ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
INGENIERÍA BIOMÉDICA

Desarrollo de una aplicación para medida y visualización de señales fisiológicas basada en LabVIEW

Development of a LabVIEW based tool for registration and visualization of physiological signals

Realizado por
Irene Mata Menéndez

Tutorizado por
Antonio Jesús Bandera Rubio
Carmen García Berdonés

Departamento
Tecnología Electrónica

UNIVERSIDAD DE MÁLAGA
MÁLAGA, SEPTIEMBRE DE 2022

Fecha defensa: Octubre de 2022

Resumen

El presente trabajo surge a raíz de la necesidad de usar, con fines docentes, un equipo de registro analógico de señales fisiológicas (LabLinc V de Colburn) que posee el Departamento de Tecnología Electrónica (DTE). El equipo contiene los sensores y la etapa de acondicionamiento para el registro de señales como Electrocardiografía o Fotopletismografía, pero se necesita complementar con una etapa de digitalización para poder visualizarlas y almacenarlas en un ordenador y así, conseguir que todo el sistema pueda usarse para la docencia en diversas asignaturas relacionadas con la Ingeniería biomédica que imparte el DTE. La implementación de este sistema es el objetivo de este Trabajo Fin de Grado. Para la parte de digitalización, se ha usado un módulo comercial de adquisición de datos (NI-6000 de National Instruments), que se escogió tras un estudio de las características de las señales fisiologicas que se permite adquirir el equipo analógico. La aplicación ha sido programada sobre el sistema de desarrollo LabView de National Instruments, que usa programación gráfica, por el interés de la autora de este trabajo por aprender las bases de este tipo de programación. Se ha pasado un plan de pruebas, modular e integral, que permite afirmar que el sistema resultante cumple con los requisitos mínimos establecidos. Se trata de una primera versión de la herramienta docente objetivo, susceptible de incorporar mejoras, algunas de la cuales se sugieren también en este documento.

Palabras clave: LabView, Módulos de adquisición de datos, Señales fisiológicas, Equipos de poligrafía.

Abstract

This work arises from the need to use, for teaching purposes, an analog recording equipment for physiological signals (LabLinc V from Colburn) owned by the Department of Electronic Technology (DTE). The equipment contains the sensors and the conditioning stage for recording signals such as Electrocardiography or Photoplethysmography, but it needs to be complemented with a digitalization stage to be able to visualize and store them in a computer and thus make the whole system usable for teaching in various subjects related to Biomedical Engineering taught by the DTE. The implementation of this system is the objective of this Final Degree Project. For the digitization part, a commercial data acquisition module (NI-6000 from National Instruments) has been used, which was chosen after a study of the characteristics of the physiological signals that the analog equipment allows to acquire. The application has been programmed on the LabView development system of National Instruments, which uses graphical programming, due to the interest of the author of this work to learn the basics of this type of programming. A modular and integral test plan has been passed, which allows us to affirm that the resulting system complies with the minimum requirements established. This is a first version of the target teaching tool, susceptible of incorporating improvements, some of which are also suggested in this document.

Keywords: LabView, Data Acquisition Modules, Physiological Signals, Polygraph equipment.

Índice

Resumen **2**

Abstract **3**

Índice **5**

Introducción **7**
1.1 Motivación **7**
1.2 Objetivos **8**
1.3 Metodología de trabajo **9**
1.4 Estructura de la memoria **9**

Estudio de las tecnologías implicadas **11**
2.1 Señales fisiológicas **11**
2.1.1 Biopotenciales 12
2.1.1.1. Electromiografía 12
2.1.1.2. Electrocardiografía 13
2.1.1.3. Elecroencefalografía 15
2.1.2 Fotopletismografía 17
2.1.3 Requisitos derivados para el sistema de adquisición de datos 19
2.2 Hardware disponible **20**
2.2.1 Módulos comerciales de adquisición de datos 20
2.2.1.1. Arquitectura general y prestaciones típicas 20
2.2.1.2. Adecuación a los requisitos de los módulos disponibles para la Implementación del sistema 22
2.2.2 Módulo de poligrafía 23
2.3 Software disponible **26**
2.3.1 Sistema de desarrollo LabVIEW: Carácterísticas generales 26
2.3.2 Sistema de desarrollo LabVIEW: Entorno de trabajo y funciones 27
2.3.2.1. Panel frontal: Terminales 29
2.3.2.2. Diagrama de bloques: Estructuras de programación 30
2.3.2.3. Diagrama de bloques: Funciones para la adquisición de datos 32
2.3.2.4. Diagrama de bloques: Funciones para el manejo de ficheros 33

Especificaciones del sistema **35**
3.1 Requisitos de implementación **35**
3.2 Casos de uso y Requisitos funcionales **35**
3.3 Consideraciones de diseño **38**

Desarrollo del sistema **39**
4.1 Descripción del interfaz de usuario. **39**
4.2 Diseño e implementación en LabView de la máquina de estado **41**
4.3 Descripción de los principales estados **46**
4.3.1 Adquisición, visualización y procesado de la señal 46
4.3.1.1. Almacenamiento 50
4.3.1.2. Procesado de los datos adquiridos 52
4.3.2 Lectura del fichero y visualización de los datos 52
4.3.3 Gestión de errores 54
4.3.4 Interfaz de usuario 55

Pruebas del sistema 59
5.1 Pruebas modulares 59
5.1.1 Adquisición, visualización y procesado de la señal 60
5.1.2 Configuración de los tiempo de adquisición 63
5.1.3 Almacenamiento en fichero 64
5.1.4 Lectura y representación desde fichero 66
5.2 Pruebas integrales 67
Conclusiones y líneas futuras 77
6.1 Conlusiones 77
6.2 Líneas futuras 78
Referencias 79
Manual de Instalación y Uso 81

1

Introducción

1.1 **Motivación**

El registro de señales fisiológicas no solo es util para la diagnosis o monitorización clínica, también la denominada Psicofisiología usa este tipo de señales para estudiar el estado de las personas que las generan. Por ejemplo, se puede determinar el nivel de carga mental, midiendo tasas respiratorias y cardiacas (A. Tiwari et al., 2020), o el estado emocional, midiendo vasodilataciones, actividad electrica facial e impedancia de la piel(Hassan et al., 2019) .Como en los ejemplos recién expuestos, a menudo es necesario registrar más de una señal, con lo que los equipos de registro asociados a este tipo de aplicaciónes se denominan de forma genérica polígrafos o equipos para investigación psicofisiológica.

Estos equipos de poligrafía suelen implementar la cadena de medida completa, incluyendo una conexión con un ordenador, y un software específico que, instalado en ese ordenador, permite visualizar las señales, guardarlas o procesarlas. Suelen ser equipos muy completos pero totalmente cerrados que no permiten adquirir o procesar la señal de una foma diferente a la que su propio software permite. Los equipos de la marca Biopac (BIOPAC, n.d.) son un ejemplo de este tipo de soluciones. En el otro extremo, encontramos sistemas que son completamente abiertos, como el e-Health Sensor Shield para Arduino (Cooking Hacks, n.d.) cuyas aplicaciónes pueden ser programadas a nivel de microcontrolador. En un punto intermedio se encuentra el equipo LabLinc V de Colburn Instruments (Coulbourn Instruments, Lab Linc V System), adquirido por el Departamento de Tecnología Electrónica (DTE) hace más de una década, que contiene los sensores y la etapa de acondicionamiento para el registro de señales como Electrocardiografía (ECG) o Fotopletismografía (PPG, del inglés Photoplethysmography), pero no las digitaliza, permitiendo el acceso a las salidas analógicas y por tanto, permitiendo hacer cualquier tipo de adquisición y procesado con ellas, pero haciendo necesario la digitalización y posterior tratamiento de ellas en un ordenador.

El presente proyecto responde a la necesidad de usar el equipo LabLinc V para la docencia en diversas asignaturas relacionadas con la Ingeniería biomédica que imparte el DTE, . Para ello se hace necesario implementar, por una parte, una etapa de digitalización que permita disponer de las señales analógicas adquiridas en un ordenador, y, por otra, un interfaz de usuario que posibilite su visualización, almacenamiento y la programación de los parámetros de adquisición.

Se ha planteado usar, para la digitalización requerida, alguno de los módulos comerciales de adquisición de datos que también están disponibles en los laboratorios docentes del DTE y, para el desarrollo de la aplicación, LabVIEW de National Instruments (NI). Este sistema de desarrollo, por un lado, ofrece funciones para el manejo de los módulos de adquisición y, por otro, usa programación gráfica. El interés de la autora de este trabajo por aprender las bases de este tipo de programación, que no ha sido vista en el grado para el que se presenta este TFG, ha sido el factor determinante para elegir este sistema de desarrollo.

En la Figura 1 se representan los componentes involucrados en este trabajo.

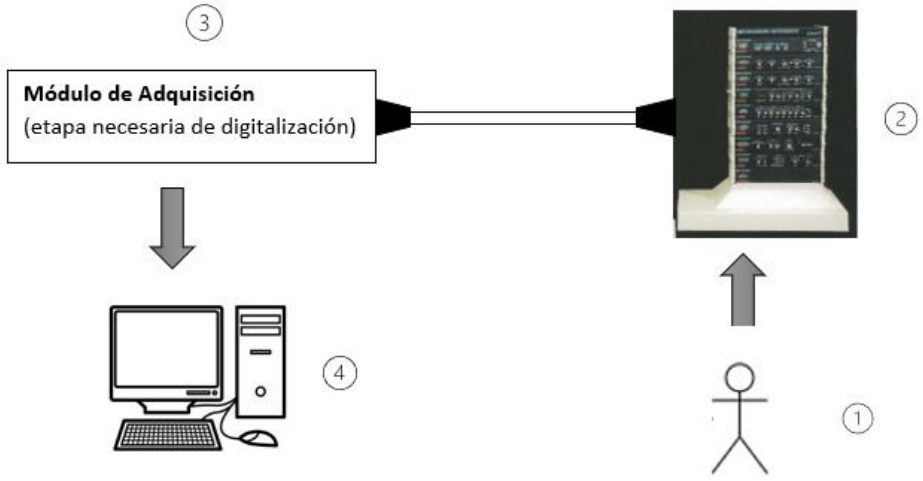


Figura 1. Componentes del proyecto: (1) Persona sobre la que se toman medidas (2) Equipo de poligrafía (3) Módulo de adquisición (4) PC.

1.2 **Objetivos**

El principal objetivo de este proyecto es el desarrollo de una aplicación capaz de adquirir en tiempo real los datos obtenidos a partir de un equipo de registro analógico de señales fisiológicas, haciendo uso de un módulo comercial de adqusición de datos. Esta aplicación permitirá la representación, procesado y recogida de datos en un fichero de las señales fisiológicas de interés, así como la visualización de las señales previamente adquiridas y almacenadas. Será programada en el sistema de desarrollo LabVIEW de NI, ya que como segundo objetivo se plantea el aprendizaje de la autora de este proyecto del tipo de programación en que se basa este sistema de desarrollo: la programación gráfica.

1.3 **Metodología de trabajo**

Para alcanzar el objetivo principal se plantea alcanzar los siguientes objetivos parciales:

1. Conocer y, en su caso seleccionar, el hardware y el software de partida:
   1.1. Las características del equipo de registro, esto es, para cada una de las señales fisiológicas que es capaz de registrar, los sensores que se usan, los parámetros de registro sobre los que puede actuar el usuario y las características de la señal analógica de salida.
   1.2. Las características de los módulos de adquisición disponibles en el DTE para poder escoger aquel o aquellos que mejor se adecuen a las características de las señales que provienen del equipo de registro.
   1.3. Los principios básicos de la programación en Labview, las funciones que ofrece para el manejo de los módulos de adquisición de datos y para la realización de interfaces de usuario.
2. Redactar los requisitos detallados de la aplicación objetivo. Algunos de estos requisitos serán marcados como de cumplimiento opcional y serán, si el tiempo lo permite, satisfechos por la aplicación, o propuestos como líneas futuras de este trabajo.
3. Diseñar, implementar, probar y documentar la aplicación.

Los tutores de este trabajo tomarán el rol de clientes de la aplicación y junto con la estudiante, establecerán sus requisitos mínimos. Con estos requisitos, y una vez que se esté familiarizado con el software y el hardware implicados en el desarrollo de la aplicación, se planteará la estructura modular de la misma, de forma que sea posible la implementación, documentación y prueba de cada módulo por separado y su posterior integración y verificación de los requisitos de alto nivel de la aplicación. Las pruebas modulares se realizarán primero haciendo uso de un generador de funciones como emulador de las señales provenientes del equipo de registro real, de esta manera se podrá, de forma controlada, inyectar a la aplicación señales en los límites de amplitud y frecuencia recogidos en la literatura para las distintas señales fisiológica.

1.4 **Estructura de la memoria**

Tras el presente capítulo de introducción, en el que se ha planteado la motivación y objetivos del trabajo, se continuará, en el capítulo 2, profundizando en los principales temas implicados en el desarrollo de la aplicación objetivo y, así, en primer lugar, se expondrán tanto las características relevantes para su adquisición de algunas señales fisiológicas típicas, como los recursos hardware y software de los que se dispone, para, en segundo lugar, seleccionar aquellas señales que sean susceptibles de adquirir con los recursos disponibles. En el capítulo 3, se mostrarán los requisitos que se han establecido para la aplicación objetivo, dedicando los dos siguientes capítulos a describir su desarrollo detallado y las pruebas que se han realizado para verificar los requisitos. El capítulo 6 mostrará las conclusiones a las que se ha llegado al finalizar este trabajo y algunas de las líneas que, para complementarlo, se pueden seguir en el futuro. La memoria finaliza con un manual que guíe al futuro usuario en la instalación y manejo de la aplicación desarrollada.

2

Estudio de las tecnologías implicadas

En este proyecto vamos a hacer uso de un equipo de registro analógico de señales fisiológicas, que se conectará a un módulo comercial de adquisición de datos. La combinación de ambos sistemas hardware deberá garantizar la correcta adquisición de las señales. Por tanto, y en primer lugar, se hará una breve descripción de algunas señales fisiológicas típicas que el equipo puede adquirir, para continuar presentando las características de los recursos hardware de los que se dispone. Este capítulo finalizará con una breve presentación del sistema de desarrollo software que se usará para el desarrollo de la aplicación objetivo.

2.1 **Señales fisiológicas**

Una señal es un medio de transmisión de información, cuya adquisición permite obtener información sobre la fuente que la generó. En el caso de los bioseñales, las fuentes son los diferentes sistemas fisiológicos del organismo. Las bioseñales se pueden clasificar atendiendo a su naturaleza y, así, encontramos señales acústicas, mecánicas, bioquímicas ópticas o eléctricas. En este capítulo nos centraremos en las señales eléctricas, los denominados biopotenciales, y en una de tipo óptico, la fotopletismografía, básicamente porque, como se verá, son las que el equipo de registro analógico es capaz de acondicionar. A continuación, se presentarán su origen y sus correspondientes características.

2.1.1 Biopotenciales

Un biopotencial es el resultado de un cambio electroquímico que se da en las células excitables cuando se despolariza y repolariza la membrana de la célula. Esta diferencia de potencial bioeléctrico se puede captar en la superficie del cuerpo mediante electrodos. Dependiendo de la zona en la que coloquemos los electrodos, se podrá obtener señales que reflejen por ejemplo la actividad eléctrica muscular (EMG, Electromiografía), cardiaca (ECG, Electrocardiografía) o neuronal (EEG, Electroencefalografía).

2.1.1.1. Electromiografía

La electromiografía (EMG) registra la actividad eléctrica y la información sobre el estado de la estructura y función de los músculos. Toma información sobre la función controladora de los sistemas nerviosos central y periféricos sobre los músculos.

Un concepto básico en electromiografía es la denominada unidad motora (MU), que representa el elemento anatómico y funcional del sistema neuromuscular. Los cambios eléctricos generados por la actividad de la MU son los que se adquieren y amplifican mediante electrodos ubicados en la masa muscular. La representación de los cambios generados por una MU es el llamado potencial de acción de la unidad motora (MUAP). Esta onda suele tener un rango de amplitud entre 0.25 y 5mV y un rango de frecuencia entre 70 y 130Hz. (L Sörnmo & P Laguna, 2005). En la Figura 2 podemos ver representado seis MUAPs diferentes con distintas amplitudes.

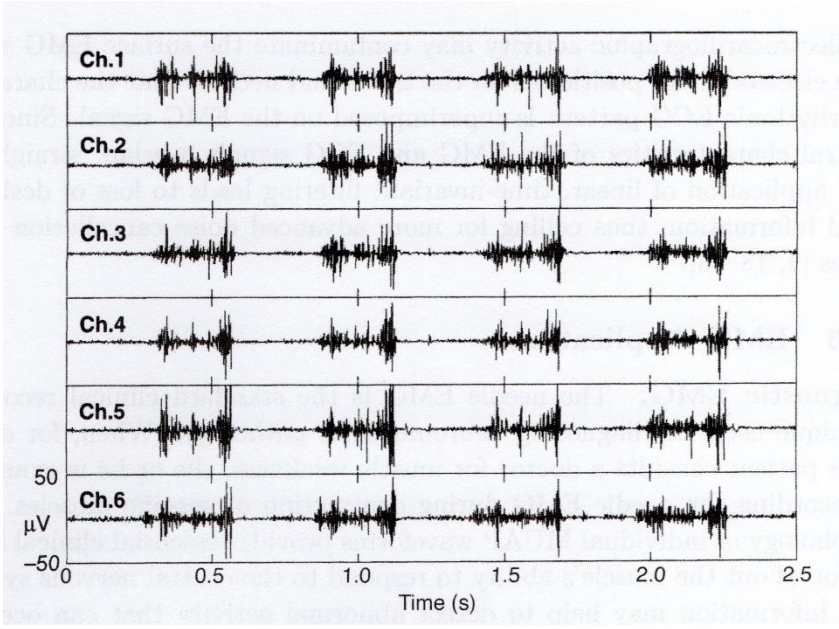


Figura 2. Ejemplo de 6 MUAPs de distintas amplitudes. (L Sörnmo & P Laguna, 2005)

2.1.1.2. Electrocardiografía

La electrocardiografía (ECG) registra el funcionamiento eléctrico del corazón y representa los cambios de polarización que suceden en las cámaras contráctiles que lo forman. En la unión de la aurícula derecha y vena cava superior se encuentra el nodo sino-auricular (NSA) que forma el marcapaso o activador cardiaco y que fija la frecuencia cardiaca.

Para obtener esta señal se sitúan electrodos en determinadas posiciones del cuerpo. En la Figura 3 podemos ver el sistema de 12 derivaciones, el más comúnmente usado. En la Figura 3.a se muestra las posiciones de los electrodos para registra las derivaciones bipolares I, II y III, en la Figura 3.b las posiciones para registrar las derivaciones unipolares aVR, aVL y aVF y en la Figura 3.c las derivaciones precordiales V1…V6. Se puede observar que se trata de medidas diferenciales.

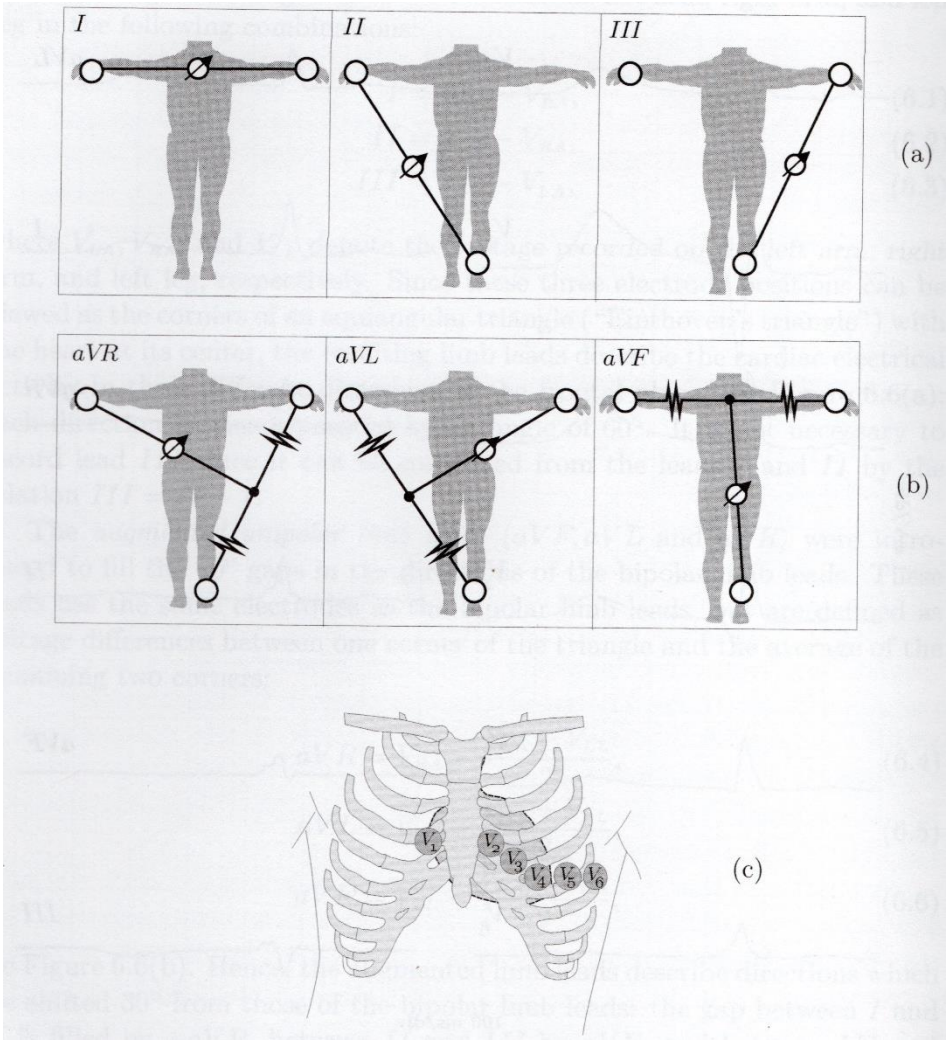


Figura 3. Sistema de 12 derivaciones de ECG. (L Sörnmo & P Laguna, 2005)

A continuación, se nombran las ondas que forman un ECG de una persona sin patologías y obtenida con los electrodos en la derivación III junto a algunas características importantes.

- La **onda P** representa la despolarización auricular, que da como resultado una contracción o sístole auriculares. Su amplitud es normalmente inferior a 300uV y su duración es inferior a 120ms. La característica espectral de una onda P normal suele ser de frecuencia baja, por debajo de 10-15Hz.
- El **complejo QRS**. representa la despolarización que precede a la contracción de los ventrículos. Dura alrededor de 70-110ms. El complejo QRS tiene la mayor amplitud del ECG, a veces alcanza hasta 2-3mV. Del mismo modo, la frecuencia de este es considerablemente la mayor de las ondas de ECG; se concentra en un intervalo de 10-50Hz.
- El **segmento T**. No es realmente una onda pero representa el intervalo de tiempo donde el ventrículo permanece en un estado activo y despolarizado.
- La **onda T**. Afecta a la repolarización ventricular y se extiende unos 300ms después del complejo QRS. A veces esta onda va seguida de otra onda lenta, llamada onda U, cuyo origen se cree que es la representación de la repolarización de las fibras de Purkinje.
- El **intervalo RR**. Representa la duración de un ciclo cardiaco ventricular; sirve como indicador de la frecuencia ventricular.

