finalmente, se llama al procedimiento encargado de guardar el archivo, utilizando como parámetros el texto final generado y el nombre asignado al fichero. Una vez completado

el proceso, se muestra una nueva alerta en pantalla con el mensaje *"CSV file saved"*, indicando al usuario que la exportación se ha realizado correctamente.

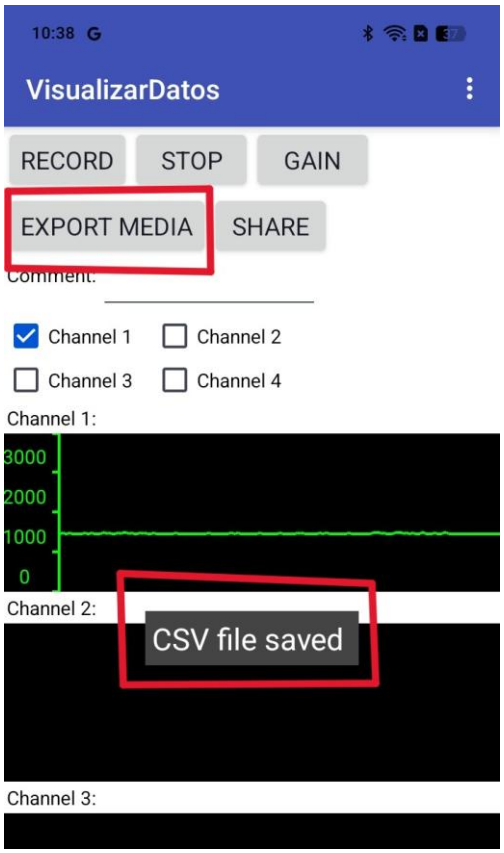


Ilustración 52: Pantalla principal, botón Export Media

A continuación, se explicará detalladamente la lógica de los tres procedimientos mencionados anteriormente: *"getSelectedChannels"*, *"generateMetaData"* y *"generateCSV"*.

getSelectedChannels:

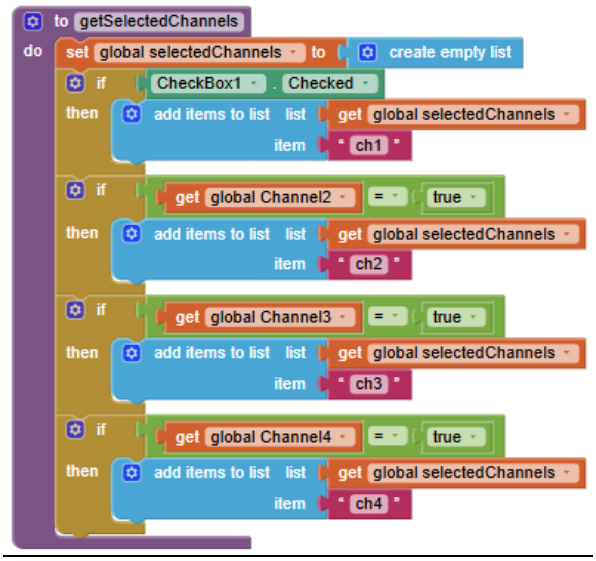


Ilustración 53: Procedimiento getSelectedChannels

El procedimiento comienza inicializando la lista *"selectedChannels"* como una lista vacía, con el objetivo de garantizar que no exista información residual procedente de ejecuciones anteriores del programa. Este paso resulta fundamental para asegurar que únicamente se almacenan los canales seleccionados en el momento actual y evitar posibles errores o duplicidades durante el proceso de exportación.

Posteriormente, se implementa una estructura condicional if…then para cada uno de los canales disponibles en la aplicación. La función de estos bloques es comprobar el estado del correspondiente CheckBox asociado a cada canal. Si el valor del CheckBox es true, lo que indica que el canal ha sido seleccionado por el usuario, se añade a la lista *"selectedChannels"* una cadena de texto con el formato "chX", donde X representa el número identificativo del canal seleccionado.

De esta manera, la lista *"selectedChannels"* actúa como un registro dinámico de los canales activos durante la sesión de grabación y exportación. Esta información resulta posteriormente esencial para generar correctamente tanto los metadatos descriptivos

como la estructura del archivo CSV, permitiendo identificar con precisión el origen de cada una de las señales almacenadas.

generateMetaData:

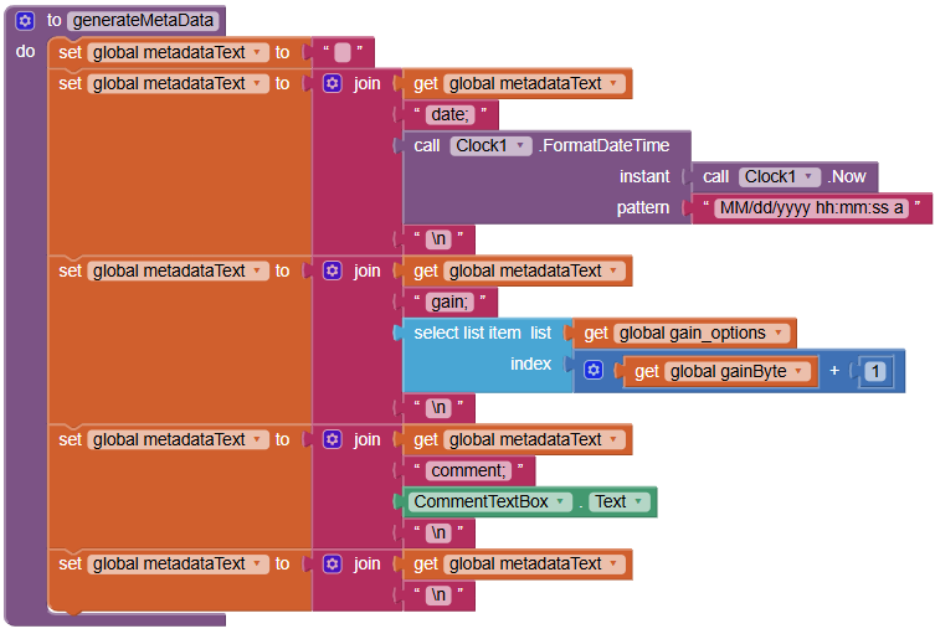


Ilustración 54: Procedimiento generateMetadata

En primer lugar, el procedimiento se encarga de inicializar la variable *"metadataText"* como una cadena de texto vacía, garantizando así que no existe información residual procedente de ejecuciones anteriores.

Una vez confirmada la inicialización de la variable, se añade a *"metadataText"* una primera línea de información correspondiente a la fecha y hora de creación del archivo. Para ello, se concatena el contenido actual de la variable con el texto *"date:"*, seguido de una llamada al reloj principal de la aplicación (Clock1), encargado de devolver la fecha y hora exactas del instante en el que se realiza la exportación. Finalmente, se añade un salto de línea (\n) con el objetivo de organizar correctamente la estructura del fichero y facilitar su lectura posterior.

Seguidamente, se lleva a cabo un proceso similar para añadir información relacionada con la ganancia aplicada a las señales registradas. En este caso, se concatena el contenido ya existente en *"metadataText"* con la etiqueta *"gain:"*, seguida del valor de ganancia seleccionado. Dicho valor se obtiene accediendo a la posición *gainByte +1* de la lista global *"gainOptions"*, que contiene todas las configuraciones de ganancia disponibles. La variable *"gainByte"* almacena el índice correspondiente a la opción seleccionada, y es necesario sumarle una unidad debido a que MIT App Inventor indexa las listas comenzando desde 1 en lugar de desde 0.