En la Figura 4 quedan representadas las ondas mencionadas en una persona sana durante un latido cardiaco.

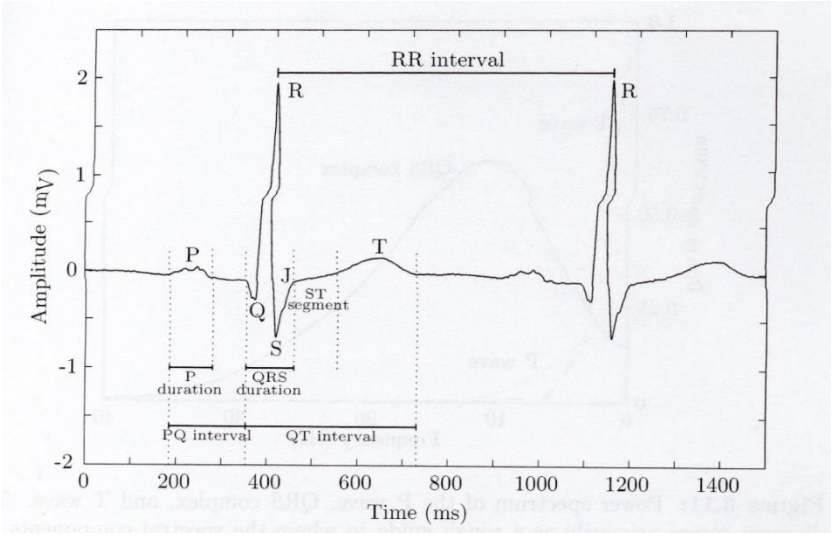


Figura 4. Ondas del ECG. (L Sörnmo & P Laguna, 2005)

Como se puede ver, la onda de mayor amplitud (R) es la que marca el máximo rango dinámico de la señal. Aunque es del orden de magnitud que se muestra en la Figura 4, la amplitud de esta onda depende de cada persona, así que la literatura suele establecer un rango dinámico de 10 mV para asegurar que los equipos que se ajusten a dicho rango sean capaces de adquirir el ECG de cualquier persona sin que se produzca una saturación en la medida (UNE, UNE-EN 60601-2-47, 2015) Respecto al rango de frecuencias en la que se considera que hay información útil, depende de la aplicación para la que se registre el ECG Por ejemplo, la normativa de marcado

CE para equipos de registro ambulatorio (UNE, UNE-EN 60601-2-47, 2015), que son equipos de monitorización, establece el rango en [0,5-40,0] Hz, mientras que para diagnosis se establece un rango más amplio, entre 0,05 y 200 Hz (Webster, 2009).

2.1.1.3. Elecroencefalografía

La electroencefalografía (EEG) capta la señal que recoge la actividad bioeléctrica de las neuronas de la corteza cerebral del cerebro.

La medida de la actividad de una sola neurona cortical requeriría un procedimiento invasivo. Sin embargo, la actividad conjunta de millones de neuronas corticales produce un campo eléctrico lo suficientemente fuerte para ser medido a nivel. El campo eléctrico es generado principalmente por corrientes que fluyen durante la excitación sináptica de las dendritas. La diversidad de ritmos de esta onda es enorme y depende, en gran parte, del estado mental del sujeto, como el grado de atención, vigilia y sueño.

El ritmo encefalográfico se clasifica en cinco bandas de frecuencias diferentes (L Sörnmo & P Laguna, 2005), representadas en la Figura 5.

- **Ritmo deltha** < 4Hz. Se suele encontrar durante la etapa de sueño profundo. Tiene una gran amplitud.
- **Ritmo theta** 4-7Hz. Ocurre durante la somnolencia y ciertas etapas del sueño.
- **Ritmo alfa** 8-13Hz. Este ritmo predomina en sujetos relajados.
- **Ritmo beta** 14-30Hz. Este ritmo es rápido y de baja amplitud.

Los ritmos de alta frecuencia y baja amplitud reflejan un cerebro activo asociado con el estado de alerta o sueño profundo, mientras que los ritmos de baja frecuencia y alta amplitud se asocian a estados de somnolencia o de sueño sin sueños. Esto tiene su lógica ya que, cuando la corteza está más involucrada en la producción de información, el nivel de actividad de las neuronas corticales es relativamente alto pero también activamente desincronizado. En otras palabras, cada neurona está involucrada en distintas tareas cognitivas, dispara rápidamente pero no simultáneamente con las demás. Esto lleva a una baja sincrónica, por lo que la amplitud es baja. Por el contrario, durante el estado de sueño profundo, las neuronas corticales no participan en el procesamiento de información y están excitadas por una misma entrada rítmica. En este caso la sincronía es alta por lo que la amplitud es alta. En la Figura 6 se muestra una serie de EEG observados durante diferentes estados.

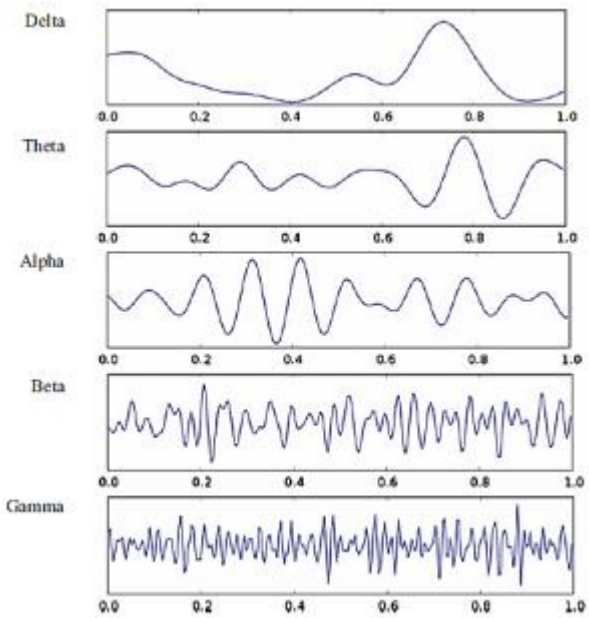


Figura 5. Ritmos encefalográficos. (Koudelková et al., 2018)

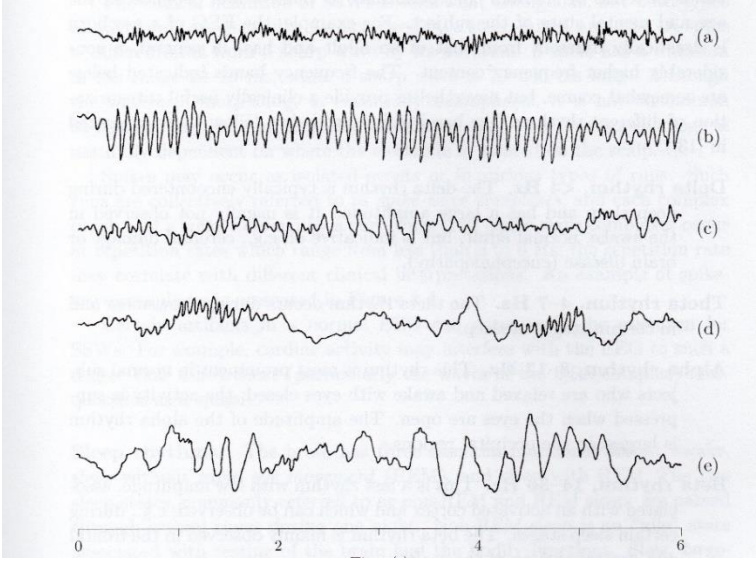


Figura 6. Ritmos encefalográficos observados durante distintos estados: a) excitado, b) relajado, c) soñoliento, d) dormido y e)profundamente dormido. (L Sörnmo & P Laguna, 2005)

En general, las señales registradas desde el cuero cabelludo tienen una amplitud que van hasta los 100 μV y una frecuencia que oscila entre 0.5Hz y 30-40Hz (L Sörnmo & P Laguna, 2005). Estos valores pueden variar dependiendo del estado mental del sujeto, de la posición de los electrodos y del propio sujeto.

Al igual que en la ECG, existen posiciones de los electrodos estandarizadas, siendo el denominado sistema 10/20 el más extendido (Figura 7). El registro de la señal se hace siempre entre dos puntos de este sistema. Como en el ECG, se trata de registros diferenciales.

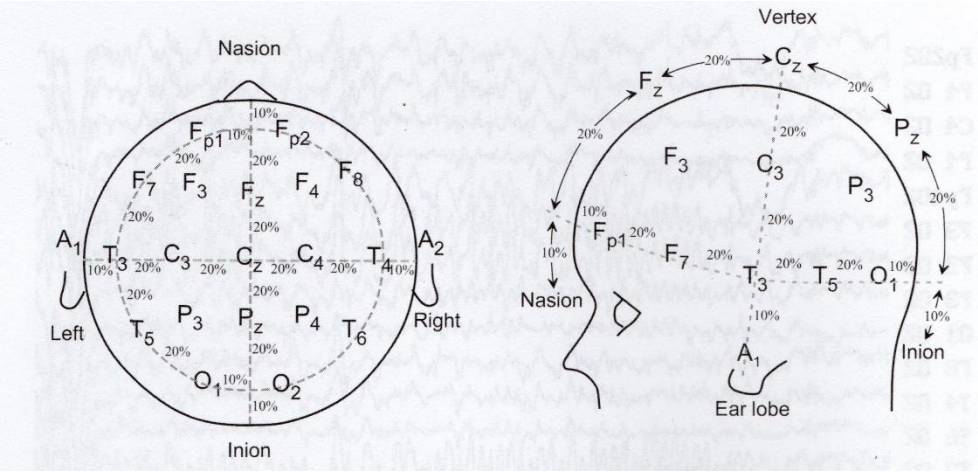


Figura 7. Sistema 10/20 de los electrodos para la ECG. [(L Sörnmo & P Laguna, 2005)

2.1.2 Fotopletismografía

La fotopletismografía (PPG, de sus en inglés Photoplethysmography) es una técnica de pletismografía que registra las variaciones de volumen sanguíneo de forma no invasiva según la absorción de luz de los tejidos. Se suele usar para medir la frecuencia cardiaca, definida como el número de latidos cardiacos por minuto.

Un fotopletismógrafo mide los cambios en la absorción de la luz que se da en cada latido. Cada ciclo cardiaco aparece como un pico en la onda del fotopletismógrafo, como se observa en la Figura 8.

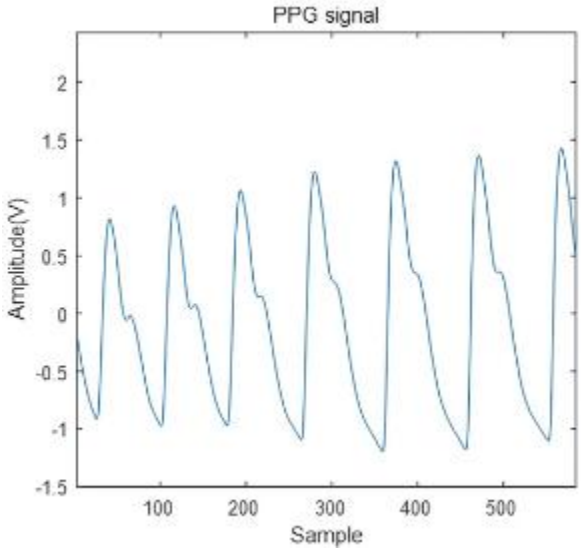


Figura 8. Señal PPG. (Liu et al., 2020)

Existen dos técnicas para adquirir la señal PPG, por transmisión/absorción, donde el emisor y detector se encuentran enfrentados y miden los cambios de luz que consigue atravesar la zona, y por reflexión, donde el sensor se coloca en cualquier superficie vascular pulsátil. Esta última técnica suele tomar una señal más débil en comparación

con la técnica de transmisión (Li et al., 2018). En la Figura 9 se representan estas dos técnicas.

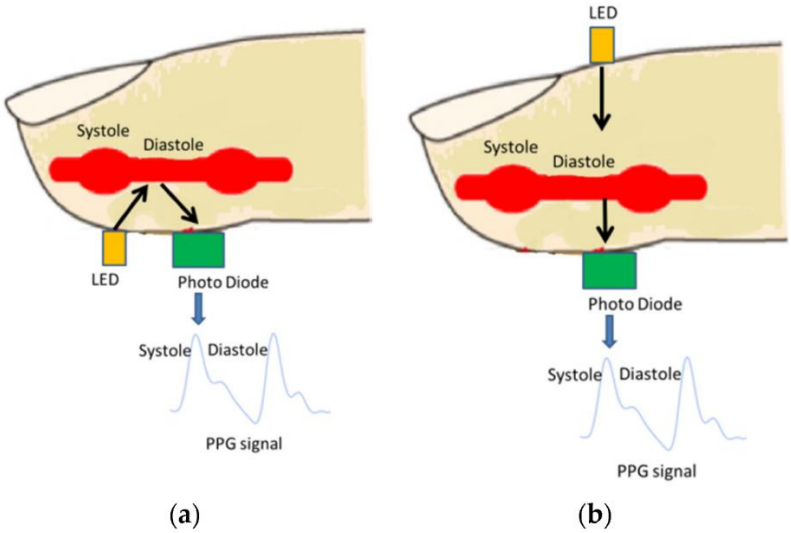


Figura 9. Técnicas obtención de PPG. (a) método de reflexión (b) método de absorción/transmisión. (Liu et al., 2020)

Las características de un pulso PPG incluyen principalmente el pico principal (pico sistólico), muesca diastólica seguido del pico diastólico, la anchura del pulso y la amplitud del mismo, tal y como se muestra en la Figura 10.

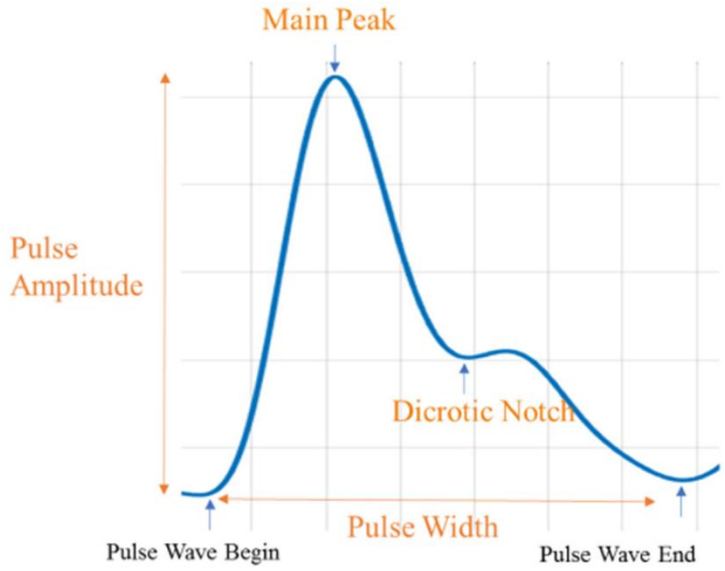


Figura 10. Características de un pulso PPG.(Liu et al., 2020)

Se trata de una señal lenta y con pocos picos pronunciados, el rango de frecuencia que contiene información útil es bajo. La literatura recomienda rangos como [0,5-10] Hz (Park et al., 2022) o [0.05-20] Hz (Allen et al., 2006).

Respecto a la amplitud, se debe resaltar que la medida exacta no es relevante para el cálculo de la tasa ya que depende totalmente de la intensidad de la luz que se emita y del sujeto (dependiendo, por ejemplo, de las características de la piel o la temperatura del dedo), con lo que el equipo deberá permitir variar la intensidad de luz emitida o la

ganancia de la señal recibida, de forma que se pueda ajustar adecuadamente el registro de la señal a cada persona.

2.1.3 Requisitos derivados para el sistema de adquisición de datos

Los factores más importantes que caracterizan a las bioseñales desde el punto de vista de la instrumentación electrónica necesaria para su correcto registro son los rangos de amplitud y frecuencia, y su modo de registro (diferencial o referenciado a tierra). Para elegir el dispositivo de adquisición de datos más adecuado es necesario saber estas características de las señales con las que vamos a trabajar. A modo de resumen, en las cuatro primeras columnas de la Tabla 1 quedan recogidas las características de las señales mencionadas en el apartado 2.1. Como se ha comentado en el apartado 2.1.2, la señal PPG puede variar mucho dependiendo de la persona por lo que no se puede fijar, ni es relevante, una amplitud como con las otras señales.

Las dos últimas columnas de la Tabla 1 pretenden dar una primera idea de las necesidades que las características de las señales impondrían sobre el sistema de adquisición. Por un lado, la frecuencia de muestreo que sería necesaria. Hay que indicar que, si bien la frecuencia de muestreo del dispositivo debe ser al menos el doble del máximo del rango de frecuencia de la señal, según se indica en el teorema de muestreo o teorema de Nyquist (Stéphane, 2009), cuando las señales no son filtradas para que desaparezcan las frecuencias superiores a las de interés, se debe sobremuestrear (de ahí que se haya multiplicado por 5). También se muestra la amplificación que sería necesaria, suponiendo un rango del módulo de adqusición de 5v. Como se puede ver, son señales muy lentas, así que se previó pocos problemas con la frecuencia de muestreo, pero también son señales de muy baja amplitud.

Como se verá a continuación, en nuestro sistema, parte de estos requistos los satisface el equipo de poligrafía (modo de registro y amplificación) y parte el módulo de adqusición escogido (frecuencia de muestreo).

Tabla 1. Características de las diferentes bioseñales

SEÑAL	SEÑAL DIFERENCIAL	RANGO MAX	RANGO FRECUENCIA	FRECUENCIA MUESTREO (X5)	GANANCIA
ECG	Sí	10mV	0.05 - 200 Hz	1000Hz	500G
EEG	Sí	100uV	0.5-40Hz	200Hz	50000G
PPG	No	---	0.05-20 Hz	100Hz	---
EMG	Sí	0.25-5mV	70-130Hz	650Hz	1000G

2.2 **Hardware disponible**

La adquisición de datos es el proceso de medicion de un fenómeno fisico o electrico como voltaje, corriente, temperatura, presión o sonido. En la Figura 11 queda esquematizado los componentes esenciales involucrados en este proceso: un sensor, un módulo de acondicionamiento y digitalización, que debe incluir conversor analógico-digital (CAD), y un PC.

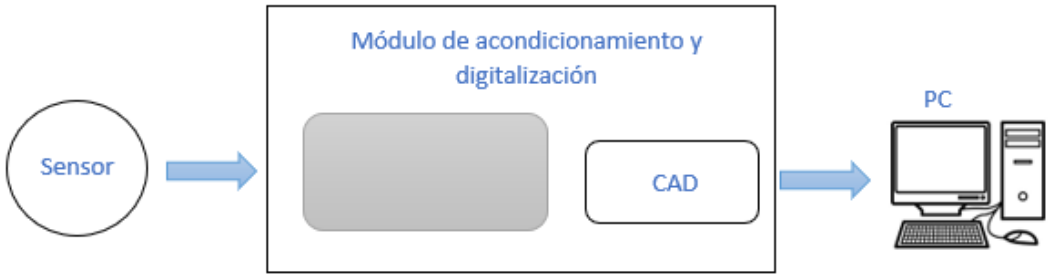


Figura 11. Componentes esenciales involucrados en DAQ.

El uso del PC en este proceso de adquisición aporta unas ventajas, en comparación con los sistemas de medidas tradicional, como pueden ser el procesamiento en tiempo real, la observación paso a paso del proceso y el manejo de cantidades de datos.

En los dos siguientes apartados hablaremos de los dispositivos disponibles para el desarrollo de este proyecto, que conformarán el módulo de acondicionamiento y adquisición.

2.2.1 Módulos comerciales de adquisición de datos

Los módulos comerciales de adquisición de datos, que denominaremos en lo que sigue dispositivos DAQ, se conectan al ordenador permitiendo al usuario recuperar valores de datos digitalizados. Suelen conectarse directamente al bus interno del ordenador a través de una ranura PCI-express o bien mediante interfaz USB. A continuación, pasamos a hablar sobre la arquitectura general de estos dispositivos y los requisitos a tener en cuenta a la hora de elegir un modelo.

2.2.1.1. Arquitectura general y prestaciones típicas

La arquitectura de los dispositivos DAQ puede variar dependiendo del modelo. La parte común se compone de un conversor analógico-digital (CAD) y una interfaz que le permite la conexión al ordenador, ya que será el que se encargue de la visualización y procesado de datos a tiempo real. Partiendo de esto, existe una amplia variedad de dispositivos DAQ dependiendo de los distintos componentes adicionales que ofrezcan. Así, pueden incluir amplificadores y filtros, que permiten el acondicionamiento y la adecuación de la señal al rango del CAD. También pueden disponer de componentes que convierte una señal digital en señal analógica, conocidos como conversor digital-analógico (CDA), o de puertos que permite

trabajar directamente con señales digitales. Normalmente tienen al menos un reloj interno (Timer), que se encarga de disparar la conversión analógico digital, esto es, la programación de este timer permite determinar la frecuencia de muestreo del dispositivo. Y también una memoria memoria FIFO ("First in, first out") para adecuar las velocidades de adquisición de los datos a las de envío de dichos datos hacia el PC. En la Figura 12 queda representado un dispositivo DAQ con los componentes recién indicados.

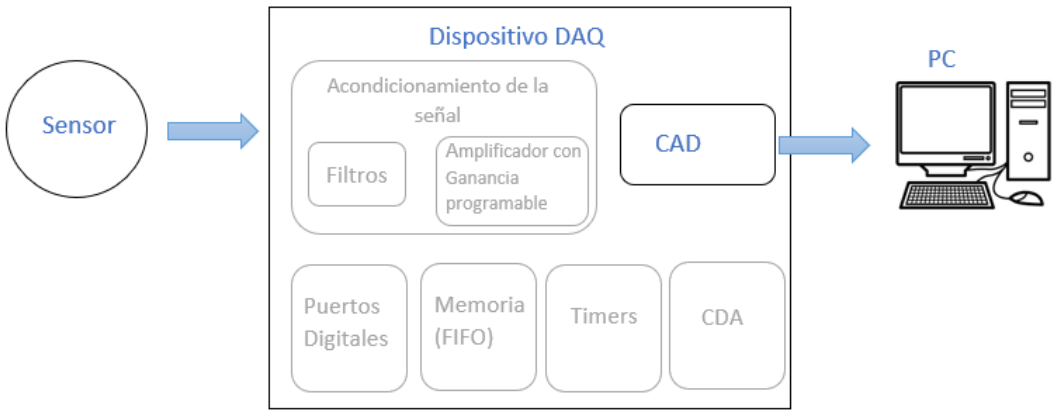


Figura 12. Componentes opcionales involucrados en la DAQ.

Por supuesto, las carácterísticas del CAD pueden variar de unos modelos a otros. Esto es, los dispositivos pueden tener diferentes valores de:

- ***Resolución.***
  La resolución del CAD corresponde al número de bits que se usan para representar la señal analógica. Cuanto mayor sea la resolución, más preciso será la representación de la señal.
- ***Frecuencia de muestreo.***
  La frecuencia de muestreo es el número de muestras por segundos que se toman de la señal origen.
- ***Rango y ganancia del amplificador.***
  El rango del CAD son los valores máximos y mínimos de la señal analógica que puede digitalizar. Si el dispositivo dsipone de un amplificador de ganancia programable, antes de que se digitalice la señal el propio dispositivo puede amplificar o atenuar para ajustala mejor al rango de entrada del CAD.

A modo ejemplo, recogemos en la Tabla 2 cuatro modelos distintos de dispositivos DAQ que comercializa NI junto algunas de sus características. El departamento DTE cuenta con los modelos que aparecen señalados en negrita.

Como se puede observar existe una gran variedad de dispositivos. Podemos encontrar desde un modelo con un precio de 131 euros y prestaciones mínimas, hasta uno de 7.912 euros que incorpora un CDA y una alta velocidad de muestreo. Los dispositivos de más altas prestaciones aseguran una frecuencia de muestreo por canal (ese es el significado de "/S/C" en la tabla), mientras que, en los de más bajas prestaciones, esta frecuencia depende del número de canales que se estén muestreando. En la tabla se

muestra la frecuencia máxima que ofrece el dispositivo (que corresponde al caso de que se esté adquiriendo un solo canal).

Tabla 2. Distintos modelos de DAQ. (National Instruments, n.d.-a)

		ENTRADA ANALOGICA				SALIDA ANALOGICA	
MODELO	**PRECIO**	**MAX NUM CANALES DE UNA SOLA TERMINAL**	**MAX NUM CANALES DIFERENCIAL**	**MAX VELOCIDAD DE MUESTREO**	**RESOLUCION**	**NUM CANALES**	**MAX VELOCIDAD DE ACTUALIZACION**
USB-6000	131€	8	0	10 KS/S	12BITS	0	-
USB-6002	535 €	8	4	50KS/S	16BITS	2	5KS/S
USB-6008	Fuera de catálogo	8	4	10 KS/S	10BITS	0	-
USB-6212	2346 €	16	8	400KS/S	16BITS	2	250KS/S
USB-6366	7912€	0	8	2MS/S/C	16BITS	2	3.33MS/s/c

2.2.1.2. Adecuación a los requisitos de los módulos disponibles para la Implementación del sistema

Si observamos los parámetros recogidos en la Tabla 1, la señal ECG es la que necesita la frecuencia de muestreo más alta (1000Hz). Por lo que, consultando la Tabla 2, podemos decir que cualquiera de los modelos disponibles, en lo que respecta a la frecuencia de muestreo, es apto para todas las señales mencionadas. Así que, motivos de disponibilidad, se preselecciona para realizar este trabajo el **modelo USB-6000**.

En la Figura 13 se muestra los componentes funcionales del modelo USB-6000 (NI, USB-6000 Specifications). El modelo cuenta con ocho canales no diferenciales de entrada analógicas de rango fijo ±10 V (el amplificador -AMP- tiene una ganancia no programable de 1), un buffer con un tamaño de 2047 muestras y una velocidad de muestreo máxima del CAD de 10Ks/s. Como se puede apreciar en la figura, los canales analógicos son recibidos por un multiplexor (MUX) que los direcciona hacia un único CAD y, por tanto, esa frecuencia máxima de muestreo tiene que ser dividida por el número de canales que se estén adquiriendo. Teniendo activos los 8 canales, aun se supera la frecuencia máxima requerida por canal para nuestra aplicación.

Hay que apuntar que, en el estudio de las señales fisiológicas realizado en el apartado anterior, no se ha incluido la mínima resolución con la que se debe adquirir cada señal, que impondría los requisitos relacionados con la resolución del CAD. Entendemos que los 12 bits de la tarjeta son suficientes para la correcta visualización de la señal en nuestra herramienta docente, y, en caso de requerir más

para alguna aplicación futura de procesado, habría que plantearse el uso de la USB-6002 o de la compra de otros modelos.

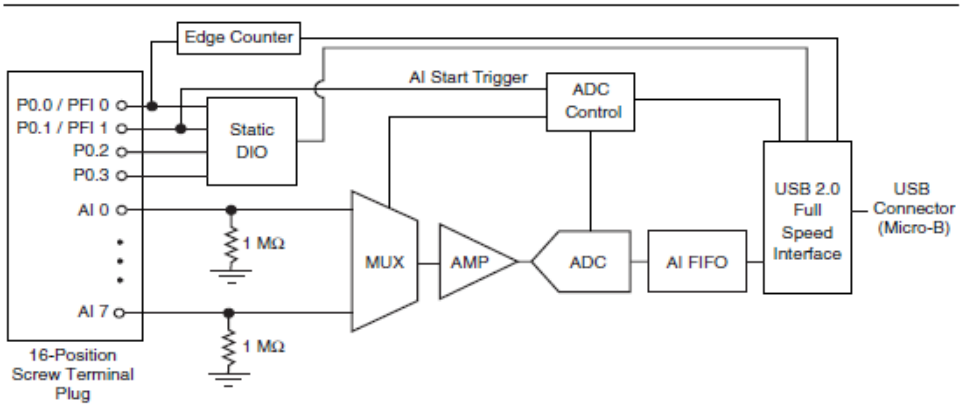


Figura 13. Arquitectura modelo USB-6000. (National Instruments, n.d.-b)

Los inconvenientes que encontramos es que el dispositivo ni cuenta con canales de entrada analógica diferencial ni puede aportar la ganancia requerida (Tabla 1). El uso del módulo de poligrafía, que se expondrá en el siguiente apartado, solventará ambos problemas.

2.2.2 Módulo de poligrafía

LabLincV es un sistema modular de instrumentos diseñado para la adquisición de datos analógicos multicanal. Consta de una fuente de alimentación aislada y de una serie de módulos de adquisición, procesamiento y control de señales, que se apilan verticalmente, uno encima de otro, en la unidad base de alimentación. Su aspecto se puede observar en la Figura 14. El primer módulo se conecta a la fuente de alimentación y cada módulo sucesivo que se va añadiendo se conecta del módulo inferior. La fuente de alimentación proporciona entre +15V y -15V. (COULBOURN, Lab Linc V - Coulbourn Instruments)

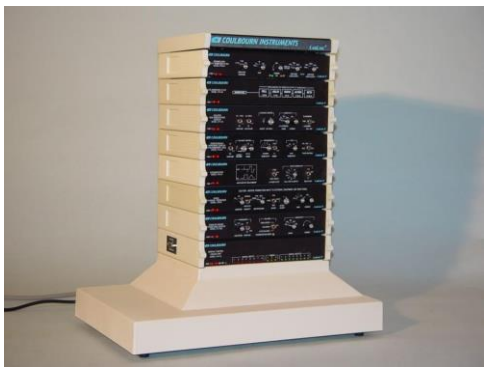


Figura 14. Sistema LabLincV.(coulbourn intruments, LabLincV.)

Existen diferentes tipos de módulos dependiendo de la señal que se desea adquirir. A continuación, se va a hablar de las características principales de los módulos encargados de medir la señal PPG, EEG y otros biopotenciales, ya que son de los que dispone el DTE.

2.2.2.1. MEDIDOR DE PPG

Este modulo, con nombre *V71-40 PULSE MONITOR/OPTICAL DENSITOMETER* , funciona según el principio de la fotopletismografía expuesto en el apartado 2.1.2. La luz es emitida y recibida por un fotoemisor y fotosensor en el cabezal del transductor. En la Figura 15 podemos el transductor que incorpora el equipo.

Figura 15. Sensor de fotopletismografía.

Como se ha indicado en el apartado 2.1.2, esta señal puede variar mucho según la persona. Este módulo cuenta con un único botón en la parte anterior, como se puede ver en la Figura 16, que sirve para amplificar la señal y ajustarla según nos convenga.

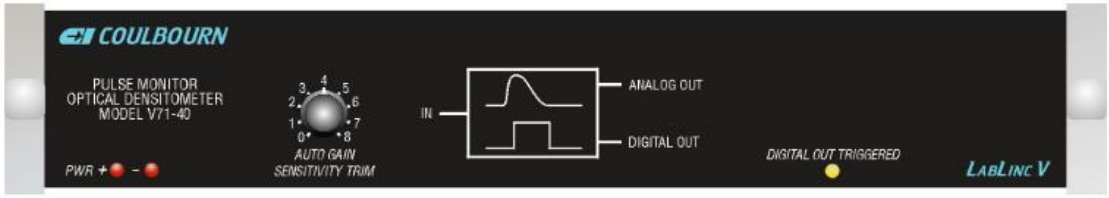


Figura 16. ModuloV71-40 PULSE MONITOR/OPTICAL DENSITOMETER parte anterior. (COULBOURN, n.d.)

En la Figura 17, podemos ver la parte posterior donde queda señalada la salida de la señal analógica.

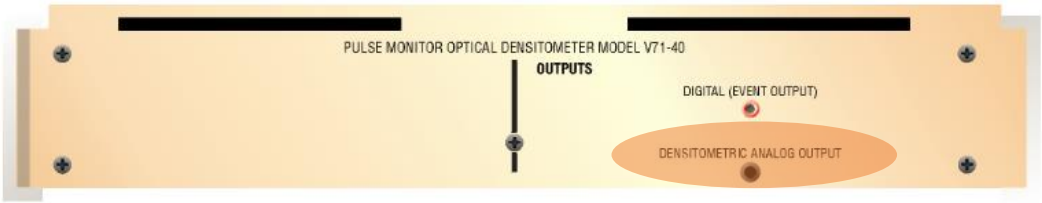


Figura 17. ModuloV71-40 PULSE MONITOR/OPTICAL DENSITOMETER parte posterior.(COULBOURN, n.d.)

2.2.2.2. MEDIDOR DE BIOPOTENCIALES

Este módulo, con nombre *V75-04 ISOLATED BIOAMPLIFIER WITH BANDPASS FILTER*, está diseñado para la adquisición de diferentes biopotenciales como

puede ser EMG, ECG y EEG. Lleva incorporado una red de filtros de paso banda que consta de 5 frecuencias de filtro de paso bajo y 5 de paso alto. Se puede aplicar a la señal de entrada ganacias en un rango de 100 a 50K, lo que hace que sea muy util para aplicaciones de señales de bajo nivel como el EEG. Cuenta con un amplificador de instrumentación de precisión diferencial y un rango de tensión de salida de ±10 voltios.

En las Figura 18 y Figura 19 podemos observar la parte posterior y anterior de este módulo. En la parte posterior se ve el canal de salida por donde sale la señal analógica, mientras que, en la parte anterior, vemos varios botones que permiten ajustar la señal modificando los rangos de los filtros paso-alto y paso-bajo, asi como la ganancia.

Figura 18. Modulo V75-04 ISOLATED BIOAMPLIFIER WITH BANDPASS FILTER parte posterior. (COULBOURN, n.d.)

Figura 19. Modulo V75-04 ISOLATED BIOAMPLIFIER WITH BANDPASS FILTER parte anterior. (COULBOURN, n.d.)

2.2.2.3. MEDIDOR DE EEG

Este módulo, con nombre *V75-08 4-CHANNEL EEG AMPLIFIER*, cuenta con cuatro amplificadores independientes para la captación de EEG. Cada amplificador cuenta con su propio filtro Butterworth[1] incorporado para toda la banda de EEG de 1 a 45Hz. Está formado por un amplificador de instrumentación de precisión diferencial y un rango de tensión de salida de ±10 voltios.

Como se puede observar la cara anterior de este módulo (Figura 20) cuenta con cuatro canales donde se puede ajustar cada ganancia en un rango de 1k a 50K y, en la parte posterior (Figura 21), podemos ver las cuatro salidas analógicas.

[1] El filtro de Butterworth es uno de los filtros electrónicos básicos, diseñado para producir la respuesta más plana que sea posible hasta la frecuencia de corte.

Figura 20. Modulo V75-08 4-CHANNEL EEG AMPLIFIER parte anterior. (COULBOURN, n.d.)

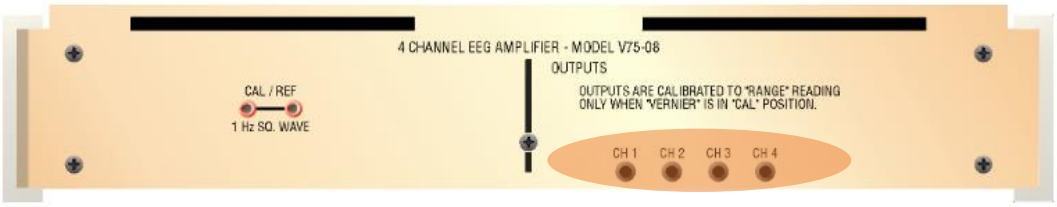


Figura 21. Modulo V75-08 4-CHANNEL EEG AMPLIFIER parte posterior. (COULBOURN, n.d.)

2.3 **Software disponible**

El último componente de nuestro sistema que nos queda por mencionar es el software. El software que se instalará en el ordenador es el encargado de adquirir la información procedente del dispositivo DAQ y transformarla para visualizarla o archivarla. También tiene la capacidad de controlar la función del dispositivo DAQ, indicando con qué parámetros y de qué canales se debe adquirir la información.

Para el desarrollo de este proyecto se ha hecho uso del software LabVIEW de NI del cual hablaremos en los siguientes apartados.

2.3.1 Sistema de desarrollo LabVIEW: Carácterísticas generales

LabVIEW es un sistema de desarrollo creado por NI para implementar sistemas de medición, prueba y control. Probablemente, uno de los aciertos de LabVIEW es que ofrece una programación gráfica, en principio bastante intuitiva para aquellas personas que no sean expertas en programación, como puede ocurrirles a los especialistas en los hardware de medida que deben incluir este tipo de sistemas.

De entre las ventajas de LabVIEW que menciona NI, podemos destacar que:

- Se puede combinar con otros lenguajes como .m, C, Python, and .NET, permitiendo la reutilización de scripts de análisis existentes y algoritmos ya desarrollados.
- Se compone de más de 1000 funciones integradas y diseñadas específicamente para análisis científicos y de ingeniería.
- Permite el procesamiento paralelo de forma natural como parte del lenguaje, debido a su orientación a flujo de datos. Esto es, si se desea que la aplicación realice múltiples tareas a la vez, esto es posible utilizando múltiples bucles paralelos en el diagrama de bloques.

Esta última supuesta ventaja, en la experiencia de la autora de este trabajo, puede volverse en contra del programador novato que, salvo que tenga mucho cuidado, puede programar paralelismos no deseados.

En el siguiente apartado vamos a hablar más detalladamente sobre las características y funciones que tiene LabVIEW. Haremos un repaso sobre el entorno de trabajo y las peculiaridades con las que cuenta, incluyendo las estructuras de programación, las funciones para la adquisición, almacenamiento y procesado de datos.

2.3.2 Sistema de desarrollo LabVIEW: Entorno de trabajo y funciones

Respecto al entorno de trabajo, LabVIEW está formado por dos ventanas: el panel frontal y el diagrama de bloque (Figura 22). El panel frontal corresponde a la interfaz de usuario y el diagrama de bloque al código de acompañamiento gráfico del panel frontal, donde recoge los distintos componentes (terminales del frontal y funciones) conectados entre sí por cables.

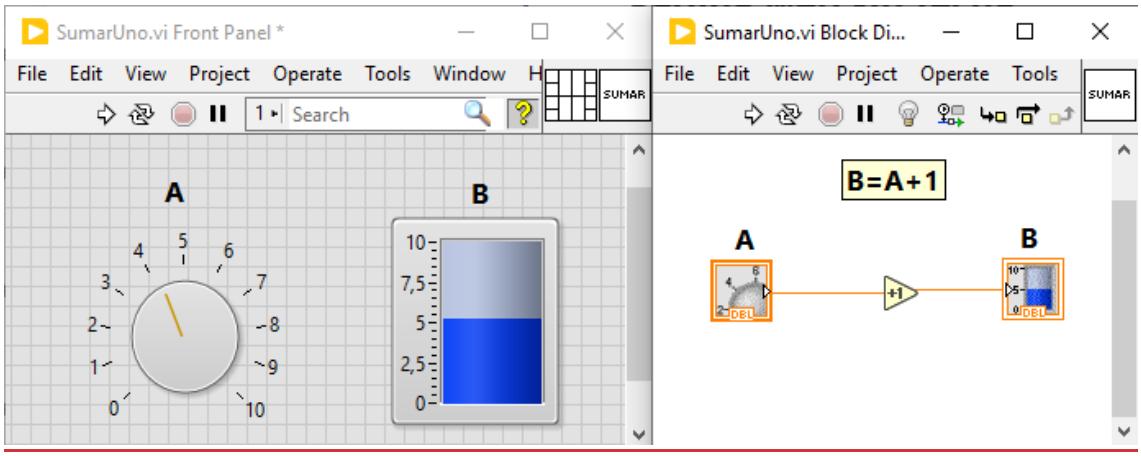


Figura 22. Panel frontal y diagrama de bloques de un programa LabVIEW.

Existen dos tipos de terminales en el panel frontal: los controladores y los indicadores. Todos los controladores e indicadores que estén en el panel frontal están representados como un terminal de conexión en el diagrama de bloques. Un ejemplo de un controlador es un botón y un ejemplo de un indicador una gráfica. Cuando se ejecuta un programa de LabVIEW, conocido como VI por su extensión .vi (Virtual Instrument), los valores de los controladores fluyen por el diagrama de bloques devolviendo el resultado final en los indicadores. Además, LabVIEW permite crear lo conocido como subVIs, que son VI que se crean para usar dentro de otro VI; es el análogo a una función en otros lenguajes de programación. También análogamente a otros lenguajes, se pueden usar dos tipos de variables: las variables locales, que se usan cuando se desea transferir datos entre distintas ubicaciones dentro del mismo VI, y las variables globales que se usan en caso de querer transferir datos entre distintos VI.

En el panel frontal hacemos uso de una paleta de controladores (*Controls*), divido en varias categorías (Figura 23) según el tipo de terminal, de lo que hablaremos en el apartado 2.3.2.1. Gracias a los Property Nodes es posible modificar el aspecto de los objetos que componen el panel frontal. Estos permiten controlar, mediante programación, las propiedades de un objeto como el color, la visibilidad o la posición.

Por otro lado, en el diagrama de bloque, contamos con una paleta de funciones (Functions) dividido en varias categorías según su aplicación (Figura 24). Hablaremos de ellas a partir del apartado 2.3.2.2.

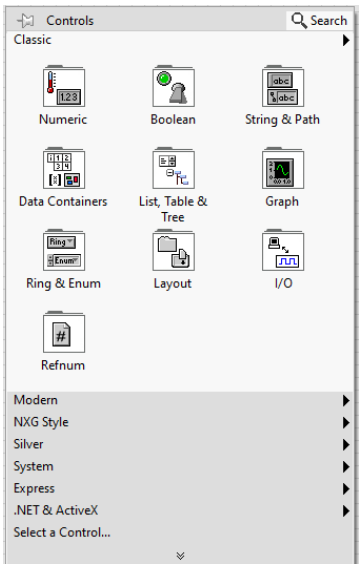


Figura 23. Paleta *Controls*.

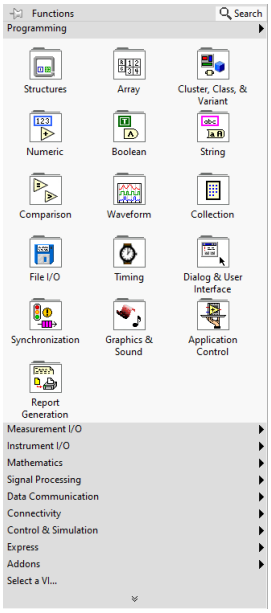


Figura 24. Paleta *Functions*.

A diferencia de otros lenguajes de programación, que siguen un orden secuencial de los elementos del programa, LabVIEW sigue un modelo de flujo de datos donde, para ejecutar un nodo del diagrama de bloque, deben estar disponibles los datos de todos los terminales de entrada. Cuando se ejecuta un nodo, se producen unos datos de salida que van al siguiente nodo. El movimiento de datos a través de los nodos determina el orden de la ejecución de los VI.

2.3.2.1. Panel frontal: Terminales

Como ya hemos comentado, en el panel frontal contamos con una paleta que recoge los distintos tipos de controladores divididos en categorías según el tipo de terminal. Los tipos de terminales que se han usado en esta aplicación son:

- **Numeric**: recoge valores numéricos, ya sean de tipo entero o real.
- **Boolean**: toman valor true/on o false/off. Simulan los interruptores, botones y LEDS.
- **String**: recoge las cadenas de caracteres. Se utiliza para recibir texto del usuario, como por ejemplo el nombre del usuario, de un archivo o una contraseña.
- **Data Container:** arrays y clusters. Un *Cluster* es un tipo de dato formado por la unión de varios elementos de distintos tipos. En la paleta *Functions* existen las funciones Bundle y Unbundle que sirve para escribirlos o leerlos, respectivamente, desde el programa.
- **Graph**: LabVIEW permite la representación gráfica de arrays. Dentro de LabVIEW existen varios tipos de gráficas, siendo las más destacadas las gráficas de tipo Chart y tipo Graph.
    - ***Gráfica de tipo Chart***: muestra datos que se adquieren a un ritmo constante, es decir, guarda y muestra cierto número de puntos almacenados en un buffer. Mantiene un historial de datos, por lo que mantiene los datos ya existentes mientras muestra los nuevos datos recibidos.
    - ***Gráfica de tipo Graph***: muestra los datos recibidos a la vez; no mantiene un historial. Es muy útil para poder ver más detallado la información recibida.
- **Ring & Enum:** se utilizan para crear listas de cadenas. Si los datos que recoge son valores enteros y consecutivos hablaremos de un Enum, en caso contrario, se tratará de un Ring.
- **Layout**: esta categoría recoge elementos que permiten organizar la información y modificar el aspecto del Panel Frontal. Por ejemplo, los Tabs que muestra pestañas separadas en un mismo panel. En esta aplicación se han usado los subpanel, que permiten mostrar en panel de la aplicación principal el panel de un subvi.
- **I/O:** recoge los controladores de E/S que se utilizan junto a las funciones de las librerías de adquisición de datos.
- **Refnum:** se utilizan para trabajar con archivos, directorios, dispositivos y conexiones de red. Un refnum, o número de referencia, es un identificador único de un objeto. Permiten enviar información del objeto del panel a los subVIs. Gracias estos podemos ejecutar a la vez, de forma dinámica, distintos VIs y actuar sobre el aspecto estético de los objetos mediante los Property Node. Hablaremos de esto en el apartado 4.3.3.

2.3.2.2. Diagrama de bloques: Estructuras de programación

Dentro de la paleta *Functions* del diagrama de bloque, aparecen un conjunto de elementos que corresponden a las estructuras de programación. La mayor parte de las que se han usado en este trabajo, que describiremos a continuación, son las que se pueden encontrar en otros lenguajes de programación.