Posteriormente, se añade al contenido de *"metadataText"* una nueva línea que incorpora el comentario introducido por el usuario en la casilla de texto disponible en la pantalla principal de la aplicación, identificada mediante la etiqueta "Comment", situada justo debajo de los botones *"Export media"* y *"Share"*. Esta funcionalidad permite asociar anotaciones o información adicional a cada sesión de grabación, proporcionando un contexto más completo sobre los datos registrados.

Por último, se añade un salto de línea adicional al final de *"metadataText"*. La finalidad de este espacio es separar visualmente los metadatos generados de los datos reales contenidos en el archivo CSV, mejorando así la organización y legibilidad del documento exportado.

generateCSV:

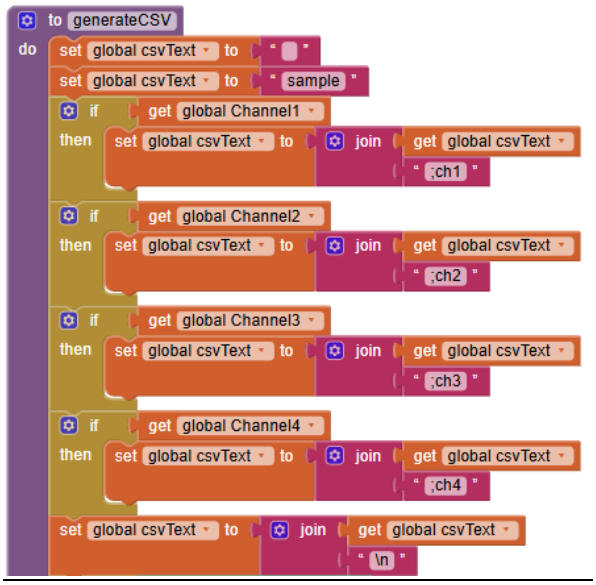


Ilustración 55: Procedimiento generateCSV, parte 1

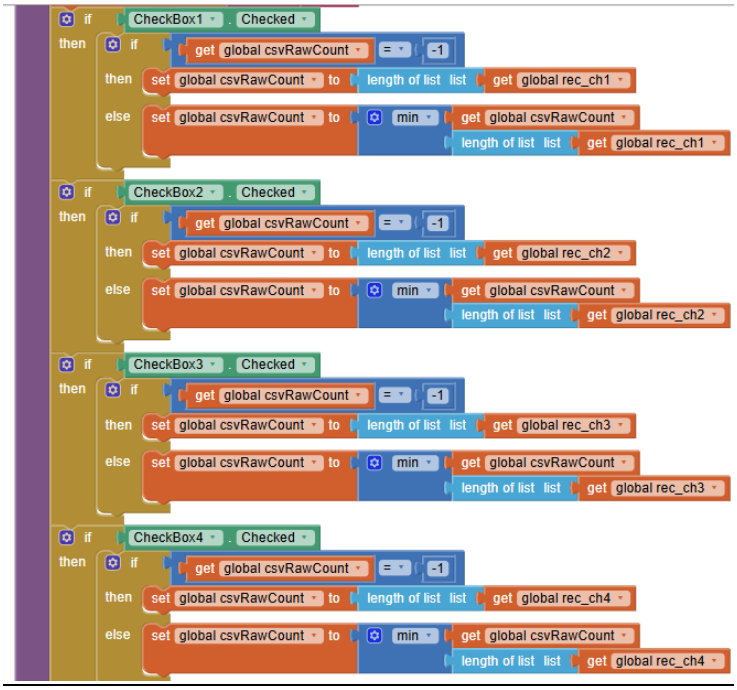


Ilustración 56:Procedimiento generateCSV, parte 2

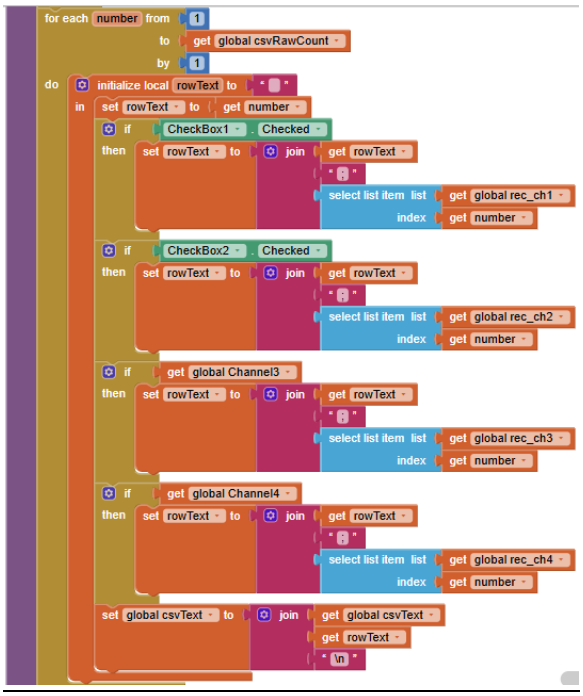


Ilustración 57:Procedimiento generateCSV, parte 3

En este último procedimiento se describe detalladamente el proceso mediante el cual se genera el archivo de CSV, que contendrá todos los datos registrados durante la sesión de grabación.

En primer lugar, se inicializa la variable global de tipo texto *"csvText"* como una cadena vacía. Esta variable será la encargada de almacenar progresivamente todo el contenido del fichero CSV. Una vez inicializada, se añade el texto *"sample",* que actuará como cabecera de la primera columna del archivo y servirá para identificar el índice o número de muestra correspondiente a cada dato registrado.

Seguidamente, se implementan cuatro bloques condicionales if…then , uno para cada uno de los posibles canales disponibles en la aplicación. En cada bloque se comprueba si el canal correspondiente se encuentra seleccionado por el usuario. En caso afirmativo, se concatena a la variable *"csvText"* el identificador del canal, siguiendo el formato *"chX",* donde X representa el número del canal. De esta manera, se generan dinámicamente las cabeceras de las columnas del archivo CSV únicamente para los

canales de activos. Finalmente, se añade un salto de línea para separar la cabecera del contenido de los datos.

A continuación, se implementa una nueva serie de cuatro bloques if…then , nuevamente destinados a comprobar qué canales han sido seleccionados. Sin embargo, en esta ocasión el objetivo es calcular el número de filas que tendrá el archivo CSV. Dentro de cada condición then, se incorporará un nuevo bloque if…then…else. En la condición if se comprueba si el valor de la variable *"csvRawCount"* es igual a -1. Este valor inicial actúa como indicador de que todavía no se ha determinado el número de muestras disponibles.

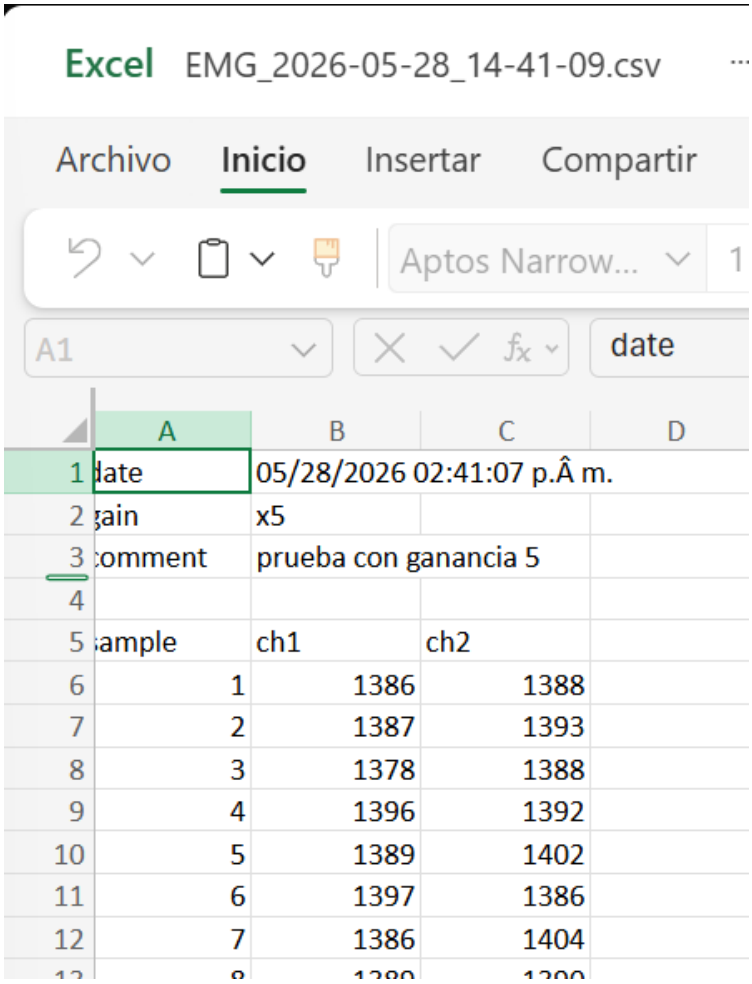
Si la condición se cumple, se asigna a *"csvRawCount"* la longitud de la lista *"rec_chX",* correspondiente al canal analizado en ese momento. Estas listas (*"rec_ch1","rec_ch2",*etc.) son las estructuras encargadas de almacenar los datos grabados de cada canal. En caso de que *"csvRawCount"* ya contenga un valor válido, se actualiza tomando el menor valor entre el contenido actual de *"csvRawCount"* y la longitud de la lista del canal correspondiente. Este procedimiento garantiza que el archivo CSV únicamente genere filas hasta el número mínimo de muestras disponibles entre todos los canales seleccionados, evitando posibles errores por diferencias de longitud entre listas.

Posteriormente, se crea un bucle for each number que recorre todos los índices desde 0 hasta el valor de *"csvRawCount"*. Este bucle es el encargado de generar cada una de las filas del archivo CSV. Dentro de él, se inicializa una nueva variable local denominada *"rowText"* como una cadena vacía. A continuación, se asigna a esta variable el valor correspondiente al índice actual del bucle, representando así el número de muestra asociado a la fila.

Seguidamente, se vuelven a implementar cuatro bloques if…then, uno para cada canal posible. En cada caso, se comprueba si el canal correspondiente está seleccionado. Si la condición se cumple, se concatena al contenido de *"rowText"* un punto y coma (;), utilizado como separador de columnas dentro del archivo CSV, seguido del valor

almacenado en la posición correspondiente al índice actual del bucle dentro de la lista rec_chX.

Una vez completada la construcción de la fila, se concatena el contenido de *"rowText"* a la variable global *"csvText"*, añadiendo así una nueva línea de datos al archivo CSV final.



	A	B	C	D
1	date	05/28/2026 02:41:07 p.Â m.		
2	gain	x5		
3	comment	prueba con ganancia 5		
4				
5	sample	ch1	ch2	
6	1	1386	1388	
7	2	1387	1393	
8	3	1378	1388	
9	4	1396	1392	
10	5	1389	1402	
11	6	1397	1386	
12	7	1386	1404	

Ilustración 58: Ejemplo de archivo CSV generado

En la última parte de este bloque de código se implementa la lógica asociada al botón *"Share"*, cuya función es permitir al usuario compartir el archivo generado después de haber realizado correctamente el flujo completo de grabación, detención y exportación mediante los botones *"Record"*, *"Stop"* y *"Export media"*, respectivamente.

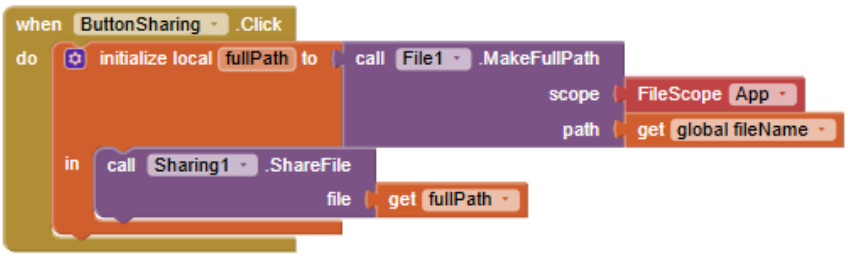


Ilustración 59: Lógica asociada al botón "Share"

En este último bloque de código se implementa la lógica necesaria para compartir el archivo CSV previamente generado y almacenado por la aplicación.

En primer lugar, se inicializa una nueva variable local denominada *"fullPath"*. Esta variable obtiene su valor mediante una llamada al procedimiento interno MakeFullPath del componente File. La función de este procedimiento es construir la ruta completa del archivo que se desea compartir, utilizando para ello dos parámetros principales: el scope y el path.

El parámetro scope se configura con el valor FileScope.App, indicando que el archivo se encuentra almacenado dentro del espacio de almacenamiento privado asociado a la propia aplicación. Por otro lado, el parámetro path toma como valor el contenido de la variable global *"fileName"*. La cual contiene el nombre específico generado previamente para el archivo CSV exportado.

Gracias a este procedimiento, la variable *"fullPath"* almacena la ruta completa y válida del fichero dentro del sistema de almacenamiento del dispositivo, permitiendo que posteriormente pueda ser localizado y utilizado correctamente por otras funciones de la aplicación.

Finalmente, se realiza una llamada al procedimiento *"ShareFile"*, utilizando como parámetro el archivo identificado mediante *"fullPath"*. Este procedimiento es el encargado de abrir el menú de compartición del sistema operativo, permitiendo al usuario enviar el archivo CSV generado a través de diferentes aplicaciones o servicios

disponibles en el dispositivo, como correo electrónico, aplicaciones de mensajería o servicios de almacenamiento en la nube.

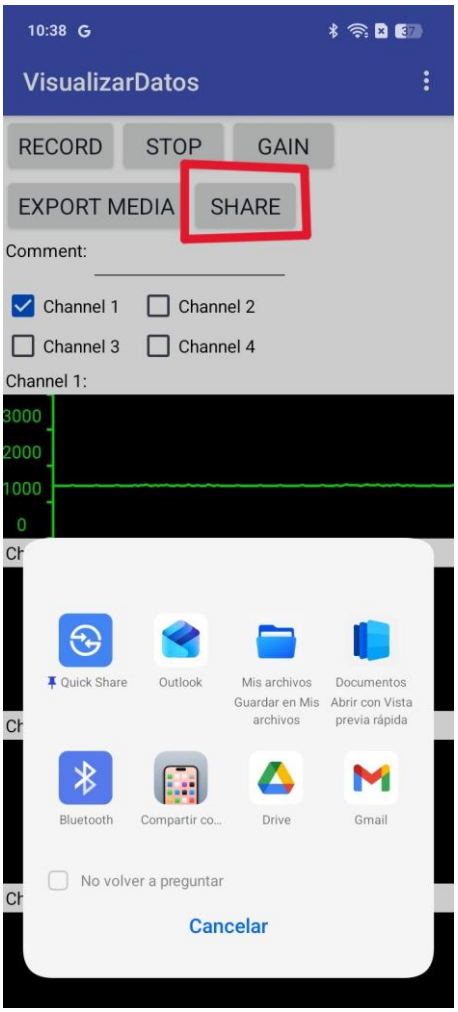


Ilustración 60: Pantalla principal, botón Share

4.1.2.3 Muestra de las características principales de BluEMG

Aunque en la primera pantalla de la aplicación ya se muestran algunos datos característicos del sensor, se considera conveniente incorporarlos también en la pantalla principal. De este modo, el usuario puede supervisar en todo momento determinados parámetros relevantes durante el uso del sistema, como el porcentaje de batería restante o la potencia de carga y descarga del dispositivo.