- **Estructura de caso** *(Case Structure)***.** Similar a la estructura condicional *IF-ELSE*. Esta estructura puede estar formado por dos o más subdiagramas (casos). La estructura ejecuta solo un subdiagrama a la vez. El valor de entrada determina qué subdiagrama debe ejecutarse. Como se puede observar en la Figura 25, la estructura tiene una etiqueta en la parte superior donde indica el valor de entrada. En los extremos aparecen dos flechas para poder ver los distintos subdiagramas y una flecha hacia abajo que muestra un menú desplegable que recoge los distintos casos.

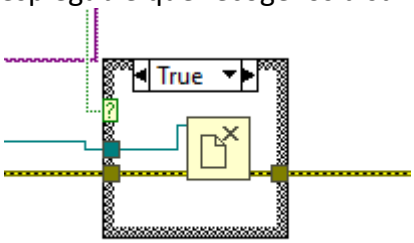


Figura 25. Ejemplo de una estructura de caso.

- **Bucle WHILE** *(While Loop)*. Se utiliza para ejecutar un código hasta que se cumpla una condición determinada. Esta estructura, en comparación con el bucle FOR, se utiliza cuando el objetivo es que se ejecute el código de forma indefinida, hasta que se cumpla una condición y/o usuario indique su detención. En la Figura 26 se observa como estructura principal un bucle WHILE compuesto por una estructura de evento. Este bucle finaliza cuando el programa está en un evento llamado "Salir". La estructura de evento que aparece en la figura se describe a continuación.

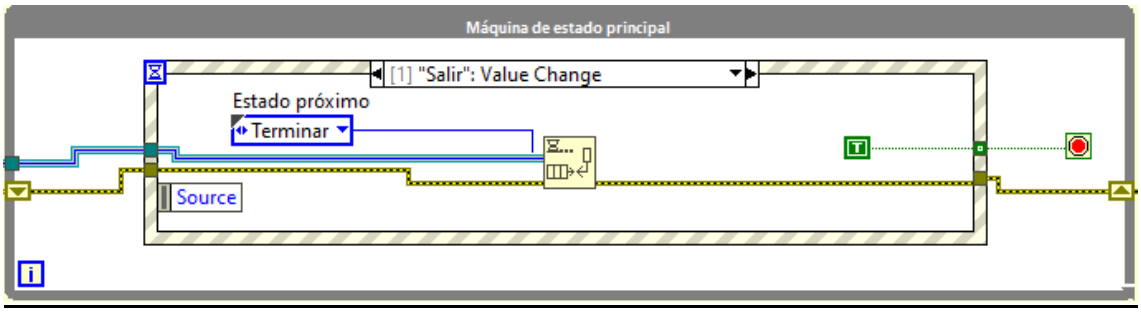


Figura 26. Ejemplo de un bucle WHILE.

En este tipo de estructuras mencionadas es muy importante hacer uso adecuado de los denominados 'Shift Register' (registros de almacenamientos) para poder ir almacenando y actualizando los datos dentro de un bucle de una iteración a la siguiente. En la Figura 26 podemos ver uno marcado en amarillo con un triángulo. De esta manera, el dato que

transporta el cable marrón, siempre se mantiene actualizado durante el bucle WHILE.

- **Estructura de evento** *(Event Structure)***.** LabVIEW permite ejecutar diferentes códigos conforme vayan ocurriendo eventos. Esta estructura se compone de uno o más subdiagramas. Al igual que en la Estructura de Casos, solo se ejecuta a la vez un subdiagrama. En la parte superior tiene una etiqueta donde especifica qué evento hace que se ejecute el bloque de código que contiene.
- **Estructura de secuencia plana** *(Flat Sequence Structure)*. Como se indica en el apartado 2.3.1, a diferencia de otros sistemas de desarrollo, LabVIEW está orientado al flujo de datos, no a instrucciones. Por esto mismo se utiliza esta estructura, para indicar el orden de la ejecución de los bloques de código. En la Figura 27 se observan dos estructuras unidas de manera secuencial que representa un ejemplo de este tipo de elemento.

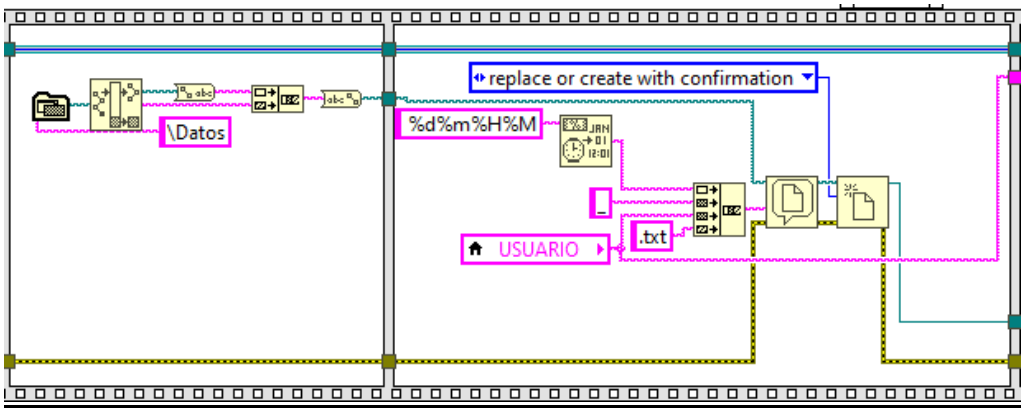


Figura 27. Ejemplo de una estructura de secuencia plana.

- **Colas**. Una cola es una lista con buffer que mantiene un orden de entrada y de salida (FIFO) de los elementos de datos. En LabVIEW, una cola puede ser usada cuando hay varios procesos relacionados dentro de un programa. En la Figura 28 podemos ver recogidas todas las funciones de las colas. Las funciones más destacables son:
  - **Obtain Queue**: crea una cola y devuelve su referencia
  - **Enqueue Element**: añade un elemento en la parte posterior de una cola.
  - **Enqueue Element At Opposite End**: añade un elemento en la parte anterior de una cola.
  - **Dequeue Element:** devuelve un elemento de la parte delantera de una cola.
  - **Flush Queue:** elimina todos los elementos de una cola.

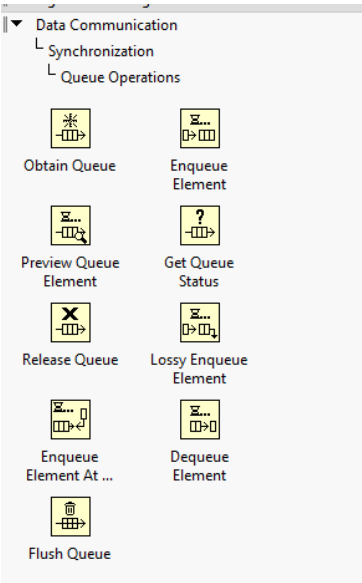


Figura 28 Funciones de las colas

2.3.2.3. Diagrama de bloques: Funciones para la adquisición de datos

En la paleta *Functions* se encuentra una librería enfocada a la adquisición de datos con los dispositivos de NI. Esta librería se llama NI-DAQmx y contiene varias funciones que quedan recogidas en la Figura 29.

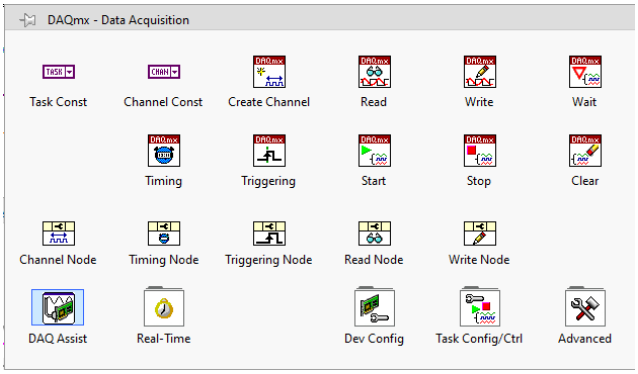


Figura 29 Funciones de la librería DAQmx

Las funciones que se han usado en este trabajo para manejar el dispositivo USB-6000 de NI son:

- **DAQmx Create Channel**: crea los canales de adquisición que se vayan a usar y sus parámetros de uso.
- **DAQmx Start Task** : pone en marcha la ejecución de adquisición o generación de datos.
- **DAQmx Stop Task**: finaliza la adquisición de datos.
- **DAQmx Read**: lee los datos recogidos en uno o más canales de entrada analógica.
- **DAQmx Timming (Sample Clock)**: establece el tiempo de muestreo y el modo de adquisición, esto es si se van a adquirir un número finito de muestras o se van a ir adquiriendo de forma continua hasta que se pare desde programa la adquisición usando la función DAQmx Stop Task.

2.3.2.4. Diagrama de bloques: Funciones para el manejo de ficheros

Aunque LabVIEW tiene la capacidad de asociarse tanto al Excel como al Word, permitiendo crear directamente un documento de alguno de los dos tipos que almacene los datos obtenidos, nuestra aplicación genera un fichero .txt, por lo que se ha trabajado a más bajo nivel. Las principales funciones usadas han sido:

- **Open/Create/Replace File**: esta función abre un archivo existente, crea una nuevo o reemplaza, de forma programada o interactiva utilizando un cuadro de diálogo.
- **Close File:** cierra un archivo abierto especificado en el parámetro de entrada.
- **Write to Text File**: escribe una cadena o una matriz de cadenas como líneas en un archivo.
- **Array to Spreadsheet String**: convierte un array en una cadena. Se puede usar para convertir los datos obtenidos desde la función DAQmx Read en una cadena y así poder almacenarlo en un fichero.
- **Read from Text File**: Lee un número determinado de caracteres o líneas de un archivo.

Podemos decir que existen otras funciones que, de manera indirecta, forman parte de este grupo ya que pueden ayudar en el almacenamiento de datos.

- **Match Pattern**: Busca una expresión especifica en la cadena que se le pasa como parámetro de entrada. Si la función encuentra una coincidencia, divide la cadena en tres subcadena: la anterior a la expresión indicada, la propia expresión y la posterior.
- **Insert Into Array**: introduce un array o subarray dentro de un array de N dimensiones en el punto especificado por el índice. Podemos obtener el índice mediante otra función que se llama *Index Array*.

3

Especificaciones del sistema

Tras el estudio mostrado en el capítulo previo y teniendo en cuenta los objetivos generales planteados en el capítulo de introducción, a continuación se detallan los requisitos de implementación y funcionales de la aplicación objetivo. Debido al carácter docente de dicha aplicación, parte de los requistos funcionales han sido planteados por el profesorado que será el futuro usuario de la aplicación.

3.1 Requisitos de implementación

La aplicación será desarrollada en LabVIEW, y hará uso del Lablinc V equipado con los módulos de PPG (*V71-40)*, Amplificador de biopotenciales (*V75-04*) y Electroencefalografía (*V75-08*), conectado con el módulo de adquisición de datos USB-6000 de NI.

3.2 Casos de uso y Requisitos funcionales

Se muestra en la Figura 30 un diagrama de caso de uso representando las distintas funciones que puede realizar el usuario con nuestra aplicación. El usuario podrá arrancar la adquisición seleccionando previamente los canales deseados y, con ello, representar, guardar y procesar los datos que se vayan adquiriendo; cuando quiera, podrá parar la adquisición. También tiene la posibilidad de leer el fichero obtenido, en esa sesión o en otra anterior. Y, cuando desee terminar con el uso de la aplicación, podrá apagarla.

Esta funcionalidad se detalla en la Tabla 3 que muestra los requisitos funcionales de la aplicación. En la columna P aparece codificada la prioridad en la ejecución de los mismos: 0 representa requisitos de imprescindible cumplimiento para este trabajo y 1, aquellos que se abordarán si el tiempo lo permite, quedando, en caso contrario como líneas futuras del presente trabajo.

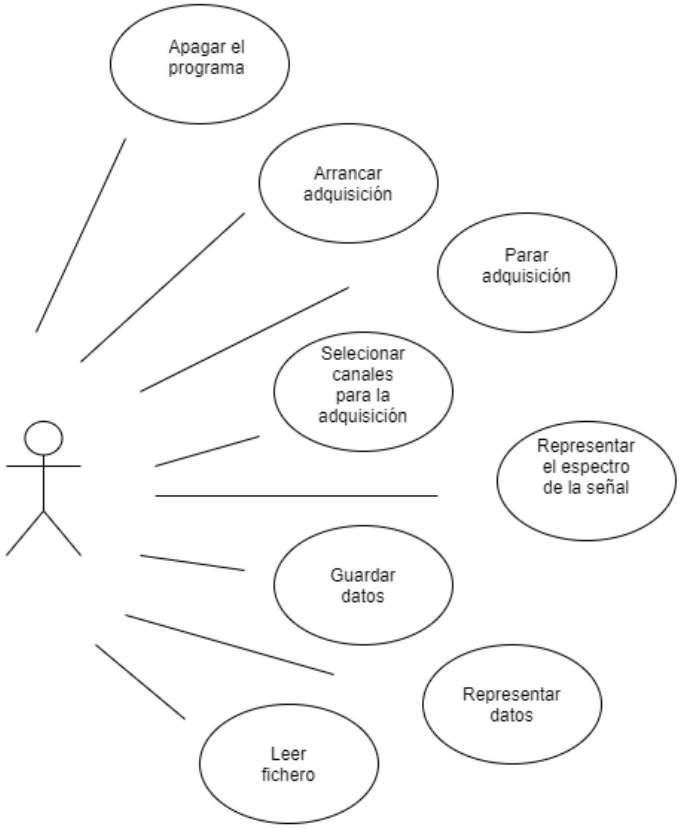


Figura 30. Diagrama de caso de uso.

Tabla 3. Requisitos funcionales de la aplicación.

Número	Nombre Corto	Descripción	P
1	Identificar	La aplicación comenzará pidiendo al usuario que introduzca su nombre y hasta que no realice esta acción no podrá hacer ninguna otra.	0
2	Acción	Una vez que el usuario haya informado de su nombre, la aplicación le permitirá elegir si quiere comenzar una adquisición, si quiere visualizar una adquisición anterior que previamente haya guardado en un fichero o abandonar la aplicación.	0
2.1	Adquirir	Si el usuario así lo indica, el sistema debe adquirir y visualizar simultáneamente hasta 8 señales provinientes del polígrafo.	0
2.1.1	Adq_Conf_Canal	El sistema debe permitir al usuario configurar cuales de los 8 canales disponibles en la USB-6000 serán usados y dar un nombre a cada uno de los canales que ha configurado	0
2.1.2	Adq_Conf_Fm	El sistema debe permitir al usuario configurar la frecuencia a la que se hará el muestreo.	0
2.1.3	Adq_Gráficar	El sitema debe representar las señales aduiridas en dos tipos de gráficas: en una se	0

Número	Nombre Corto	Descripción	P
		representará la ventana temporal de las señales recién adquiridas (instantánea) y en la otra, esa misma ventana junto a un número determinado de ventanas anteriores, esto es, se verá una parte de la historia del registro (historica).	
2.1.3.1	Graf_Tiempo	El sistema permitirá al usuario escoger la longitud en tiempo de ambas gráficas.	0
2.1.3.2	Graf_Visible	El sistema permitirá dejar de visualizar alguna o algunas de las señales representadas.	0
2.1.3.3	Graf_Leyenda	Ambas gráficas tendrán una leyenda que permita identificar a las señales representadas con el nombre que el usuario les dio.	0
2.1.4	Adq_Arr_Para	El sitema permitirá al usuario arrancar y parar la adqusición, permitiéndole ver el tiempo que transcurre entre el arranque y la parada.	0
2.1.5	Adq_Escribir_ Fichero	El sitema permitirá, si el usuario así lo indica, guardar en un fichero los datos adquiridos desde el comienzo hasta la parada de la adqusición.	0
2.1.5.1	Nombre_Fich	El sistema nombrará automáticamente el fichero con la fecha, hora y nombre del usuario.	0
2.1.5.2	Cabecera_Fich	El fichero tendrá en su cabecera: el nombre del usuario, la fecha y hora de arranque de la adqusición, el nombre de los canales adquiridos y la frecuencia de muestreo programada para la adqusición.	0
2.1.6	Adq_Procesado	El sistema permitira procesar en tiempo real las señales adquiridas y visualizar el resultado del procesado.	1
2.1.6.1	Proc_FFT	El sistema realizará la FFT de la señal que escoja el usuario en un tiempo de ventana igual al seleccionado en el requisto 2.1.3 para la gráfica instantánea y lo representará en una gráfica.	1
2.1.6.2	Proc_HR	Si el usuario así lo indica, el sistema calculará la frecuencia cardiaca en base a las muestras de la señal que escoja el usuario.	1
2.2	Recuperar	El sistema permitirá visualizar en una gráfica los valores de adquisiciones previas que hayan sido almacenadas en un fichero.	0
2.2.1	Rec_Fichero	El sistema permitirá al usuario escoger el fichero fuente.	0
2.2.2	Rec_Graf_tiempo	En la gráfica se representaran las señales cargadas del fichero respecto al tiempo.	0

Número	Nombre Corto	Descripción	P
2.2.3	Rec_Graf_Visible	El sistema permitirá dejar de visualizar alguna o algunas de las señales representadas.	0
2.2.4	Rec_Graf_Leyenda	La gráfca tendrán una leyenda que permita identificar a las señales representadas con el nombre almacenado en el fichero.	0
2.3	Salir	La aplicación le pedirá confirmación al usuario cuando desee salir.	
3	Warning	El sistema advertirá al usuario de los siguientes errores.	0
3.1	Err_Conexión	El sistema advertirá al usuario cuando no esté bien conectada la tarjeta de adquisición.	0
3.2	Err_Config	El sistema advertirá al usuario cuando no esté bien configurada la tarjeta de adquisición.	0
3.3	Err_Path	El sistema advertirá al usuario cuando no esté indicada la ruta del path del fichero que quiere leer.	0
3.4	Err_Fichero	El sistema advertirá al usuario cuando seleccione un fichero erróneo.	0
4	Error	La aplicación finalizará si se produce caulquier otro error no contemplado en el requisto 3, advirtiendo al usuario de que ha ocurrido este evento.	0
5	Usabilidad	El sistema irá indicando al usuario que acciones puede tomar en cada momento y mostrando u ocultando en el interfaz de usuario los controles pertinentes para facilitar su correcto uso.	0

3.3 Consideraciones de diseño

En relación con el requisito funcional número cinco indicado en la tabla anterior, Usabilidad, se debe indicar que hemos intentado crear una aplicación que sea de fácil uso programando las acciones que aparecen en la descripción del requisito. Pero, aunque durante todo el desarrollo se ha tenido en cuenta este punto, no podemos confirmar que el requisito se haya cumplido ya que no se han realizado pruebas con usuarios donde quede demostrado.

También se debe decir, que durante el diseño se ha tenido presente la posibilidad de que nuestro programa sea fácilmente ampliable en un futuro, simplificando así la acometida de nuevas líneas futuras de desarrollo. Hemos preferido directamente no incluir este punto como requisito por lo subjetivo de su verificación.

4

Desarrollo del sistema

El presente capítulo pretende describir detalladamente el código desarrollado para la aplicación. El funcionamiento de la misma dependerá de las decisiones del usuario, por lo que debe gestionar eventos asincronos y, por tanto, su diseño se basó en una máquina de estados. En primer lugar, se describirá el interfaz de usario implementado, despúes se muestra el diseño de la máquina de estados realizado y la filosofía de su implementación en LabView, para continuar describiendo los principales estados que constituyen la máquina junto con las funciones (que recordamos que se denominan subVis en LabVIEW) que se han desarrollado para cada uno de ellos.

4.1 **Descripción del interfaz de usuario.**

Aunque la funcionalidad del interfaz de usuario se encuentra descrita con detalle en el apartado correspondiente de este documento (Manual de usuario), aquí haremos también una breve descripción de su funcionalidad para contextualizar los tipos de terminales usados y las acciones que sobre ellos hace el programa para facilitar el uso de la aplicación (gracias a los property nodes, explicados en el apartado 2.3.2., los terminales apareceran habilitados o deshabilitados según el estado de ejecución en el que nos encontremos).

En la Figura 31 podemos ver la interfaz del usuario o, lo que es lo mismo, lo que en LabVIEW se denomina el panel frontal. Cuenta con cuatro botones, tres gráficas, un reloj y dos espacios reservados para la configuración pervia de las posibles acciones, uno para la adquisición de datos y otro para la lectura del fichero. Para facilitar el uso de la aplicación, se ha añadio también al interfaz un panel informativo que va indicando las distintas acciones a realizar junto a advertencias al usuario.

La Figura 31 en concreto corresponde a un momento de la ejecución en el que la aplicación está esperando que el usuario indique que acción quiere realizar. Podemos

apreciar que las gráficas, el botón de parar adquisición y los valores máximo y mínimo, se encuentran deshabilitados. Como hemos mencionado en el apartado 2.2.1.2, hemos estado trabajando con el modelo USB-6000. Este modelo no cuenta con una ganancia programable por lo que no se puede modificar el valor máximo y minimo. Los terminales máximo y minimo quedan representado con un valor de ±10V para informar al usuario del rango que tiene el modelo que usamos, pero no serán habilitados en ningún momento.

Por último, el panel tiene también dos terminales de tipo subpanel, que no se aprecian en la Figura 31 porque son invisibles hasta que no son usados, pero que se mencionarán en lo que sigue.

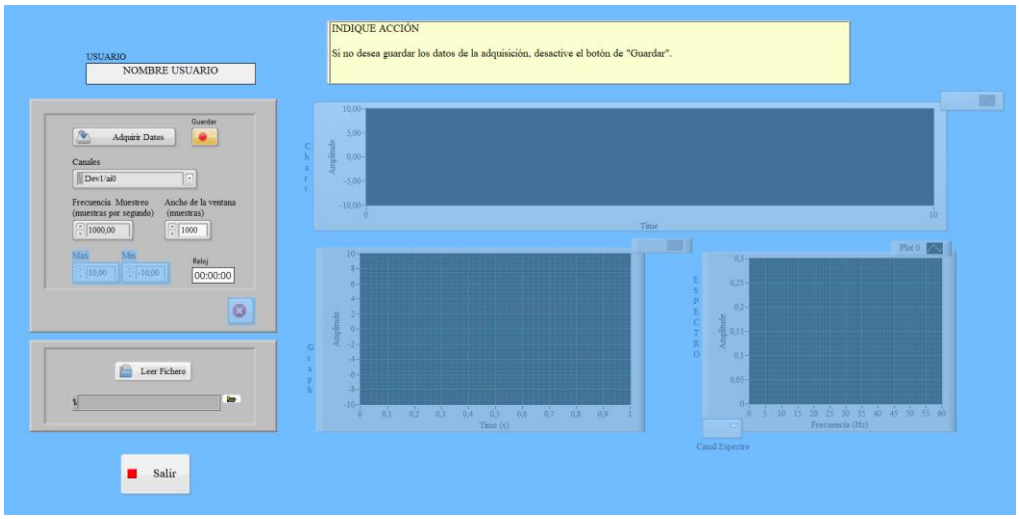


Figura 31. Panel frontal (Interfaz del usuario)

En caso de querer iniciar la adquisición de datos, el usuario deberá configurar los valores de la tarjeta desde el panel frontal y pulsar el botón de "Adquirir datos". El usuario también podrá tomar la decisión de guardar y/o representar los datos dependiendo de si se encuentra activo o no el botón de guardar. Una vez se accione el botón de adquirir, y se compruebe que la configuración del dispositivos es correcta (canales seleccionados y frecuencia de muestreo), se permitirá al usuario dar un nombre a los canales que se van a adquirir, mostrando un subpanel (Figura 32). Una vez que el usuario valide los nombres, se deshabilitarán los terminales de configuración y se podrá ver en tiempo real los datos de las señales representados en tres gráficas. Una es de tipo chart, que representará la ventana actual de adquisición junto con las ventanas adquiridas anteriormente, hasta completar diez kilo muestras, y la otra de tipo graph, que mostrará los valores con más detalle ya que representará solo la ventana actual de datos adquiridos; la terecera gráfica, también de tipo graph, muestra el espectro de la señal de la ventana actual. Como se habrá iniciado la adquisición, se habilitará el botón correspondiente a la orden de parar. Cuando el usuario pare la adquisición, la aplicación volverá al estado de ejecución en la que se espera que el usuario indique una acción.