Para ello, se decide reutilizar parte de la lógica ya implementada en la primera pantalla, adaptándola y trasladándola a la pantalla principal de la aplicación. Con el fin de permitir el intercambio de información entre ambas pantallas, se utiliza una base de datos local denominada *TinyDB*. En ella se almacenan los datos obtenidos en la primera pantalla, de forma que posteriormente puedan recuperarse desde la pantalla principal y mostrarse correctamente al usuario.

A continuación, se describen de forma detallada los distintos bloques utilizados para implementar esta funcionalidad.

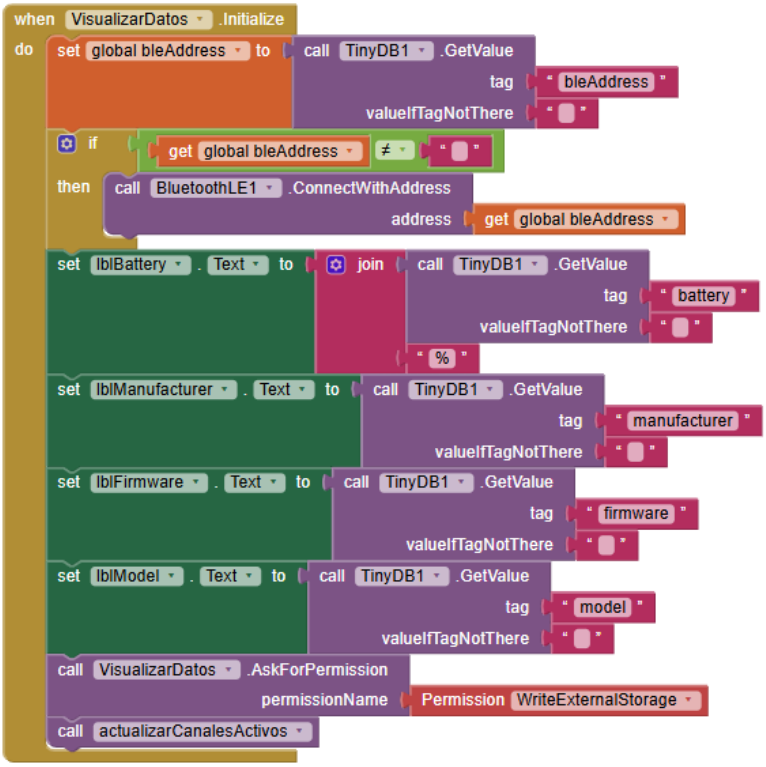


Ilustración 61: Lógica asociada a la visualización de características en pantalla principal

El primer y principal bloque de esta parte de la pantalla principal sigue el siguiente transcurso de operación:

1) Recuperación de la dirección BLE almacenada

   Se configura la variable global *"bleAddress"*. Con el valor almacenado bajo el tag *"bleAddress"* dentro de la base de datos local *TinyDB*. De esta manera, la aplicación recupera automáticamente la dirección MAC del dispositivo utilizado anteriormente, evitando que el usuario tenga que volver a seleccionarlo manualmente cada vez que inicia la aplicación.

2) Reconexión automática mediante Bluetooth Low Energy

   A continuación, se implementa un bloque condicional if…then, en el que se comprueba si el contenido de la variable *"bleAddress"* no está vacío. En caso de existir una dirección válida, se llama el procedimiento interno de Bluetooth Low Energy, denominado ConnectWithAddress. Este procedimiento permite establecer la conexión BLE utilizando la dirección especificada en el campo address, que en este caso corresponde al valor almacenado en la variable global *"bleAddress"*. Gracias a esta lógica, la aplicación puede reconectarse automáticamente al sensor sin necesidad de intervención manual por parte del usuario.

3) Actualización del porcentaje de batería

   Posteriormente, se configura el valor mostrado en el label Battery siguiendo el mismo procedimiento utilizado para recuperar la dirección MAC. Para ello, se extrae desde TinyDB el valor asociado al tag battery. Con el objetivo de facilitar la interpretación de este dato por parte del usuario, el valor obtenido se concatena con el símbolo %, indicando así el porcentaje de batería restante del dispositivo.

4) Carga de información adicional del sensor

   Seguidamente, se repite el mismo proceso para los labels manufacturer, firmware y model. De esta manera, la aplicación recupera y muestra automáticamente información relevante sobre el fabricante, la versión del firmware y el modelo del sensor conectado.

5) Solicitud de permisos de almacenamiento

   A continuación, se realiza una llamada al procedimiento AskForPermission utilizando el parámetro permissionName: Permission WriteExternalStorage. Esta

instrucción permite solicitar al sistema operativo permisos de escritura sobre el almacenamiento del dispositivo. Dicho permiso resulta necesario para poder crear, guardar y compartir correctamente los archivos CSV generados por la aplicación fuera del entorno interno de esta.

6) Actualización del estado de los canales activos

   Por último, se actualiza el valor correspondiente a los canales activos con el objetivo de sincronizar el estado de la interfaz gráfica con la configuración actual del sistema. Esto permite que la aplicación conozca en todo momento qué canales se encuentran habilitados y garantiza que tanto la visualización de las señales como los procesos de grabación y exportación trabajen únicamente con los canales seleccionados por el usuario.

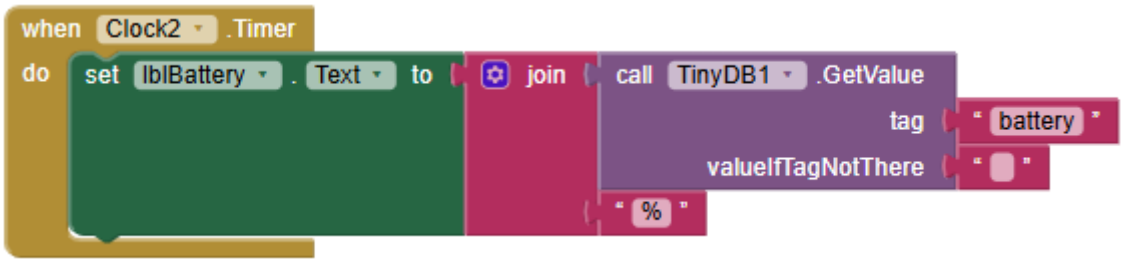


Ilustración 62: Lógica asociada a la finalización de Clock2 en pantalla principal

El último bloque de esta parte de la aplicación consiste en un segundo temporizador, denominado Clock2, configurado con un período de 1000 ms, equivalente a un segundo. La principal función de este temporizador es actualizar periódicamente el valor del porcentaje de batería restante del sensor mientras la aplicación permanece en funcionamiento.

Gracias a esta actualización automática, la información mostrada en pantalla se mantiene constantemente sincronizada con el estado real del dispositivo, permitiendo al usuario supervisar en tiempo real el nivel de batería disponible durante toda la sesión de uso. Este aspecto resulta especialmente importante en dispositivos inalámbricos alimentados por

batería, ya que permite detectar con antelación niveles bajos de carga y evitar posibles interrupciones en la adquisición de señales.

Cada vez que el temporizador alcanza el intervalo establecido, se ejecuta nuevamente la lógica encargada de recuperar el valor actualizado de la batería desde la base de datos TinyDB y mostrarlo en la interfaz gráfica de la aplicación.

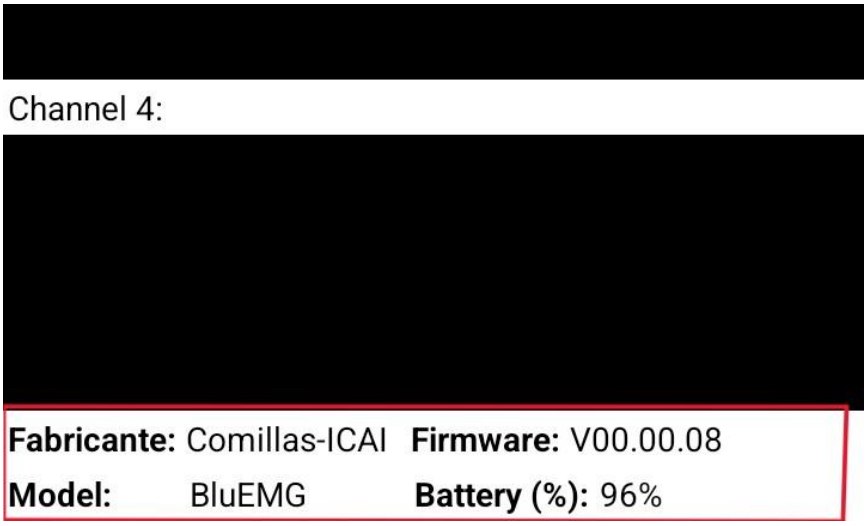


Ilustración 63: Pantalla principal, datos BluEMG

Capítulo 5. ANÁLISIS DE RESULTADOS

Debido a que la lógica correspondiente a cada una de las pantallas de la aplicación ha sido diseñada e implementada de forma independiente, resulta conveniente analizar y presentar los resultados obtenidos de manera separada para cada una de ellas. Esto permite realizar una evaluación más clara y organizada del funcionamiento de cada apartado de la aplicación, así como comprender con mayor detalle las funcionalidades desarrolladas en cada pantalla.

En primer lugar, se presentan los resultados obtenidos durante el desarrollo de la primera pantalla de la aplicación. Esta pantalla constituye una de las partes fundamentales del proyecto, ya que es la encargada de gestionar toda la lógica relacionada con la conexión mediante Bluetooth Low Energy (BLE) entre el dispositivo móvil y el sensor BluEMG. Además, desde esta misma interfaz también se muestran en tiempo real diferentes características e información relevante del sensor, permitiendo al usuario conocer el estado del dispositivo durante su funcionamiento.

Al iniciar la aplicación, la primera interfaz que aparece en pantalla es la que se muestra en la siguiente imagen:

Ilustración 64: 1º pantalla, con sensor desconectado

Como se puede observar en la interfaz mostrada por pantalla, en la parte superior y centrado se encuentra el botón denominado "Conectar", cuya función es iniciar el proceso de búsqueda y conexión con dispositivos Bluetooth. Low Energy, cercanos.

Justo debajo de ese botón aparece una zona de color gris, destinada a mostrar dinámicamente la lista de dispositivos BLE detectados por la aplicación durante el escaneo. Esta sección permanecerá vacía hasta que se inicie la búsqueda de dispositivos cercanos y se detecten sensores compatibles disponibles para la conexión.

A continuación, bajo dicha zona, se muestra el estado actual de la conexión Bluetooth. En el caso representado en la imagen, la aplicación indica que no existe ningún dispositivo

conectado mediante Bluetooth Low Energy, ya que todavía no se ha establecido comunicación con ningún sensor.

Finalmente, en la parte inferior de la pantalla se encuentra el espacio reservado para la visualización de los valores correspondientes a las distintas características del sensor. En este momento, dichos campos aparecen vacíos, puesto que la información únicamente se obtiene y muestra una vez que la conexión con el dispositivo Bluetooth se ha realizado correctamente.

Una vez el usuario pulsa sobre el botón conectar, se ejecuta automáticamente la función encargada de realizar la búsqueda de dispositivos Bluetooth Low Energy cercanos disponibles para la conexión. Durante este proceso, la aplicación inicia el escaneo de dispositivos BLE presentes en el entorno y actualiza la interfaz gráfica para reflejar el estado de búsqueda activo.

Mientras se lleva a cabo dicho escaneo, la interfaz de usuario adopta el aspecto que se muestra en la siguiente imagen:

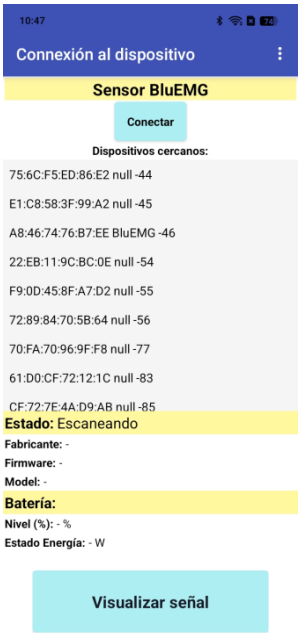


Ilustración 65: 1º pantalla, en proceso de escaneo

Llegados a este punto del funcionamiento de la aplicación, puede observarse cómo la zona gris de la interfaz, que anteriormente permanecía vacía, ahora muestra una lista con los distintos dispositivos Bluetooth Low Energy detectados durante el proceso de escaneo. Esta lista se genera dinámicamente a medida que la aplicación identifica dispositivos BLE cercanos disponibles para establecer una posible conexión.

Entre los dispositivos detectados, se puede apreciar que el tercer elemento de la lista corresponde al sensor de interés utilizado en el proyecto, denominado BluEMG. Esto confirma que el proceso de búsqueda se ha realizado correctamente y que la aplicación es capaz de detectar el sensor biomédico dentro del entorno cercano.

Asimismo, también se observa una modificación en la etiqueta correspondiente al estado de la aplicación. Mientras que inicialmente se indicaba que no existía ninguna conexión activa, ahora el estado mostrado es “Escaneando”, reflejando que el proceso de búsqueda de dispositivos BLE ha finalizado satisfactoriamente.

Por último, puede comprobarse que todavía no se muestran en pantalla los valores asociados a las características del sensor. Eso se debe a que, aunque el dispositivo BluEMG ya ha sido detectado, todavía no se ha establecido la conexión Bluetooth Low Energy con él, por lo que la aplicación aún no puede acceder ni recibir la información proporcionada por el sensor.

Una vez el usuario selecciona el dispositivo Bluetooth deseado de la lista mostrada en pantalla, la interfaz gráfica de la aplicación vuelve a actualizarse para reflejar el establecimiento de la conexión con el sensor seleccionado.

En primer lugar, la línea correspondiente al dispositivo elegido pasa a mostrarse en un tono gris oscuro, permitiendo identificar visualmente de manera clara cuál es el sensor con el que se ha realizado la conexión. Este cambio visual facilita la interacción con la aplicación y evita posibles confusiones cuando existen múltiples dispositivos BLE disponibles en la lista.

A continuación, los campos correspondientes a las distintas características del sensor, que anteriormente aparecían vacíos, se completan automáticamente con la información recibida desde el dispositivo BluEMG. De este modo, la aplicación comienza a mostrar por pantalla

los datos asociados al sensor, permitiendo al usuario conocer diferentes parámetros de información relevantes del dispositivo conectado.

Finalmente, también se produce una actualización en la etiqueta correspondiente al estado de la aplicación. El texto mostrado pasa a ser "Conectado", indicando que la conexión Bluetooth Low Energy se ha establecido correctamente y que la comunicación entre la aplicación móvil y el sensor se encuentra activa y funcionando de manera satisfactoria.