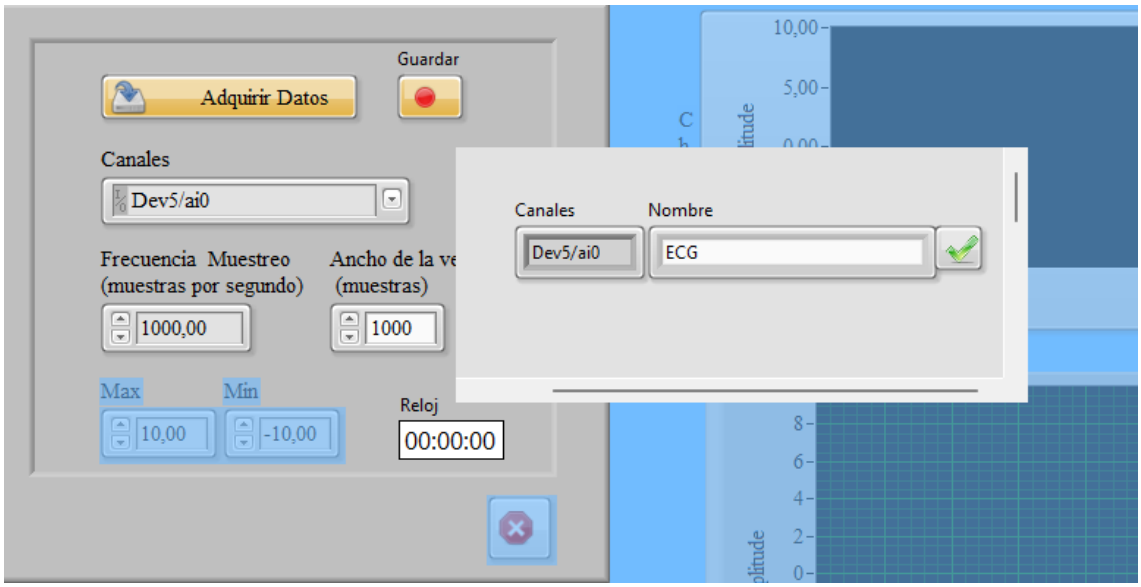


Figura 32. Subpanel de los canales de adquisición.

Si el usuario desea leer un archivo, deberá introducir la ruta de un archivo con el formato correcto para continuar, si la ruta y el archivo son correctos, los datos leidos serán mostrados en un subpanel que se deberá cerrar para continuar (Figura 33).

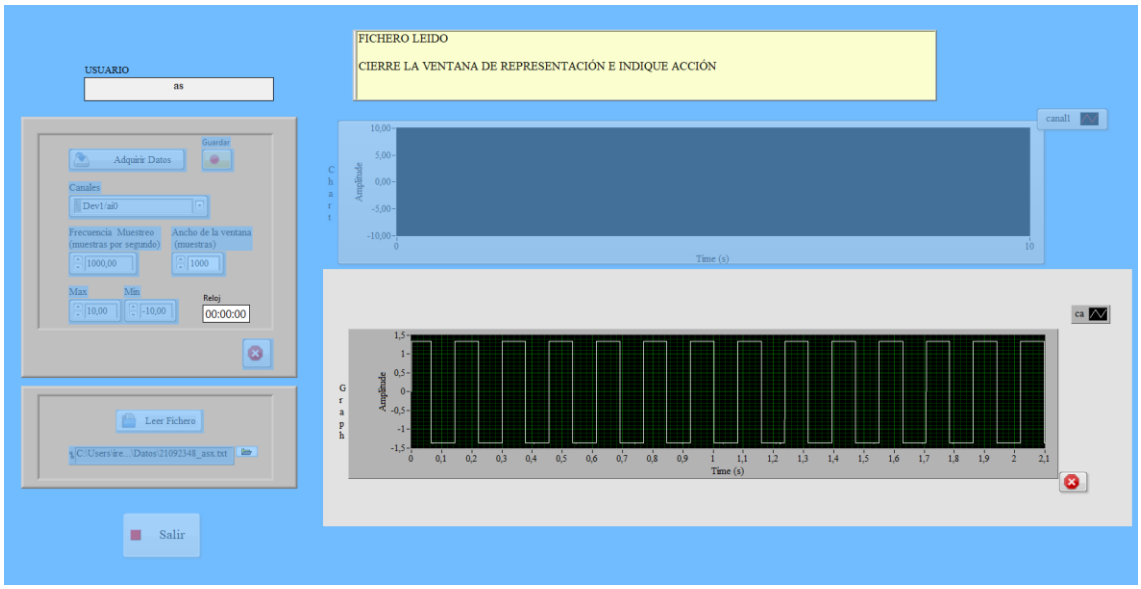


Figura 33. Subpanel de la gráfica de lectura.

Estas acciones mencionadas se podrán repetir tantas veces como el usuario desee. Por último, cuando el usuario accione el botón salir, se lanzará un popup para confirmar la decisión y apagar el programa. En caso contrario, volveremos al de ejecución en que se espera una elección de usuario.

4.2 **Diseño e implementación en LabView de la máquina de estado**

Antes de comenzar a desarrollar nuestra aplicación, creamos un diagrama de estado para plantear los distintos comportamientos y funcionalidades que queríamos desarrollar. Este diagrama fue cambiando durante el desarrollo, quedando finalmente como se muestra en la Figura 34.

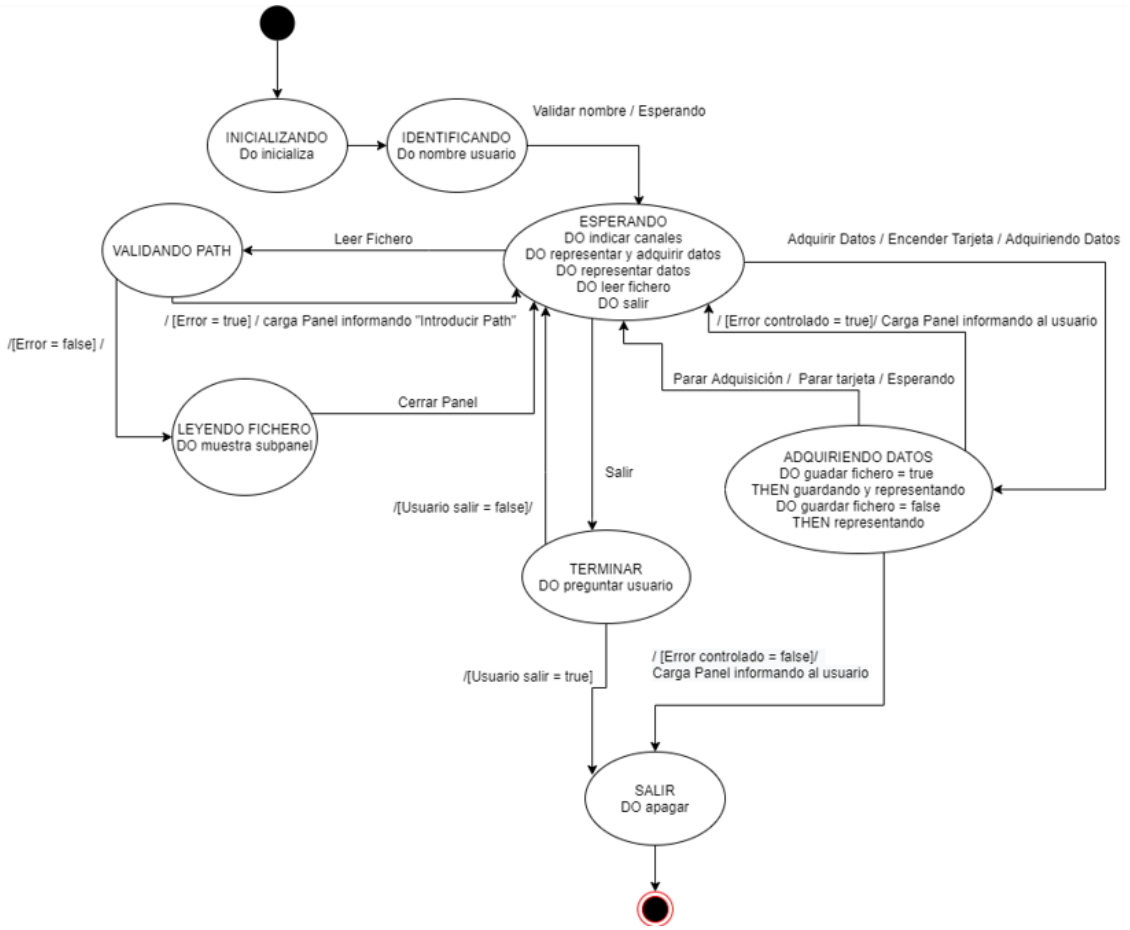
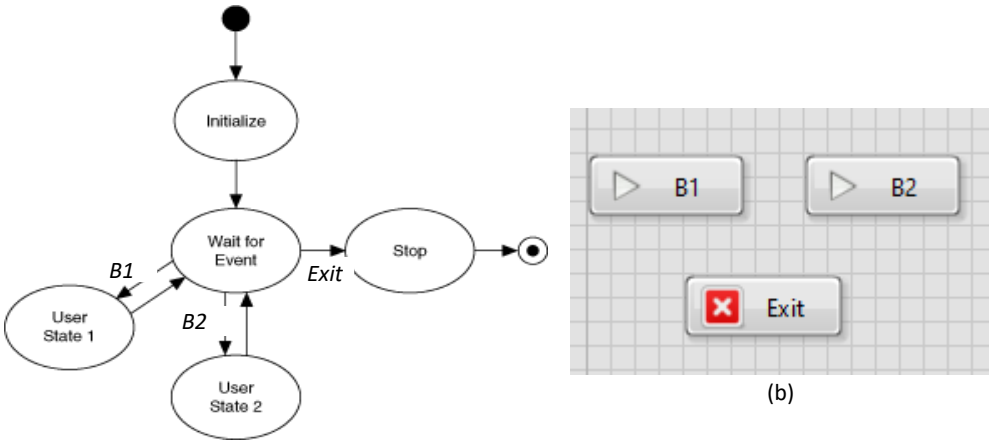


Figura 34. Diagrama de estado.

Antes de describir la implementación realizada, ilustraremos con un ejemplo sencillo la forma de realizar en LabVIEW una máquina de estados.

La implementación del diagrama de estados de la Figura 35. a se muestra, parcialmente, en la Figura 36. Como se puede ver, (Figura 36.a, estado "User State 1"-), se trata de un bucle while que contine una estructura *case*, cuyo selector es alimentado con el estado al que la máquina debe transitar ("Wait for event") usando un *shift register* (señalado con un 1 en la figura). Las posibles variables que cada estado puede escribir o leer, son realimentadas entre los estados con otro *shift register* (señalado con un 2 en la figura). La gestión de eventos asíncronos que provienen del interfaz de usuario (Figura 35.b), la realiza el estado "Wait for Event" (Figura 36.b) usando una estructura de eventos a la que se le ha definido un *frame* a cada uno de los tres botones del interfaz. En la figura se muestra el frame asignado al botón "B1". La máquina se detendrá cuando el usuario pulse el botón "Exit" que la llevará al estado correspondiente (Figura 36.c), en el que el control de salida del bucle while (señalado con un 3 en la figura) se pondrá a *true*.



(a) (b)

Figura 35. Ejemplo de diagrama de estados sencillo (a) y panel de control asociado (b)

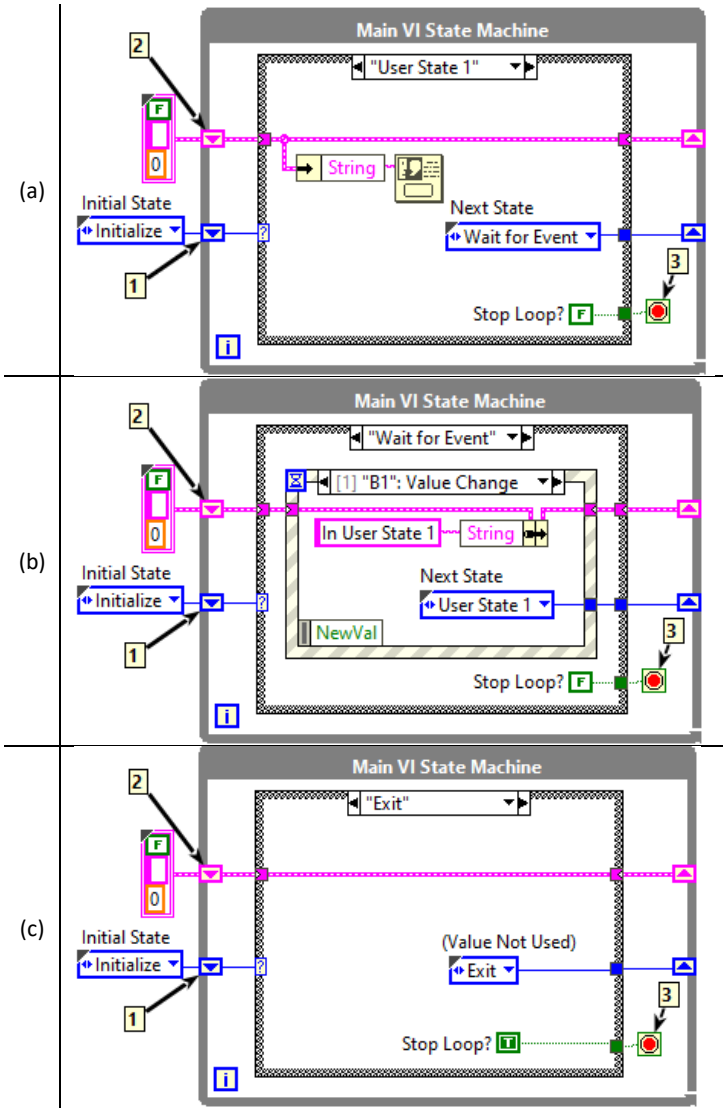
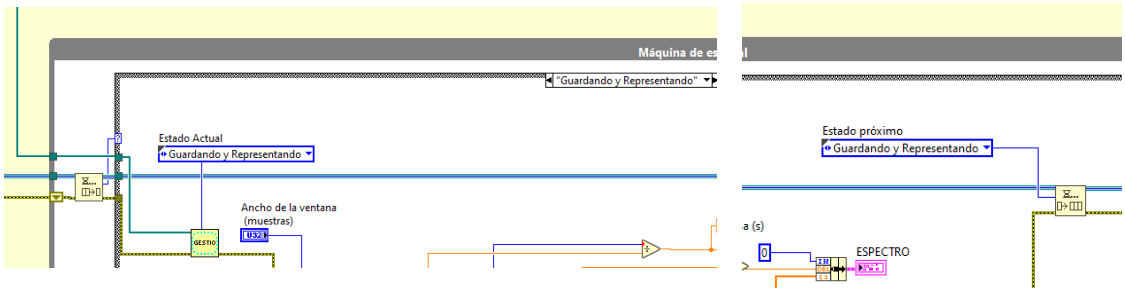


Figura 36. Ejemplo de Implementación de una máquina de estados sencilla.

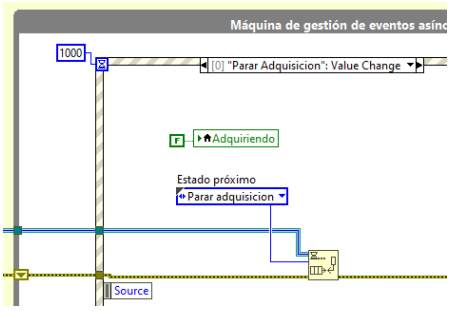
El diagrama de estados del ejemplo es muy sencillo y no contempla que un estado se llame a sí mismo hasta que el usuario decida, desde el panel frontal, interrumpir este

flujo. La implementación mostrada, en la que todos los botones son gestionados por un único estado mediante la estructura de eventos ya no es aplicable en este caso, ya que solo en ese estado serían monitorizados (y no durante la ejecución rentrante del estado a interrumpir). Una posible solución a este problema, es añadir al código otro bucle while que, ejecutando en paralelo a la máquina de estados, monitorice este tipo de botones que deben cambiar el flujo de ejecución de la máquina cuando sean pulsados. Con esa solución, los estados no son pasados directamente mediante el shift register, como en el ejemplo, si no que son almacenados y leidos de una cola que comprarten los dos while. Así, el while auxiliar podrá cambiar cuando se produzca cualquiera de los eventos que monitoriza, el rumbo de la ejecución de la máquina principal, sin más que cargar en la parte superior de la cola el estado al que la máquina debe transitar.

La implementación en LabVIEW del diagrama lógico de la Figura 34 se realizó con esta filosofía de dos bucles while, que hemos denominado Máquina de estados principal y de gestión de eventos asíncronos, que ejecutan en paralelo y que se comunican entre sí gracias al uso de una cola. En la Figura 37 se muestra un detalle del código donde se pueden ver los dos bucles, en (a) aparece parte del código de un estado de la máquina de estados principal y en (b) el bucle de gestión de eventos asíncronos que monitoriza el botón que interrumpiría este flujo.



(a)



(b)

Figura 37. Uso de cola para comunicación entre los dos bucles de la aplicación.

La comunicación y el paso entre los distintos estados es posible gracias a las funciones nombradas en el apartado 2.3.2.2 relacionadas con las colas y con las estructuras de eventos.

Los once estados definidos para la máquina principal se muestran en la Figura 38.

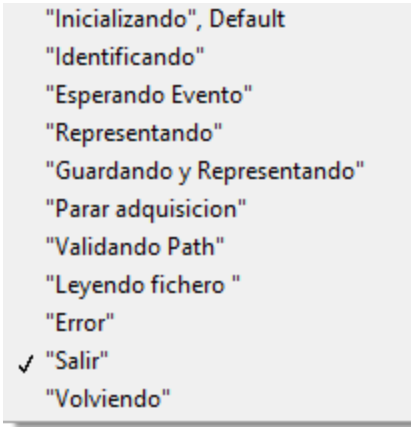


Figura 38. Estados de la máquina principal.

En la Figura 39 podemos ver un ejemplo de un caso de la máquina de estado principal. Como ya se ha dicho, se compone de un bucle WHILE con una estructura de once casos que corresponden a cada uno de los estdos implementados. En la figura se muestra el estado "Identificando" de esta máquina de estado, cuyo funcionamiento consiste en validar el nombre del usuario al comienzo del programa para darle comienzo a nuestra aplicación.

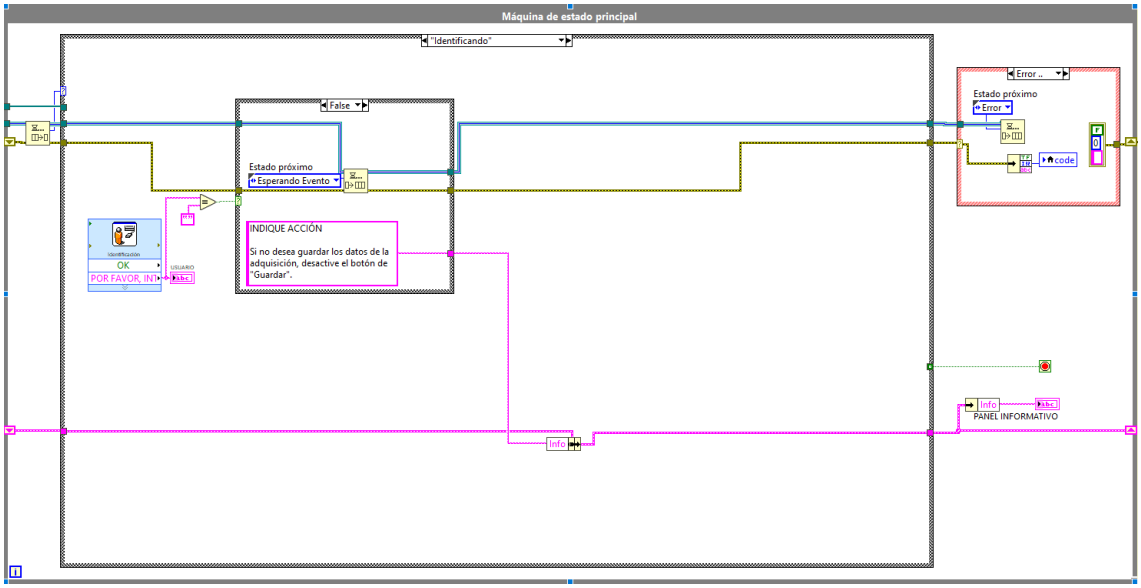


Figura 39. Máquina de estado principal.

La segunda máquina de estado, encargada de la gestión de eventos asíncronos, queda representada en la Figura 40. Esta máquina de estado está compuesta por una estructura de eventos dentro de un bucle WHILE. Recoge los eventos encargados de gestionar la parada de la adquisición de datos (una vez iniciada) o la salida del programa. En la figura queda plasmado el evento "Salir" que se activará una vez el usuario pulse el botón de "Salir" en la interfaz de usuario. Cuando esto suceda, se mostrará un popup donde se preguntará al usuario si está seguro de la decisión y, de ser así se apagará el programa.

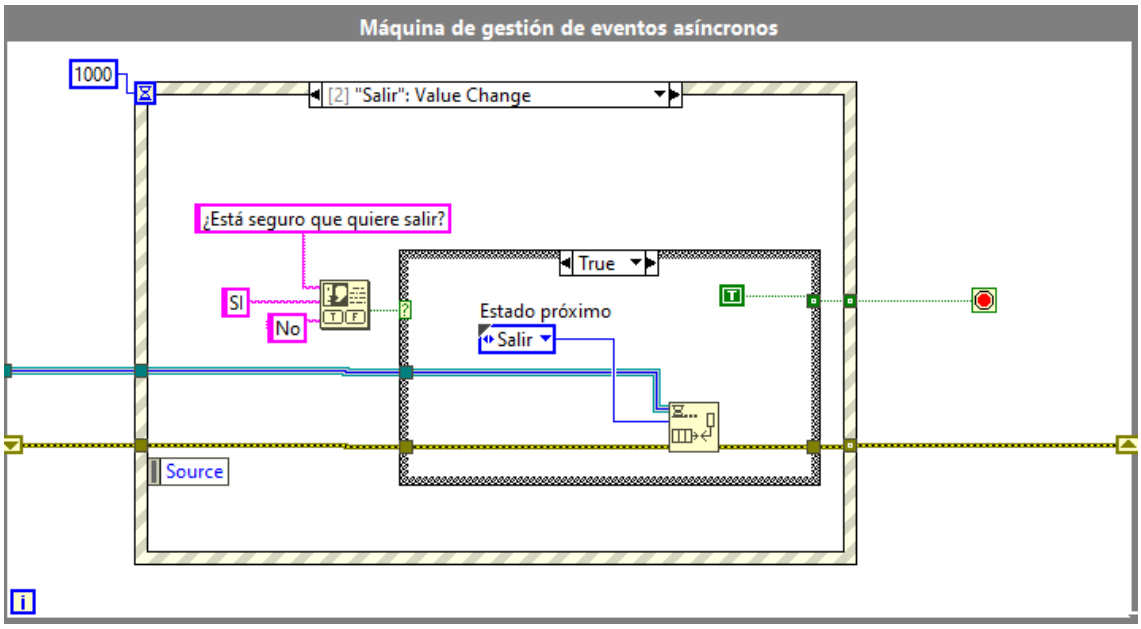


Figura 40. Máquina de gestión de eventos asíncronos.

En el siguiente apartado vamos a hablar más detalladamente de los estados implementados en la máquina principal y de las funciones (subVIs) que se han implementados para cada uno de ellos.

4.3 Descripción de los principales estados

Siguiendo el diseño realizado (Figura 34), una vez arranque nuestro programa, nos encontraremos en el estado "Inicializando" donde se inicializarizarán nuestros componentes y pasaremos al estado "Identificando" donde se deberá introducir el nombre del usuario. Hasta que el nombre no se introduzca no podremos pasar al siguiente estado. Este nombre quedará recogido en una variable y se incluirá en la cabecera del fichero de los datos adquiridos. A continuación, pasaremos al estado "Esperando" donde el usuario podrá tomar distintas decisiones, transitando a diferentes estados. En los siguientes apartados indicaremos los estados relacionados y los principales subVIs incluidos en la adquisición, visualizacion y procesado de la señal, en la lectura del fichero y en la gestión de errores.

4.3.1 Adquisición, visualización y procesado de la señal

Estados relacionados: Esperando, Guardando y Representando, Representando, Parando adquisición.
Principales subVIs: Lectura_Tarjeta, NombreCanales, CabeceraNombre, AdecuacionGrafica, Espectro.

Cuando el usuario desee adquirir datos (pulse el botón de "Adquirir datos") pasamos al caso "Adquirir Datos" dentro del estado "Esperando". Allí nos encontramos el primer subVI llamado "Lectura_Tarjeta", que se encarga de inicializar los datos de entrada tarjeta de adquisición. Este subVI cuenta con cinco parámetros de entrada y tres de salida como podemos ver en la Figura 41. Los parámetros de entrada hacen referencia a la frecuencia de muestreo y canal/es que haya indicado el usuario en el panel frontal

(Figura 42). Como ya hemos indicado, el valor máximo y mínimo viene predefinido con el valor ±10 correspondiente a los valores del modelo USB-600.

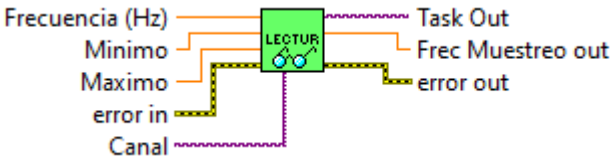


Figura 41. SubVI Lectura_Tarjeta

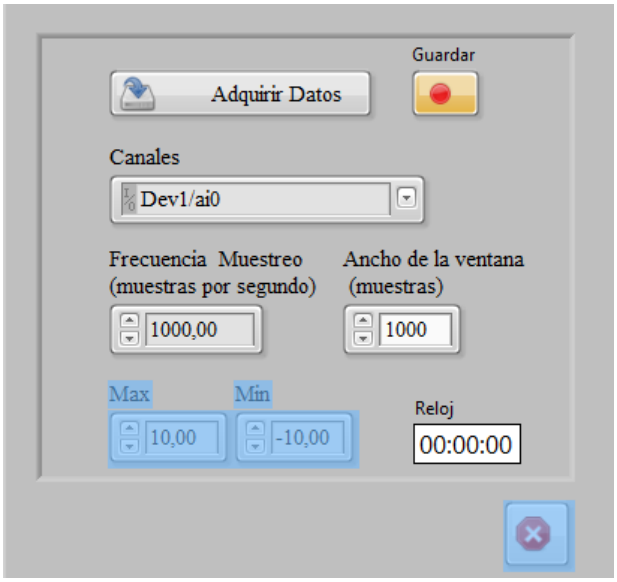


Figura 42. Sección del panel frontal destinado a la configuración de la tarjeta DAQ.

En la Figura 43 podemos ver de forma más detallada este subVI. La primera función que se ejecuta es *DAQmx Create Chanel* que recoge el canal, o canales, definido por el usuario en el panel frontal para identificar de dónde va a venir la señal. En esta función queda indicado que la entrada va a ser una señal analógica con valores de voltaje y un sistema de medición de un solo extremo (RSE), es decir, se va a medir la tensión con respecto a la tierra. Le procede la función *DAQmx Timming* que, junto al valor de la frecuencia de muestreo, queda especificado que la adquisición va a ser modo continuo (*Continuos Samples*). Es muy importante tener en cuenta las características de nuestro dispositivo y los números de muestras que se configuren ya que cuanto mayor sea el número de muestras, más lenta será la adquisición de toda la ventana y la representación de la señal. Previendo los errores asociados al arranque de la tarjeta, incluimos las funciones *DAQmx Start Task y DAQmx Stop Task* en este subVI para que, en caso de que se produjese algún error, no permitiese al usuario continuar con la asignación de nombre de los canales y se pasase directamente al estado de error.

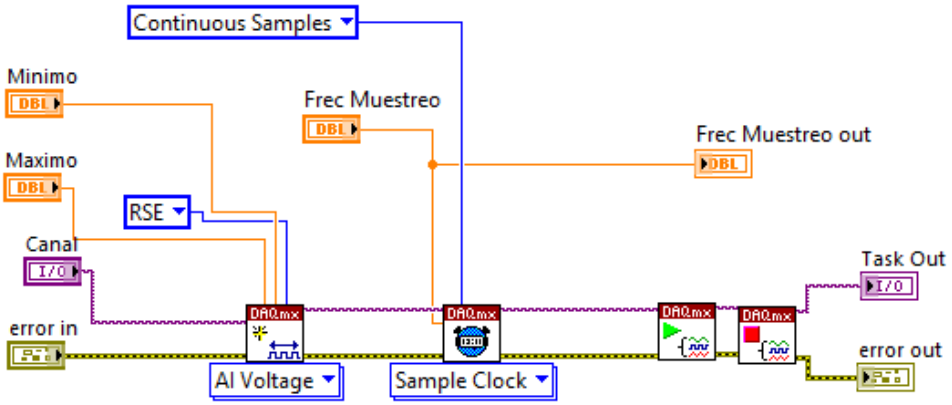


Figura 43. Funciones del subVI Lectura_Tarjeta.

Una vez inicializada la tarjeta, donde hemos indicado los canales que vamos a adquirir, se llama a un segundo subVI (NombreCanales) que permite al usuario indicar el nombre a cada canal para identificar la señal que se está tomando en cada uno de ellos (Figura 44). Este subVI devuelve un set de datos que recoge los canales configurados, es decir, cada canal con el nombre asociado.

Figura 44. SubVI NombreCanales

Gracias a la función Invoke Node *(a la que se le pasa un VI Reference de este subvi)* llamamos e insertamos este subVI en un subpanel que se muestra justo cuando el usuario le da a la acción de adquirir datos. Se mostrará tantas filas como canales estén indicados para completar con el nombre de la señal que corresponde cada uno. En la Figura 45 podemos ver un ejemplo de este comportamiento.

Figura 45. Función del subVI NombreCanales

A continuación, el subVI "AdecuacionGráfica" modifica la leyenda de las tres gráficas en función de los canales que se van a representar y de sus nombres, introducidos en subpanel de la figura anterior. En la Figura 46 podemos ver que recibe como parámetros de entrada las referencias de la gráfica Chart, Graph y el canal de Espectro para añadir los nombres de los distintos canales seleccionados indicados en el parámetro de entrada

"Canales Config". También toma la frecuencia de muestreo para autoescalar el ejeX de la gráfica Chart. Este subVI limite el número de canales a ocho ya que es el número de canales con el que cuenta nuestra tarjeta de adquisición.

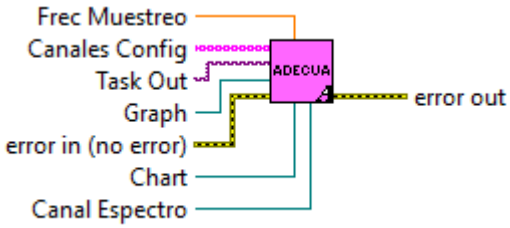


Figura 46. SubVI AdecuacionGrafica

Una vez configurada la tarjeta de adquisición y preparada las gráficas, nos podemos encontrar con dos escenarios: el escenario donde el usuario desee crear un fichero para guardar los datos de la adquisición o el escenario que solo quiera representar los datos en la gráfica, dependiendo del estado en el que se encuentre en botón de "Guardar". En caso de querer guardar los datos se creará un fichero en una carpeta destino predefinida con una cabecera especifica gracias al subVI "CabeceraNombre". La cabecera estará compuesta por el nombre de los canales, la frecuencia de muestreo y el nombre del usuario que recibe el subVI como parámetro de entrada. Sobre este subVI hablaremos de forma más detallada en el apartado 4.3.1.1.

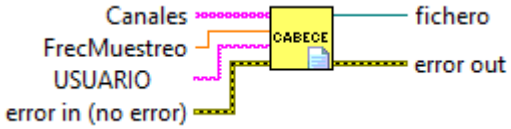


Figura 47. SubVI CabeceraNombre.

Realizadas todas estas acciones sin error, la aplicación arranca la tarjeta mediante la función *DAQmx Start Task*. A partir de ese momento, los datos quedarán representados en las gráficas gracias a la función *DAQmx Read* y al registro de desplazamiento que contiene los datos de adquisición. Esta función seguirá activa hasta que el usuario accione el botón de parar. En cuanto esto suceda, se activará la máquina de estado de gestión de eventos asíncronos y hará uso de la función *Enqueue Element At Opposite End* para cambiar al estado de "Parar adquisición", donde nos encontraremos con la función *DAQmx Stop Task,* la encargada de para la adquisición por parte de la tarjeta. En la Figura 48 podemos ver el flujo de este proceso de adquisición de datos mediante la librería DAQmx.

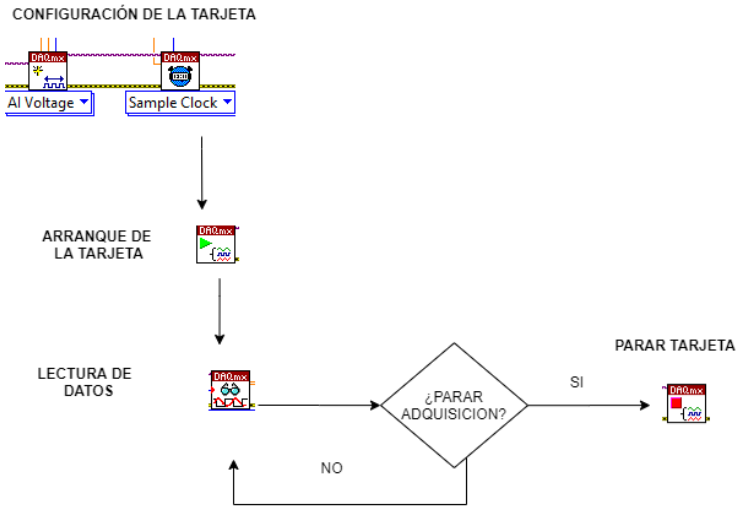


Figura 48. Flujo de la adquisición de datos mediante las funciones de la librería DAQmx.

4.3.1.1. Almacenamiento

Para el almacenamiento de los datos hemos hecho uso de las funciones mencionadas en el apartado 2.3.2.4.

Como hemos comentado en el apartado anterior, existe dos escenarios tras la adquisición: podemos representar y guardar datos en un fichero, donde pasaríamos al estado "Guardando y Representando", o solo representar, que se trataría del estado "Representando".

En caso de querer guardar los datos en un fichero, pasaríamos al estado "Guardando y Representando" donde se encuentra la función Open*/Create/Replace File* encargada de crear este fichero. Gracias al subVI "CabeceraNombre", mencionado en el apartado anterior, el fichero se creará con una cabecera compuesta por el nombre del usuario, la fecha de la adquisición, el nombre de los distintos canales, la frecuencia de muestreo y la palabra "DATOS:", la concatenación de estos campos es posible gracias a la función *Concatenate String*. Además, mediante las funciones *Applications Directory, Strip Path y String to path*, definiremos la carpeta destino de los ficheros creado por este programa. En la Figura 49 podemos ver el diagrama de bloque, donde se encuentran las funciones mencionadas junto a la función *Write to Text File* que se encarga en escribir la cabecera al comienzo del fichero.

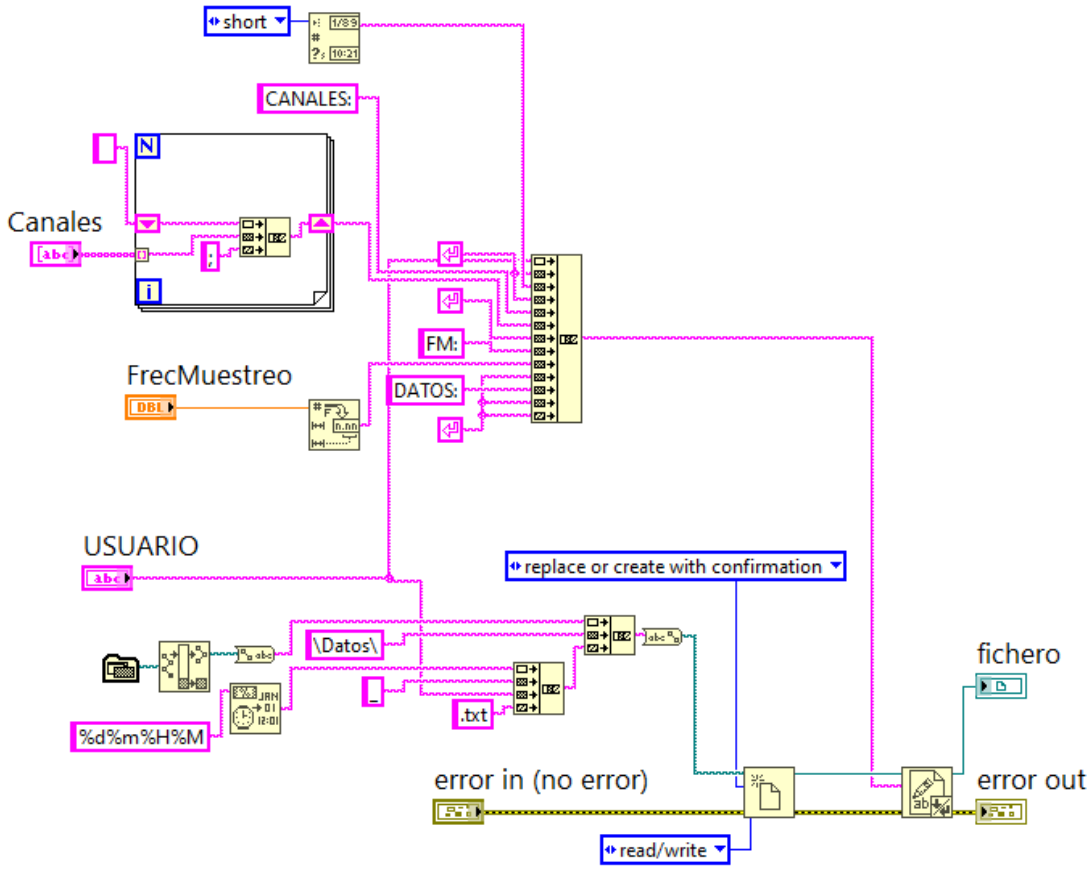


Figura 49. Preparación del fichero.

Una vez preparado el fichero con su cabecera, pasamos al estado "*Guardando y Representando*" donde gracias a la función *Set File Position* seguida de la función *Write To Text File*, se empiezan a escribir los datos en el fichero. Es muy importante hacer uso de la función Set File Position para indicar al programa dónde posicionarse cada vez que quiera añadir una ventana de datos nueva, ya que el fichero ira guardando los datos de forma sucesiva cada vez que entre en el estado "*Guardando y Representando".*

En la Figura 50 podemos ver un ejemplo de un fichero. Como se observa se creará una columna por cada canal que tengamos conectado junto a la cabecera definida mediante el subVI "CabeceraNombre".

```
USUARIO
13/09/2022
CANALES:PPG;ECG;EEG;
FM:1000,000000
DATOS:

-2,064576    1,063666    -0,021154
-2,164644    1,105796    -0,015887
-2,259444    1,142661    -0,000086
```

Figura 50. Fichero de almacenamiento.

Una vez el usuario pulse el botón de "Parar adquisición", pasaremos al caso "Parar adquisición" donde está la función *Close File* para cerrar el fichero. Como hemos mencionado, no siempre se creará un fichero cuando se inicia la adquisición, por lo que esta función no siempre deberá ejecutarse. Esto dependerá del valor del botón guardar. Para ello hacemos uso de una estructura de caso donde, mediante una variable local, distinguimos las dos casuísticas según el valor del botón. Estas dos casuísticas las podemos ver en la Figura 51.

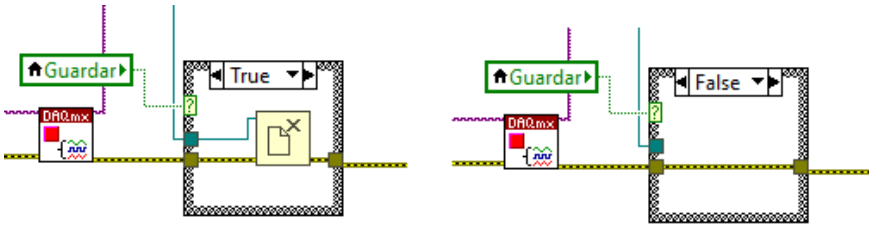


Figura 51. Casuísticas del cierre del fichero.

4.3.1.2. Procesado de los datos adquiridos

Para el procesado de la señal contamos con un subVI llamado "Espectro" encargado de calcular el espectro en tiempo real durante la adquisición gracias a la función "Power Spectrum". El resultado queda representado en la tercera gráfica, de tipo Graph, de nuestra interfaz de usuario. Esto se ha añadido a petición de los tutores ya que facilita la detención en tiempo real de cualquier interferencia que pueda producirse durante la adquisición. Este subVI se encuentra tanto en el estado de "Guardando y Representando" como en el de "Representando".

En la Figura 52 podemos ver que este subVI recibe como parámetros de entrada el número de muestra indicado por el usuario, los datos adquiridos y el canal espectro que hace referencia a los distintos caneles que pueden estar activos.

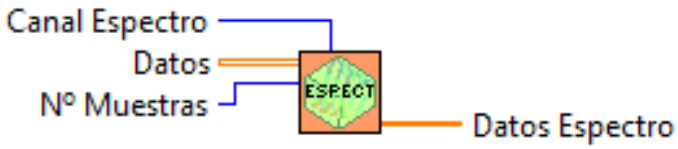


Figura 52. SubVI Espectro.

4.3.2 Lectura del fichero y visualización de los datos

Estados relacionados: Validando Path, Leyendo Fichero
Principales subVIs: LeerCabeceraDatos, VentanaLeerArchivo

A la hora de leer un fichero primero pasamos al estado "Validando Path" donde se comprueba que el usuario ha introducido un fichero en la ruta path que hay reservado para ello. En caso de que pulse el botón de "Leer fichero" pero no haya añadido la ruta path, saltará un aviso informando de esto. Una vez valide la ruta path pasaremos al estado "Leyendo Fichero" donde se procesará.

Para la lectura del fichero hay que tener en cuenta que nuestros ficheros están formados por la cabecera que termina con la palabra "Datos:", mencionada en el apartado anterior, por lo que, a la hora de leer los datos debemos hacer uso de la función *Match Pattern* para determinar dónde termina la cabecera y dónde empiezan los datos. Además, en caso de que el fichero recoja datos de más de un canal, debemos guardarlo en un parámetro (en este caso "canales" como parámetro de salida, Figura 53) para representarlo en la gráfica de forma independiente. Esto queda recogido en el subVI LeerCabeceraDatos (Figura 53) que recibe por un lado la cabecera y por otro los datos numéricos como parámetros de entrada, ya que se deben tratar de forma distinta.

En caso de que el fichero venga vacío o tenga otro formato se avisará al usuario mediante el panel informativo.

Figura 53. SubVI LeerCabeceraDatos.

Al inicio de este bloque comentamos que existe una cuarta gráfica que se representara en un terminal de tipo subpanel que se encuentra oculto al arranque de la aplicación. El subpanel se mostrará solo cuando el usuario pulse el botón de "Leer fichero". Existe un subVI llamado "VentanaLeerArchivo" que contiene esta gráfica. La gestión se hace de forma análoga a la ya comentada para el subpanel que permite nombrar los canales al usuario.

Como podemos ver en la Figura 54, este subVI recibe como parámetroo de entrada los datos y canales correspondiente a los parámetros de salida del subVI anterior. Con estos parámetros, el subVI genera los datos de la señal recogida en cada canal. En la Figura 55 podemos ver el interior de este subVI que esta formado por un bucle WHILE que trata cada valor recibido en el parámetro de entrada "canales" para construir la leyenda de la gráfica y también de un terminal, representando como un botón en el panel frontal, encargado de cerrar el subpanel.

Figura 54.SubVI VentanaLeerArchivo.

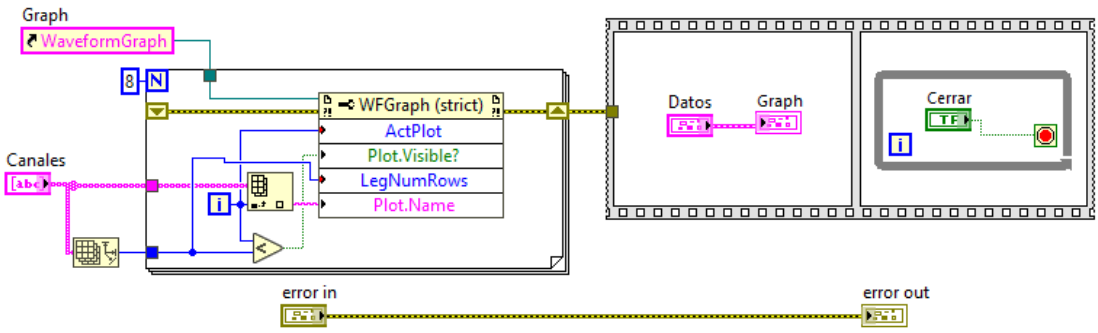


Figura 55. Diagrama de bloque del SubVI VentanaLeerArchivo.