A continuación, se muestra la interfaz gráfica en este último paso:

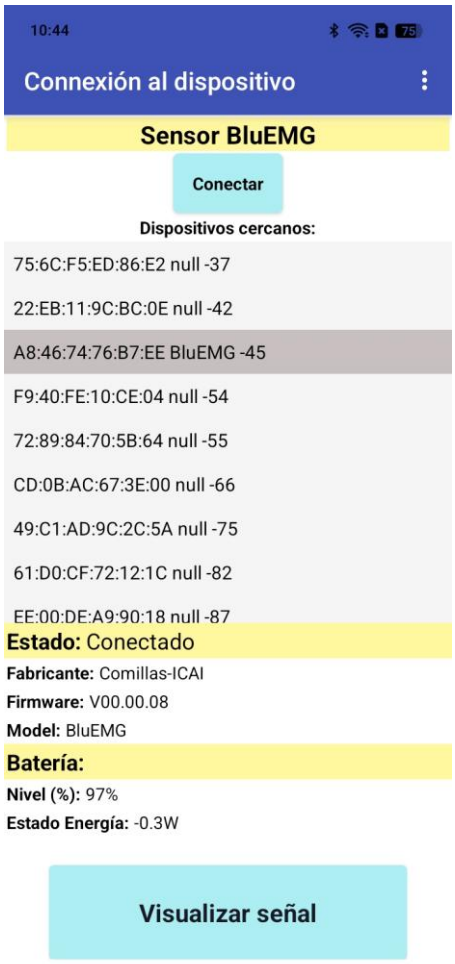


Ilustración 66: 1º pantalla con sensor conectado

Seguidamente, se mostrarán los resultados obtenidos en la segunda pantalla, en este caso, la pantalla principal de la aplicación.

Para saltar a la pantalla principal es tan sencillo como tocar sobre el botón ubicado en la parte inferior de la primera pantalla donde aparece "Visualizar señal".

Una vez se toca sobre el botón se abre la pantalla principal la cual tiene la siguiente interfaz gráfica:

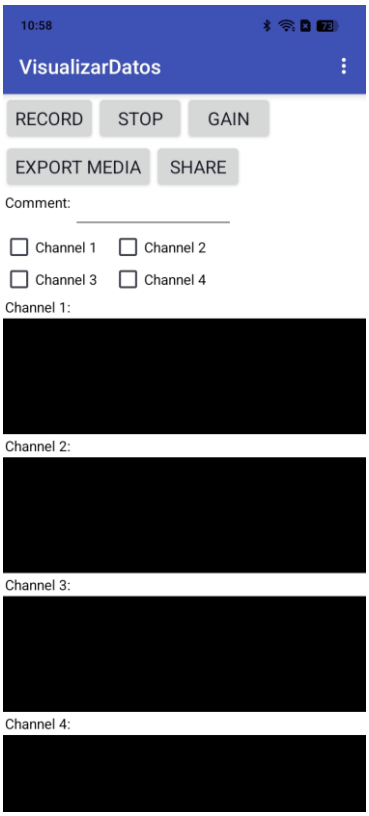


Ilustración 67: Pantalla principapal, parte superior

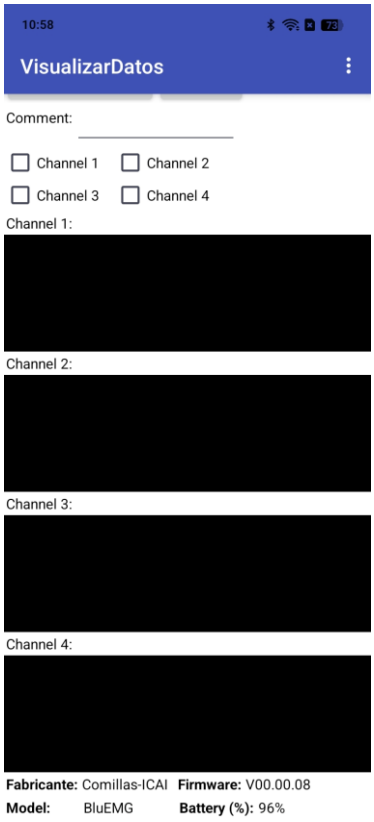


Ilustración 68: Pantalla principapal, parte inferior

La imagen situada a la izquierda muestra la parte superior de la pantalla principal de la aplicación, mientras que la imagen de la derecha representa la parte inferior de dicha

pantalla. Ambas capturas permiten observar la mayor parte de las funcionalidades implementadas en la interfaz principal, encargada de gestionar tanto la adquisición de señales electromigráficas como las diferentes opciones de configuración y almacenamiento de datos.

En la imagen de la izquierda se pueden observar los cinco botones principales que controlan gran parte de la lógica de funcionamiento de la aplicación. Estos botones permiten al usuario interactuar con el sensor BluEMG, gestionar la adquisición de datos y exportar posteriormente la información obtenida.

El primer botón, denominado *"Record",* tiene como finalidad iniciar la grabación de las señales correspondientes a los canales que se encuentran seleccionados en ese momento. La aplicación únicamente permitirá comenzar la grabación si existe al menos un canal activo; en caso contrario, la grabación no podrá iniciarse.

A la derecha del botón *"Record"* se encuentra el botón *"Stop",* cuya función es detener la grabación en curso. Este botón solamente tendrá efecto cuando previamente se haya iniciado una grabación válida, evitando así posibles errores derivados de detener procesos inexistentes.

Seguidamente, a la derecha del botón *"Stop",* aparece el botón *"Gain"*. Al pulsar sobre este botón, la aplicación despliega un submenú que contiene las diferentes opciones de ganancia disponibles para el sensor. Esto permite al usuario seleccionar el nivel de ganancia más adecuado en función de las necesidades de adquisición y visualización de la señal electromiográfica.

Debajo de estos tres botones principales se encuentran los dos botones restantes relacionados con la exportación y compartición de datos.

El botón *"Export media"* es el encargado de generar automáticamente un archivo en formato CSV que contiene los datos obtenidos durante la grabación de las señales. Para que el archivo pueda generarse correctamente, es necesario haber detenido previamente la grabación

mediante el botón "Stop". Una vez completado este proceso, los datos quedan preparados para ser almacenados y utilizados posteriormente.

Por otro lado, a la derecha del botón *"Export media",* se encuentra el botón *"Share"*. Su función consiste en abrir un menú desplegable con las distintas aplicaciones instaladas en el dispositivo móvil a través de las cuales es posible compartir el archivo CSV generado, facilitando así la exportación y distribución de los datos obtenidos.

Justo debajo de la zona de botones aparece un campo de texto acompañado de las etiquetas "Comments". Este apartado permite al usuario introducir comentarios o anotaciones relacionadas con la grabación realizada. Dichos comentarios se almacenan junto con el archivo correspondiente, permitiendo añadir información contextual útil para posteriores análisis o revisiones de las señales adquiridas.