4.3.3 Gestión de errores

Uno de los once casos con los que cuenta la máquina de estado principal es el caso "Error". Si durante el uso de nuestra aplicación se produce un error, pasaremos a este caso donde se han tratados los errores detectados durante el desarrollo del programa. En caso de que se produzca un error no controlado, el panel informativo cambiara a rojo con un mensaje de error parpadeante (Figura 56) y, tras cinco segúndos, el programa se apagará.

Figura 56. Panel informativo cuando se produce un error no controlado.

En la Figura 57 podemos ver el diagrama de bloque que compone este caso. Se muestra la casuística en la que el código de error es "-200077", este código hace referencia al error producido cuando la tarjeta no está conectada al PC o cuando la frecuencia de muestreo indicada por el usuario supera la máxima de la tarjeta. En la Figura 58 se observa la casuística del error no controlado, donde se ha hecho uso de los property nodes (para cambiar el comportamiento del panel informativo) y y la función *Waits(ms)*, para informar durante cinco segundos antes de apagar el programa.

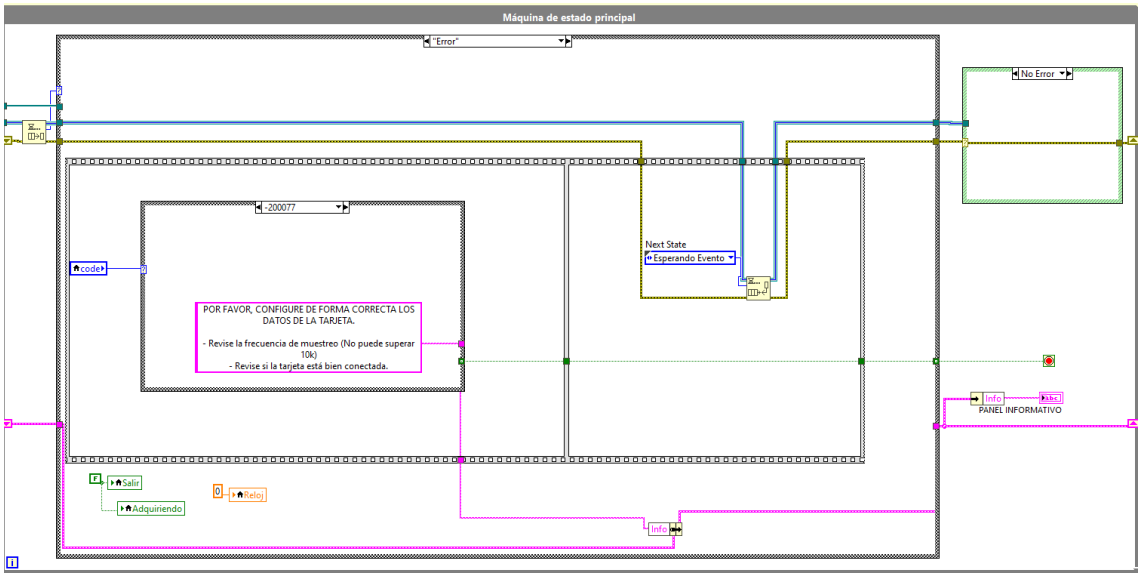


Figura 57. Diagrama de bloque del caso "Error".

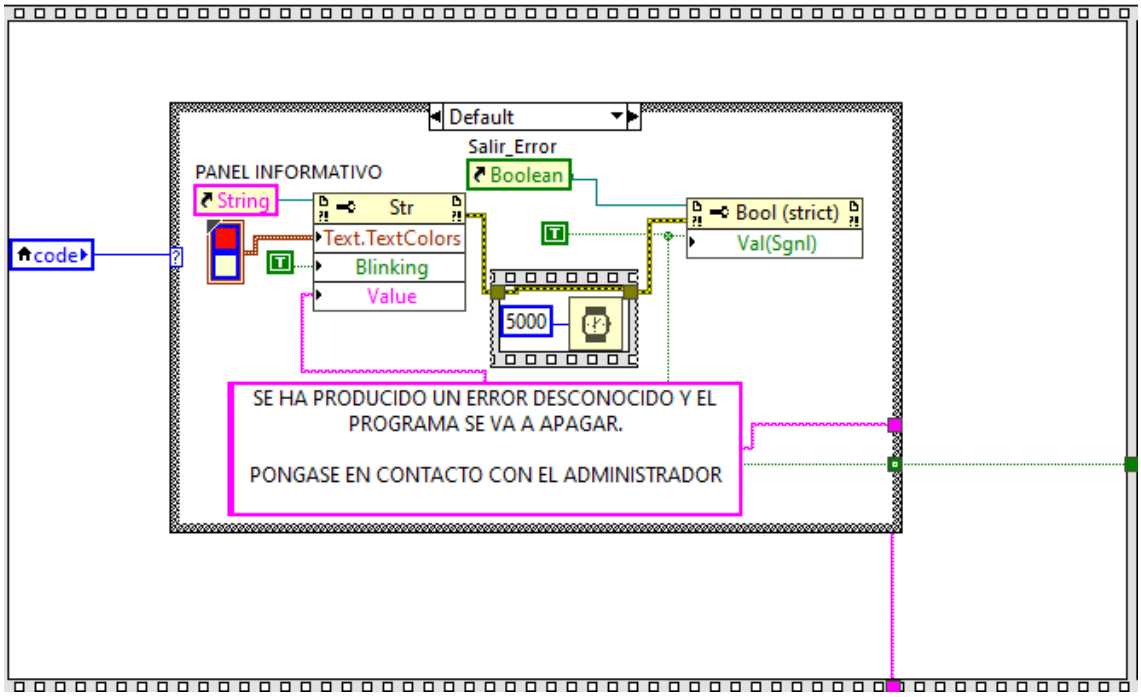


Figura 58. Detalle del caso "Error": error no controlado por la aplicación.

4.3.4 Interfaz de usuario

En todos los anteriores estados interviene un subVI encargado de gestionar las propiedades de los terminales que forman la interfaz de usuario. Este subVI toma como entrada un array que recoge las referencias de todos los terminales y el estado actual en el que se encuentra la máquina de estados principal. Mediante los Property Nodes se van asignando distintas caracteristicas a los terminales dependiendo del estado en el que se encuentre la aplicación. En la Figura 59 podemos ver la arquitectura de este subVI llamado GestionTerminales.

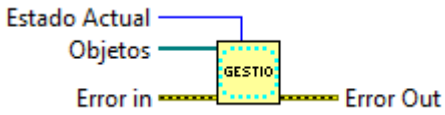


Figura 59. SubVI GestionTerminales.

En la Tabla 4 quedan recogidos todos los terminales que forman nuestra interfaz junto al comportamiento de cada uno de ellos en los distintos estados. Casi todos los terminales toman dos propiedades a lo largo del uso del programa: aparecen habilitados o deshabilitados.

Tabla 4. Comportamiento de los distintos terminales según el estado.

Posic Array	*OBJETOS / ESTADOS*	INICIALIZANDO	ESPERANDO	VALIDANDO PATH	LEYENDO FICHERO	GUARDANDO Y REPRESENTANDO	PARAR ADQUISICION
	PANEL INFORMATIVO						
0	Nombre				X		
1	Boton Salir	X	X	Deshabilitado	Deshabilitado	Deshabilitado	X
2	Graph Lectura	No Visible	No Visible	No Visible	Visible	No Visible	No Visible
3	Boton Lectura	X	X	Deshabilitado	Deshabilitado	Deshabilitado	X
4	Path Lectura	X	X	Deshabilitado	Deshabilitado	Deshabilitado	X
5	Boton Recoger Datos	X	X	Deshabilitado	Deshabilitado	Deshabilitado	X
6	Ancho de la Ventana	X	X	Deshabilitado	Deshabilitado	Deshabilitado	X
7	Frecuencia Muestreo	X	X	Deshabilitado	Deshabilitado	Deshabilitado	X
8	Min						
9	Max				Deshabilitado		
10	Boton Parar Adq	Deshabilitado	Deshabilitado	Deshabilitado	Deshabilitado	X	Deshabilitado
11	Grafica Graph	Deshabilitado e Inicializado	Deshabilitado	Deshabilitado	Deshabilitado	X	Deshabilitado
12	Grafica Graph Espectro	Deshabilitado e Inicializado	Deshabilitado	Deshabilitado	Deshabilitado	X	Deshabilitado
13	Grafica Chart	Deshabilitado e Inicializado	Deshabilitado	Deshabilitado	Deshabilitado	X	Deshabilitado
14	Canal Espectro	Deshabilitado	Deshabilitado	Deshabilitado	Deshabilitado	X	Deshabilitado
15	Boton Guardar Datos	X	X	Deshabilitado	Deshabilitado	Deshabilitado	X

En la Figura 60 podemos ver el caso de "Guardando y Representando" de este subVI. En este caso se encuentran deshabilitados la mayoría de los terminales, como podemos ver en la figura representando como "1…10", dejando solo habilitados las gráficas y el botón de parar adquisición, como se indica en la tabla anterior.

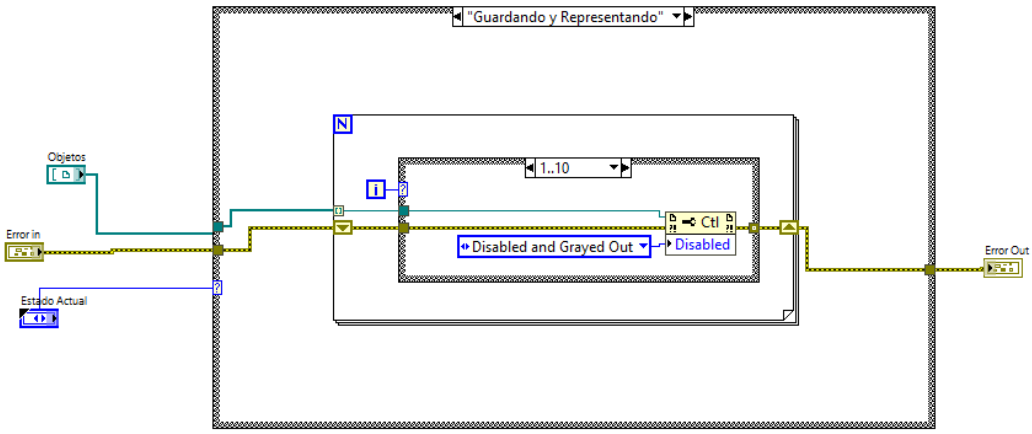


Figura 60. Diagrama de bloque del SubVI GestionTerminales.

La gráfica de lectura es la única que tiene un comportamiento diferente. Para este terminal trabajamos con la popiedad de visibildiad. En la Figura 61 podemos ver como se hace uso del property node de esta terminal en el caso de "Leyendo fichero". En los demás caso tendrá como parámetro de entrada el valor *false*, para que no sea visible.

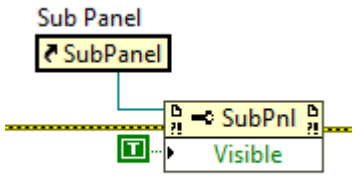


Figura 61. Property node del subpanel de la gráfica de lectura.

En la Figura 62 podemos ver en conjunto la función de los property node. Por un lado vemos visible la gráfica de lectura y por otro lado vemos que los demás terminales se encuentran deshabilitados.

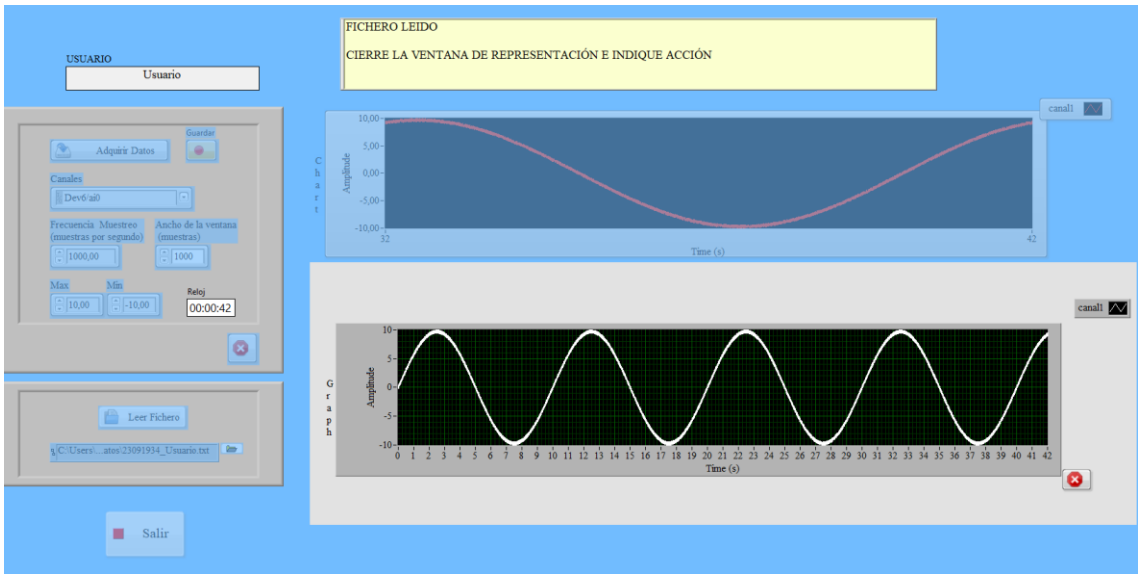


Figura 62. Panel frontal con el subpanel de la gráfica de lectura.

5

Pruebas del sistema

El sistema desarrollado ha sido sometido, tanto a pruebas de los diferentes módulos que lo componen, como a pruebas de su funcionamiento como un todo. En las primeras se ha testeado, sobre todo, el funcionamiento del dispositivo USB-6000 conectado al ordenador, usando para ello hardware y software auxiliar. En las segundas, se ha incorporado la conexión del equipo de poligrafía. A continuación, se dedican sendos apartados a detallar cada tipo de pruebas.

5.1 **Pruebas modulares**

Una vez familiarizados con el entorno de trabajo, comenzamos con el desarrollo de nuestra aplicación haciendo uso de un generador de funciones (AFG-21005 de RS-PRO), como simulador de las señales resultantes del equipo de registro real, junto al dispositivo USB-6000.

Para las pruebas también se hizo uso del software de NI Measurement & Automation Explorer (MAX). Este software, por un lado, ayuda a identificar cuáles son los canales conectados al módulo de adquisición ya que cuenta con un panel de testeo que muestra la adquisición que se está realizando. Así, sirve de guía para detectar si un fallo procede de la configuración o conexión del hardware o de la implementación del software que se está llevando a cabo sobre LabVIEW.

En la Figura 63 podemos ver un ejemplo de este panel, en el que se puede configurar el ancho temporal de la ventana que se representa (rate * samples to rate) lo que permite también comprobar la frecuencia de la señal de entrada. En este caso, el generador de señales se encuentra conectado al canal Dev3/ai0 y está generando una onda sinusoidal de amplitud 1,250 vpp y frecuencia 7Hz.

MAX también permite crear cualquier tipo de módulo de NI simulado, fijando el propio programa la frecuencia y amplitud de la onda de entrada al módulo simulado, permitiendo probar toda la parte de adquisición del programa LabVIEW sin necesidad de tener el módulo real conectado.

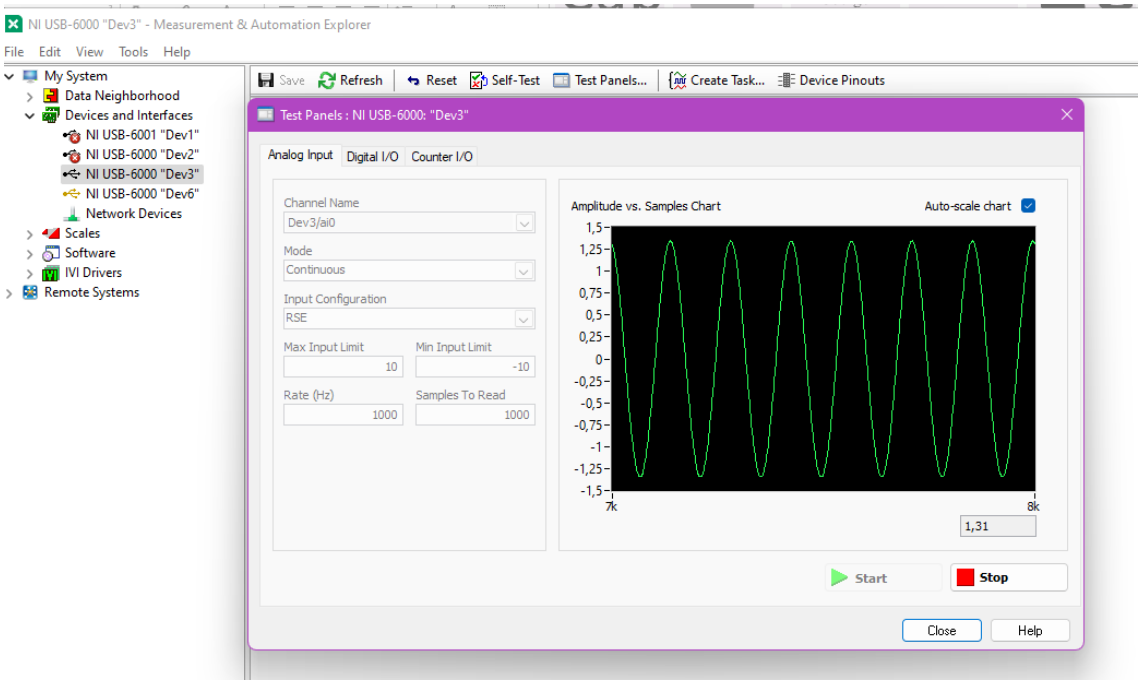


Figura 63. Test panel de NI-MAX.

Como se ha comentado, nuestra aplicación cuenta con distintas funcionalidades dependiendo del interés del usuario, por lo que las pruebas modulares quedan divididas según estas funcionalidades.

5.1.1 Adquisición, visualización y procesado de la señal

Partimos de una primera prueba que consistía en confirmar la correcta adquisición y visualización en los dos tipos de gráficas para un solo canal de adquisición. En una de ellas se debería ver el historial de registro y en la otra la señal adquirida al instante. En la Figura 64 podemos observar la diferencia de estos dos tipos de gráficas, y su correcto funcionamiento, para una salida del generador de funciones. Se probó con diversos valores de la señal del generador.

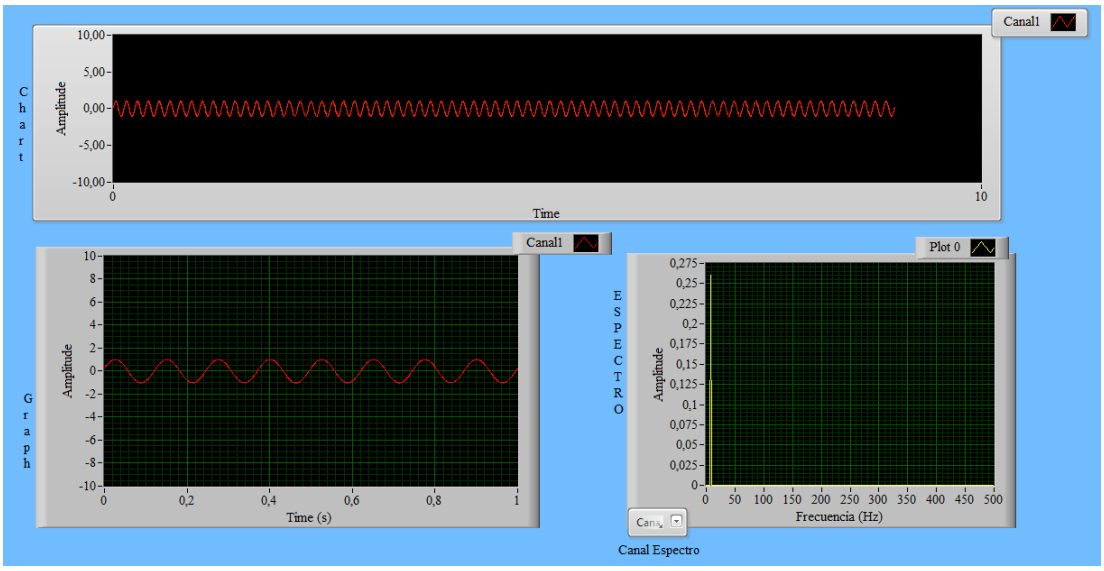


Figura 64. Prueba modular sobre la visualización de los dos tipos de gráfica.

Respecto a la tercera gráfica (Espectro) tuvimos que comprobar que los datos que representaba eran correctos. Según el valor de frecuencia que configurásemos en el generador de señales, la gráfica del espectro variaría. En la Figura 65 se puede ver una de las pruebas realizadas: comprobamos que generando una señal de una frecuencia de 4Hz en el generador se mostraba un pico en dicho valor del eje X y si lo cambiábamos a 5Hz el pico también cambiaba.

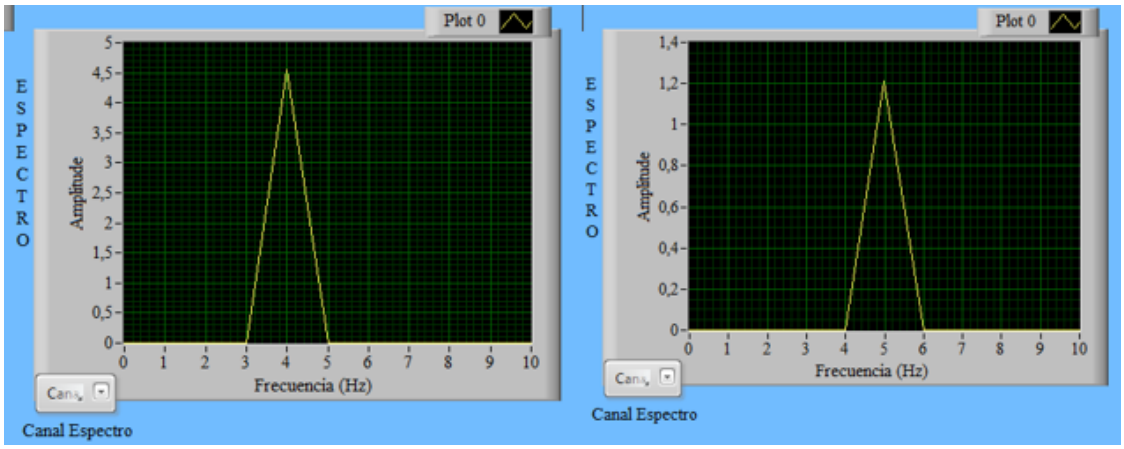


Figura 65. Distintas casuísticas gráfica Espectro. (4Hz y 5Hz)

Para ampliar una gráfica es tan sencillo como editar el último valor del eje X para que cambie. Esta diferencia la podemos ver si comparamos los ejes X de la gráfica espectro de la Figura 64 y Figura 65.

Una vez comprobado el correcto funcionamiento para un canal, pasamos a hacer pruebas con dos canales. Mediante el generador de señales, configuramos distintas formas y valores para poder identificar la distinción entre ambas. En la Figura 66 podemos ver un ejemplo.