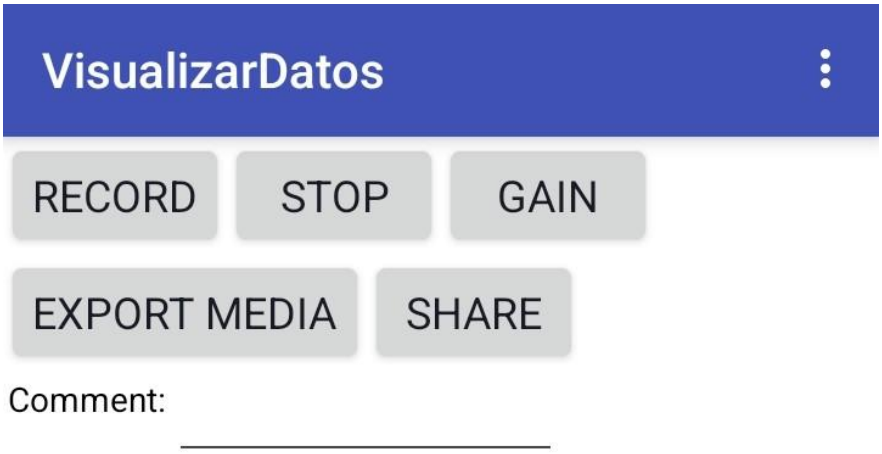


Ilustración 69: Botones pantalla principal

A continuación, debajo del campo de comentarios, se encuentran los CheckBoxes correspondientes a los cuatro canales disponibles del sensor BluEMG. Para iniciar la transmisión y visualización de datos de un canal concreto, es necesario seleccionar el CheckBox correspondiente. Una vez activado, la señal asociada a dicho canal comienza a recibirse y mostrarse en tiempo real en la interfaz de la aplicación.

Finalmente, en la parte inferior de la imagen de la izquierda, pueden observarse los cuatro componentes Canvas utilizados para representar gráficamente las señales electromiográficas recibidas desde cada uno de los canales activos. Sobre estos elementos se dibujan en tiempo real las señales adquiridas, permitiendo al usuario monitorizar visualmente la actividad registrada por el sensor.

Por otro lado, la imagen situada a la derecha muestra la misma interfaz descrita anteriormente, pero desplazada hacia la parte inferior de la pantalla. En esta zona adicional se encuentran las diferentes características e información relevante del sensor BluEMG, tales como datos relacionados con el dispositivo y su estado de funcionamiento. Debido a la disposición vertical de la interfaz, es necesario desplazarse hacia la parte inferior de la pantalla para poder visualizar completamente esta información adicional.

Ilustración 70: Menú desplegable ganancia

En la imagen anterior se muestra el menú desplegable que aparece por pantalla cuando se selecciona el botón *"Gain"*. Una vez abierta esta ventana, se podrá seleccionar el valor de

ganancia deseado y se aplicará automáticamente sobre las señales mandadas por el sensor BluEMG.

A continuación, en las imágenes 70 y 71, se puede observar la representación gráfica de las señales electromiográficas mostradas en tiempo real por la aplicación. Como puede apreciarse, únicamente se visualizan en sus respectivos elementos Canvas aquellas señales correspondientes a los canales cuyos Check Box se encuentran seleccionados, permitiendo así activar o desactivar de forma dinámica la visualización de cada canal según las necesidades del usuario.

Además, con el objetivo de facilitar la identificación y diferenciación visual de las señales adquiridas, cada uno de los canales se representa mediante un color distinto. De esta manera, se mejora tanto la claridad de la interfaz como la interpretación simultánea de múltiples señales durante el proceso de monitorización.

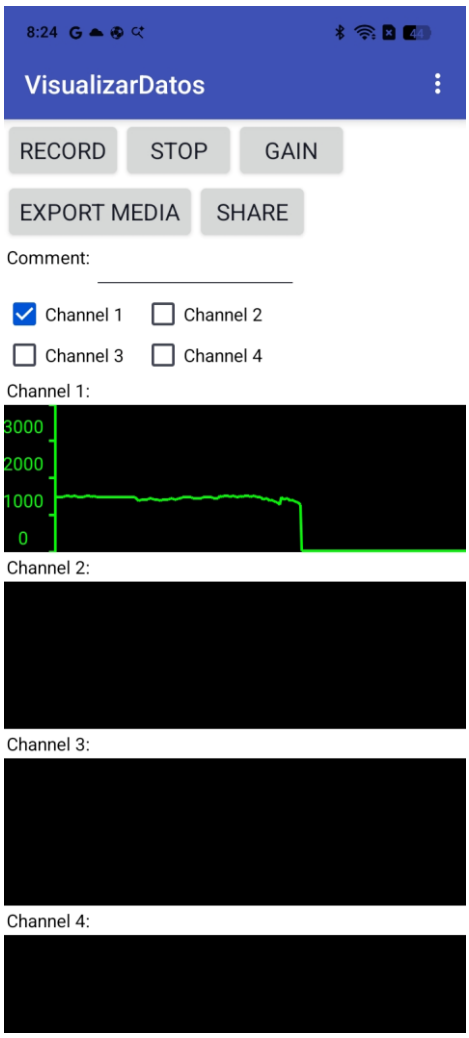


Ilustración 72: Pantall principal con el canal 1 seleccionado

Ilustración 71: Pantalla principal con canales 1 y 3 seleccionados

	A	B	C
1	date	05/28/2026 02:41:07 p.Â m.	
2	gain	x5	
3	comment	prueba con ganancia 5	
4			
5	sample	ch1	ch2
6	1	1386	1388
7	2	1387	1393
8	3	1378	1388
9	4	1396	1392
10	5	1389	1402
11	6	1397	1386

Ilustración 73: Ejemplo de fichero CSV generado

En la imagen anterior se muestra el formato correspondiente al archivo CSV generado al pulsar el botón *"Export media"*. Este archivo contiene la información registrada durante la adquisición de señales, organizada de manera estructurada para facilitar su posterior análisis y procesamiento.

En la parte superior del documento se incluyen la fecha y la hora exactas en las que se ha generado el archivo, permitiendo identificar con precisión el momento de la exportación de los datos. Justo debajo, se especifica el valor de ganancia aplicada durante la adquisición de las señales, información relevante para interpretar correctamente la amplitud de los datos registrados.

A continuación, se incorpora un apartado destinado a los comentarios introducidos por el usuario en el campo *"Comments*", permitiendo añadir anotaciones o información adicional relacionada con la grabación realizada. Posteriormente, se deja una línea en blanco con el objetivo de mejorar la organización y legibilidad visual del archivo.

Finalmente, se muestran los nombres de los canales que se encontraban activos durante la adquisición. Debajo de cada canal aparecen las muestras correspondientes a las señales registradas, organizadas secuencialmente y numeradas mediante una columna situada en la

parte izquierda del archivo, lo que facilita tanto la identificación como el análisis de los datos obtenidos.

Aunque actualmente la aplicación no establece un límite explícito al tamaño de los archivos CSV generados. El tamaño final depende directamente de la duración de la grabación y del número de canales activos durante la adquisición.

Los datos registrados se almacenan progresivamente en memoria y posteriormente se exportan en formato CSV. Por tanto, el límite práctico viene determinado principalmente por la memoria disponible del dispositivo móvil utilizado y por las limitaciones de almacenamiento del propio sistema operativo.

En grabaciones habituales de varios minutos, el tamaño generado resulta perfectamente manejable para la aplicación, pero cuando se trata de una grabación de mayor duración, como pueden ser 15 minutos o más, es posible que se dé una ralentización de la aplicación.

Esta ralentización no afecta directamente a la comunicación BLE, sino principalmente al procesamiento interno y a la gestión de memoria del dispositivo móvil.

Capítulo 6. CONCLUSIONES Y TRABAJOS FUTUROS

En conclusión, el desarrollo de la presente aplicación ha permitido cubrir satisfactoriamente la necesidad de disponer de una herramienta móvil capaz de establecer comunicación con el sensor Bluetooth EMG mediante Bluetooth Low Energy, adquirir las señales electromiográficas enviadas por el dispositivo y representarlas gráficamente en tiempo real de forma clara e intuitiva. Además, la aplicación desarrollada no solo permite la visualización de las señales biomédicas, sino que también incorpora funcionalidades adicionales orientadas a facilitar el trabajo del usuario, como la posibilidad de grabar los datos obtenidos, exportarlos en formato CSV y compartir posteriormente dichos archivos a través de las distintas aplicaciones compatibles disponibles en el dispositivo móvil utilizado.

Asimismo, este proyecto ha demostrado la viabilidad de utilizar plataformas de desarrollo visual como MIT App Inventor para la creación de aplicaciones biomédicas funcionales, capaces de gestionar comunicaciones inalámbricas, procesado básico de señales y almacenamiento de información biomédica. Esto pone de manifiesto el potencial de este tipo de herramientas para el desarrollo rápido de soluciones accesibles dentro del ámbito de la ingeniería biomédica y la monitorización de señales fisiológicas.

No obstante, todavía existen determinadas funcionalidades que podrían implementarse como futuras líneas de mejora del proyecto. Entre ellas, destaca el desarrollo de un sistema de configuración de ganancia independiente para cada canal de adquisición. Actualmente, los sensores utilizados durante el desarrollo del proyecto no disponen de dicha funcionalidad a nivel de hardware, motivo por el cual no ha sido posible incorporarla en la aplicación móvil desarrollada. En caso de disponer de versiones futuras del sensor con soporte para esta característica, la aplicación podría ampliarse para incluir esta funcionalidad de manera relativamente sencilla.

Por otro lado, otra de las limitaciones actuales del proyecto es la compatibilidad con dispositivos iOS. Aunque MIT APP Inventor dispone actualmente de soporte para este

sistema operativo, dicha compatibilidad se encuentra todavía en fase beta de desarrollo, por lo que el funcionamiento de determinadas funcionalidades relacionadas con Bluetooth Low Energy no es completamente estable ni funcional en dispositivos Apple. Debido a que esta limitación depende directamente del estado actual de desarrollo de la plataforma utilizada y no de la propia aplicación implementada, queda pendiente esperar a futuras actualizaciones de MIT APP Inventor que permitan mejorar la estabilidad y compatibilidad con iOS para poder extender el uso de la aplicación también a ese sistema operativo.

Capítulo 7. BIBLIOGRAFÍA

[1] J. Xu, A. Haider, A. Sheikh, y M. González-Fernández, «Epidemiology and Impact of Limb Loss in the United States and Globally», *Phys. Med. Rehabil. Clin. N. Am.*, vol. 35, n.º 4, pp. 679-690, nov. 2024, doi: 10.1016/j.pmr.2024.05.003.

[2] F. Ahmed, A. Lyu, N. Xu, W. ksebe, Y. Ksaibe, y R. Kadoun, «The relationships between body image, self-esteem and quality of life in adults with trauma-related limb loss sustained in the Syrian war», *J. Vasc. Nurs.*, vol. 42, n.º 3, pp. 191-202, sep. 2024, doi: 10.1016/j.jvn.2024.05.005.

[3] «Bluetooth Low Energy (BLE): A Complete Guide», Novel Bits. Accedido: 28 de mayo de 2026. [En línea]. Disponible en: https://novelbits.io/bluetooth-low-energy-ble-complete-guide/

[4] «IMPLANT DEVICE»

[5] P. F. Pasquina *et al.*, «First-in-man demonstration of a fully implanted myoelectric sensors system to control an advanced electromechanical prosthetic hand», *J. Neurosci. Methods*, vol. 244, pp. 85-93, abr. 2015, doi: 10.1016/j.jneumeth.2014.07.016.

[6] «Tipos de Electrodos en Medicina | PDF | Electromiografia | Electromagnetismo». Accedido: 19 de mayo de 2026. [En línea]. Disponible en: https://es.scribd.com/document/460103365/Tipos-de-electrodos

[7] H. Khalid, M. K. Majeed, M. A. Mansoor, V. Aiman, A. Saleem, y R. Iqbal, «Biocompatible batteries: Powering the future of wearable and implantable medical devices», *J. Energy Storage*, vol. 158, p. 121429, may 2026, doi: 10.1016/j.est.2026.121429.

[8] «Baterías para marcapasos: energía vital y duradera | Baterías CEA». Accedido: 27 de mayo de 2026. [En línea]. Disponible en: https://bateriascea.com.ar/baterias-para-marcapasos/

[9] Anh, «Marcapasos: nueva batería implantable puede funcionar de por vida gracias al oxígeno de la sangre», Infoterio - Noticias científicas que explican el mundo. Accedido: 27 de mayo de 2026. [En línea]. Disponible en: https://www.infoterio.com/2024/04/Marcapasos-nueva-bateria-implantable-puede-funcionar-de-por-vida-gracias-al-oxigeno-de-la-sangre.html

[10] L. Gila, A. Malanda, I. R. Carreño, J. R. Falces, y J. Navallas, «Métodos de procesamiento y análisis de señales electromiográficas».

[11] E. M. Graham, A. Kota, M. K. Intintoli, A. Fried, A. Shah, y S. D. Mendenhall, «From iron hooks to moving hands: The evolution of partial hand prostheses—a surgical perspective», *Orthoplastic Surg.*, vol. 12, pp. 29-43, jun. 2023, doi: 10.1016/j.orthop.2023.05.005.

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS

Los Objetivos de Desarrollo Sostenible (ODS) Constituyen un conjunto de 17 metas globales adoptadas por la Organización de las Naciones Unidas en 2015, cuyo propósito es erradicar la pobreza, proteger el medio ambiente y garantizar la paz y la prosperidad para toda la población mundial en el horizonte del año 2030. Estos objetivos abarcan ámbitos muy diversos, como la salud, la educación., la innovación o la sostenibilidad ambiental, y sirven como marco de referencia para orientar proyectos y políticas hacia un desarrollo más equilibrado y responsable.

En este contexto, resulta fundamental que cualquier proyecto de carácter tecnológico o científico esté alineado con los ODS, de manera que su impacto no solo se limite al ámbito técnico, sino que también contribuya de forma positiva a la sociedad. En particular, el presente proyecto se relaciona de forma directa con dos de estos objetivos:

ODS 3. Salud y bienestar

Este es el objetivo sobre el que el proyecto tiene un mayor impacto. La propuesta se centra en el desarrollo de una tecnología biomédica para mejorar la calidad de vida de personas con pérdida de movilidad en alguna parte de su cuerpo. Mediante la implementación de soluciones tecnológicas avanzadas, se pretende facilitar la rehabilitación, aumentar la autonomía del paciente y reducir las limitaciones funcionales derivadas de su condición. De este modo, el proyecto contribuye no solo a la mejora del bienestar físico, sino también al bienestar psicológico y social de los usuarios.

ODS 9. Industria, Innovación e Infraestructura

El proyecto también se alinea con este objetivo al basarse en el desarrollo de una solución innovadora que integra diferentes disciplinas tecnológicas. En concreto, combina el uso de comunicaciones inalámbricas de bajo consumo energético, sensores implantables y desarrollo de aplicaciones software. Esta integración multidisciplinar fomenta la innovación en el ámbito de la ingeniería biomédica y promueve el avance de infraestructuras tecnológicas más eficientes y sostenibles. Además, impulsa la transferencia de conocimiento entre sectores clave como la ingeniería, la medicina y la tecnología, contribuyendo al progreso industrial y al desarrollo sostenible.

En conjunto, la alineación del proyecto con estos ODS refuerza su valor no solo desde un punto de vista técnico, sino también social, evidenciando su contribución al desarrollo de soluciones que buscan mejorar la calidad de vida de las personas y fomentar la innovación responsable.