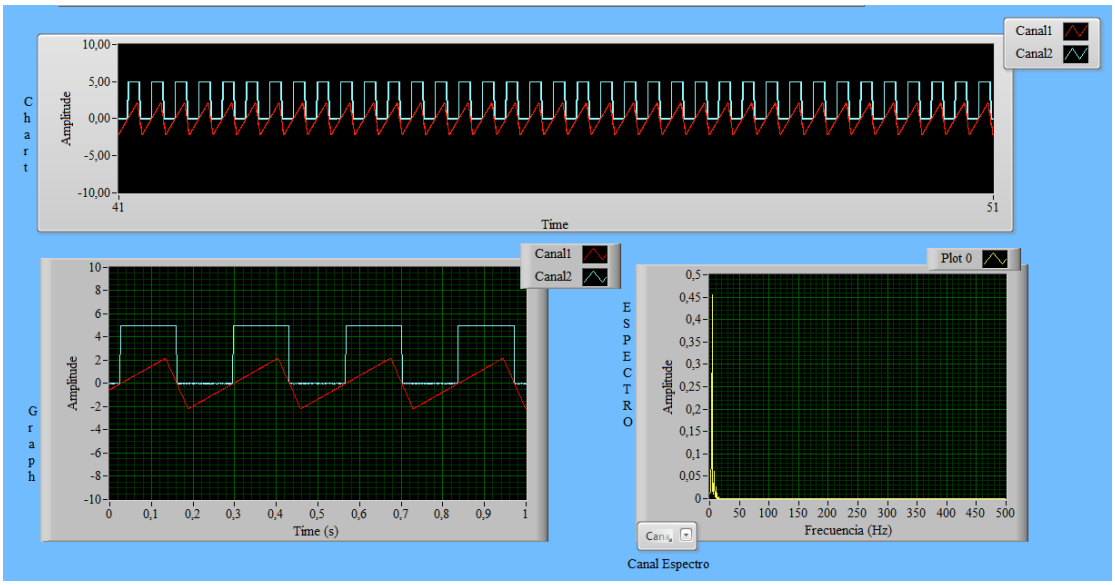


Figura 66. Adquisición de dos señales.

Otra acción con la que queríamos contar en nuestra aplicación era que, cuando se representara más de una señal, poder elegir cuál de ellas queremos visualizar, si todas, una o algunas. En la Figura 67 podemos ver que en la gráfica superior solo sale representada la señal correspondiente al nombre "Canal2" mientras que, en la inferior, salen las dos indicadas.

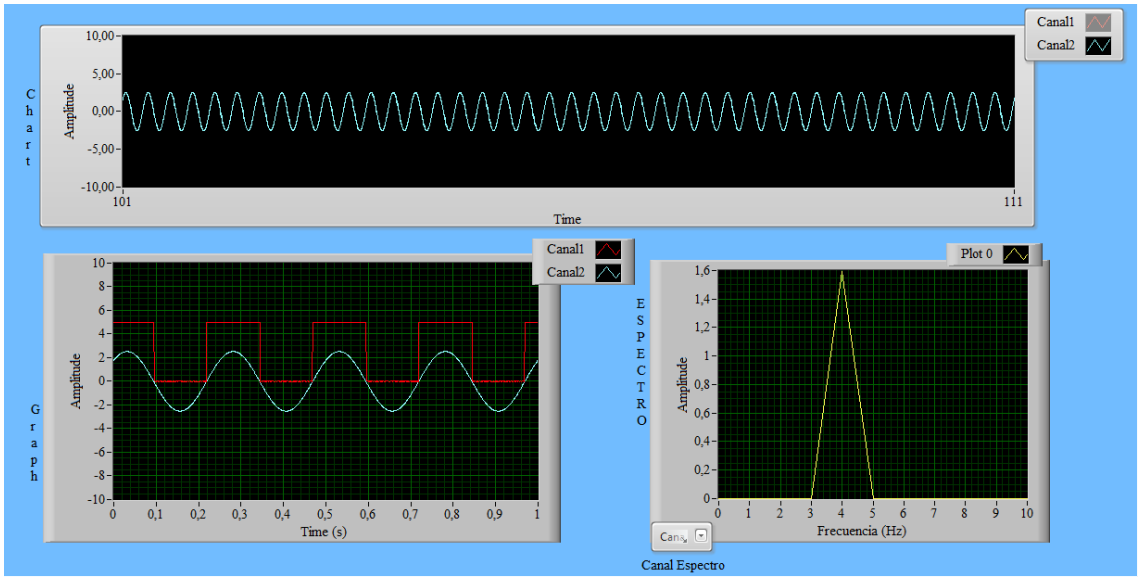


Figura 67. Visualización de una sola señal en la gráfica chart.

Una vez hecho la prueba de la representación de dos canales pasamos a hacer la prueba de hasta ocho canales. En este caso solo pudimos conectar dos al generador de señales (el generador solo tiene dos salidas), por lo que los restantes se representan como una línea recta, lo que se puede comprobar visualizando en la gráfica de uno en uno los canales (como se hizo en el caso mostrado en la Figura 67). Lo que sí se puede ver con claridad en la Figura 68, que corresponde a la adquisición de los ocho canales, es que están activas las leyendas de las ocho gráficas (obsérvese que las leyendas de las Figura 64 y Figura 66 tienen la longitud del número de canales adquiridos en cada caso).

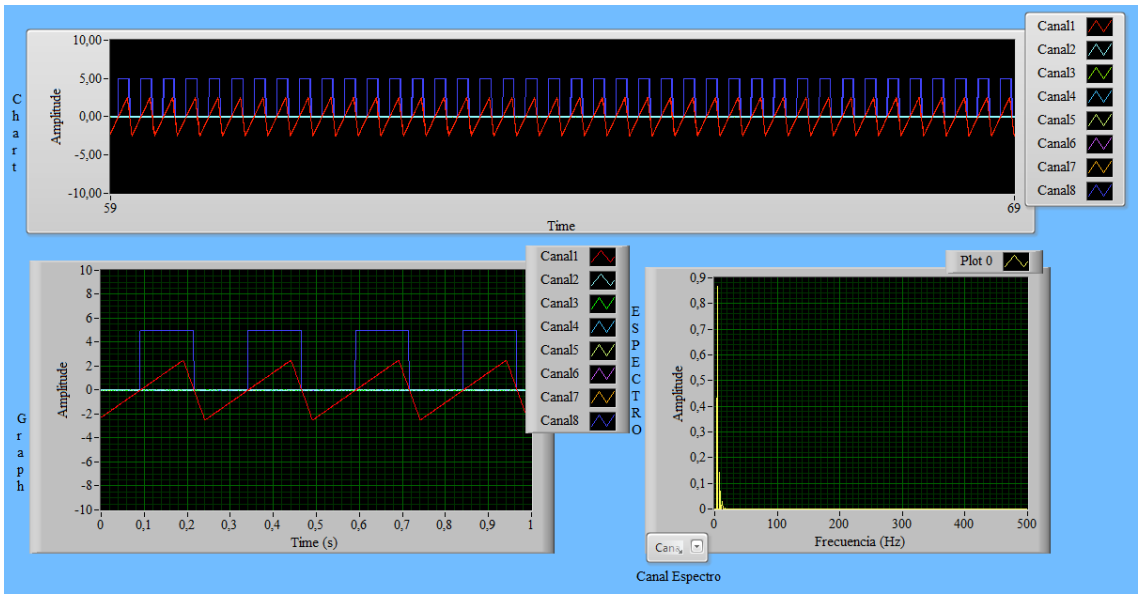


Figura 68. Prueba modular de 8 canales (módulo USB-6000 real).

Para probar de forma más exhaustiva la adquisición de ocho canales, hicimos uso del módulo simulado que emula una señal diferente en cada uno de sus canales (Figura 69).

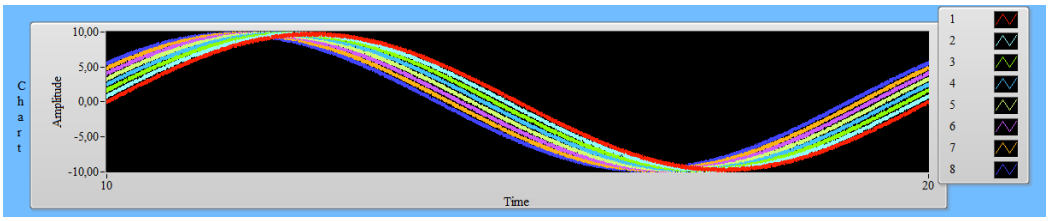


Figura 69 Prueba modular de 8 canales (módulo USB-6000 simulado).

Haciendo este tipo de pruebas modulares nos dimos cuenta de que era muy importante el correcto uso de las funciones DAQmx Start Task y DAQmx Stop Task ya que la tarjeta de adquisición cuenta con un buffer limitado y si la tarjeta no es parada antes de seguir haciendo otras acciones acaba saltando un error debido a la falta de espacio en el buffer.

5.1.2 Configuración de los tiempo de adquisición

En el panel frontal aparecen dos valores que pueden ser definidos según el interés del usuario: frecuencia de muestreo (muestras por segundos) y ancho de la ventana (muestras). La gráfica de la parte inferior, que muestra las ventanas de datos según se van adquiriendo, variará dependiendo de estos dos valores, ya que en el eje X se debe de mostrar el ancho de ventana en tiempo, no en número de muestras. Esto es, el tiempo total representado tomará el valor resultante de la división entre ancho de la ventana en muestras y la frecuencia de muestreo. La gráfica de la parte superior, debe ir representando esta misma ventana temporal, pero conservando en pantalla alguna ventanas temporales adquiridas previamente. Como ya se dijo en el capítulo de desarrollo este número de ventanas anteriores no se puede variar desde programa, con lo que se decidió dejar un ancho total fijo de 10000 muestras.

Las pruebas mostradas hasta ahora se han llevado a cabo con una frecuencia de muestreo de 1000 muestras por segúndo y un ancho de ventana de 1000 muestras,

ambos fijos. En esta etapa, se realizaron de nuevo las pruebas anteriores pero, esta vez, los datos de ambos parámetros podrán ser indicados por el usuario desde el panel frontal.

En la Figura 70 y Figura 71 podemos ver dos ejemplos con distintos valores de frencuencia de muestreo y número de muestras. Se aprecia en la primera que la gráfica inferior representa 2,5 s (que corresponden a 5000 muestras adquiridas a 2000 muestra por segúndo) y en la segúnda , 0,4 s (para una frecuencia de muestreo de 5000 y un ancho de ventana de 2000). La gráfica superior, que recoge el histórico, tiene distintas longitudes temporales en una y otra figura, al tener fijo el número de muestras. En el primer caso, el histórico de 10000 muestras fijado representa 5 s (ya que se muestrea a 2k muestras/s) y, en el segúndo, 2s (ya que se muestrea a 5k muestras/s).

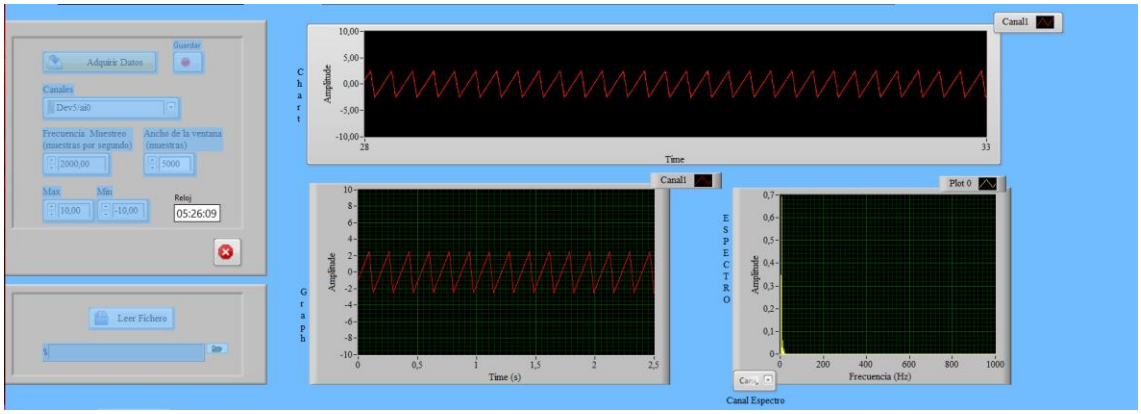


Figura 70. Representación de la señal con una frecuencia de muestro de 2000 y un ancho de ventana de 5000.

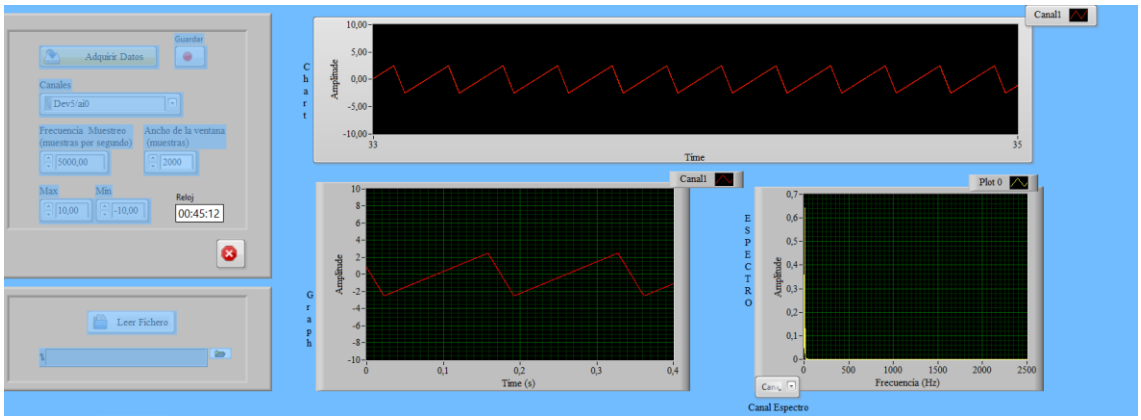


Figura 71. Representación de la señal con una frecuencia de muestro de 5000 y un ancho de ventana de 2000.

5.1.3 Almacenamiento en fichero

Tras probar la correcta adquisición y representación de los datos pasamos a hacer pruebas relacionadas con el almacenamiento de estos. Era importante comprobar que los datos se guardaran bien y con la estructura deseada.

Para comprobar el correcto almacenamiento de los datos, se hicieron diversas adquisiciones (con señales de entrada conocidas y con distinto número de canales seleccionados) y, posteriormente, se abrieron los ficheros generados con MS-Excel y se

representaron, para comprobar que la señal representada era la misma que la que se pretendió adquirir y almacenar con nuestro programa.

Una vez comprobado que los datos se guardaran de forma correcta en el fichero añadimos una cabecera formada por el nombre del usuario, la fecha y la palabra "DATOS:", que nos serviría de guía a la hora de leer el fichero en nuestra aplicación. Podemos ver un ejemplo en la Figura 72.

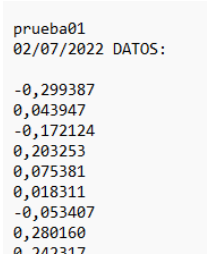


Figura 72. Fichero con la estructura inicial.

Con el desarrollo de la aplicación fuimos modificando la cabecera del fichero. Añadimos el nombre de los canales introducidos por el usuario, para poder identificar las distintas señales, junto a la frecuencia de muestreo de la adquisición, para poder representar la señal almacenada respecto al tiempo, quedando como se muestra en la Figura 73.

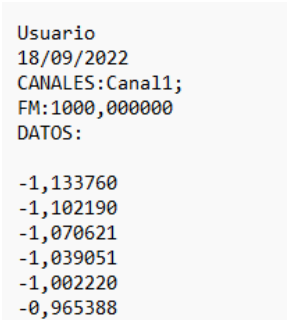


Figura 73. Fichero con la estructura final.

Para seguir evaluando si se estaban adquiriendo todos los datos de forma correcta, abrimos distintos ficheros generados con el bloc de notas y comprobamos el número de líneas. Por ejemplo, definimos la adquisición con una frecuencia de muestreo de 1000 muestras por segundo durante 10 segundos, por lo que el número de datos recogidos debían ser de 10000 datos. Tras abrir el fichero pudimos ver 10006 registros, de los cuales seis corresponden a las filas de la cabecera predefinidas (Figura 74).

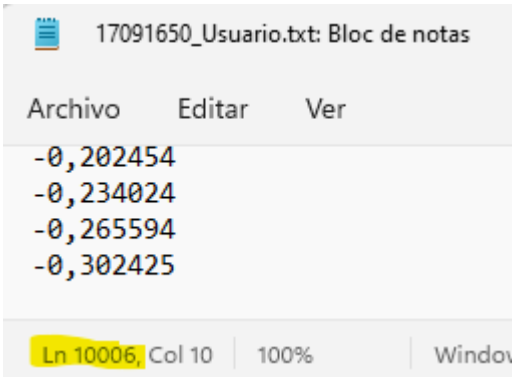


Figura 74. Número de registros durante 10 segundos con una frecuencia de muestreo de 1000.

De nuevo se recurrió al dispositivo simulado para hacer las pruebas de más de dos canales, por la imposibilidad de generar más de dos señales reales distintas con el generador de funciones, que ya se comentó (Figura 75).

```
Prueba
16/09/2022
CANALES:1;2;3;4;5;6;7;8;
FM:1000,000000
DATOS:

-0,299387       0,883206        1,500290        2,695395        3,368328        4,087039        4,759667        5,800958
0,199591        0,999176        1,494491        2,731407        3,449202        4,112674        4,737083        5,277261
 0,233161       0,776086        1,484429        2,321543        3,621937        4,077578        4,631489        5,276345
```

Figura 75. Fichero con la estructura final (8 canales).

En paralelo con estas pruebas, se verificaba que el nombre de fichero se construía según lo indicado en los requisitos, esto es, con una concatenación del día, mes, hora y minutos seguidos del nombre del usuario.

5.1.4 Lectura y representación desde fichero

Una vez hechas las pruebas referentes a la adquisición y almacenaje, pasamos a las relacionadas la lectura y correcta representación de las señales leidas del fichero. Como resultado final debiamos obtener en la gráfica de lectura lo mismo que en las gráficas de adquisición para corroborar el correcto funcionamiento.

En la Figura 76 podemos ver un ejemplo de prueba donde primero se adquieren datos de un solo canal y después se lee es mismo fichero. A simple vista no se parecen debido a que la gráfica de lectura (parte inferior) lee todo el fichero y la gráfica de adquisición (parte superior) es solo los diez últimos segundo de la adquisición.

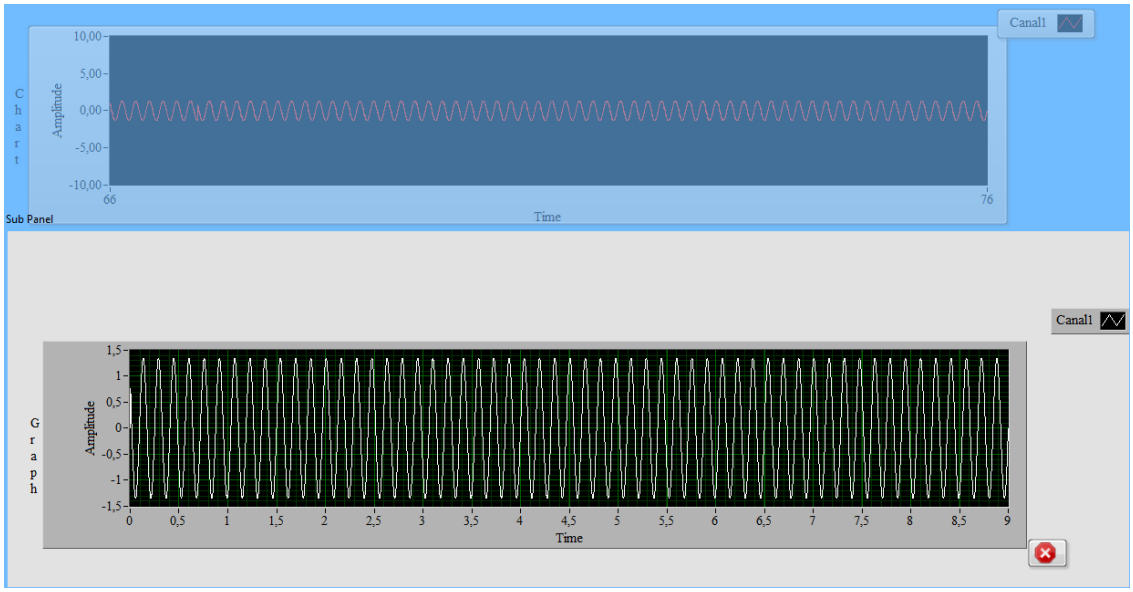


Figura 76. Lectura de un fichero adquirido.

Si cambiamos los valores del eje X de la gráfica de lectura ya si podemos ver dos gráficas iguales como se puede apreciar en la Figura 77.

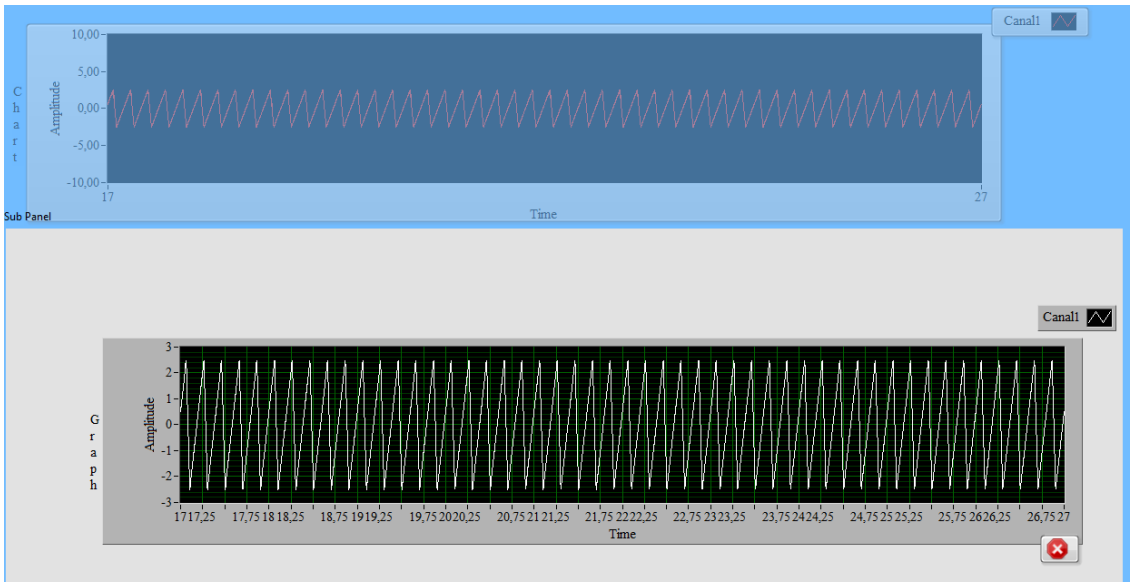


Figura 77. Ampliación de la señal del fichero.

En la Figura 78 dejamos el resultado de la lectura de un fichero que recoge los datos de dos canales. Aunque también se realizaron las pruebas correspondientes a la lectura de ficheros de ocho canales, generados, como ya se indicó, con el módulo simulado.

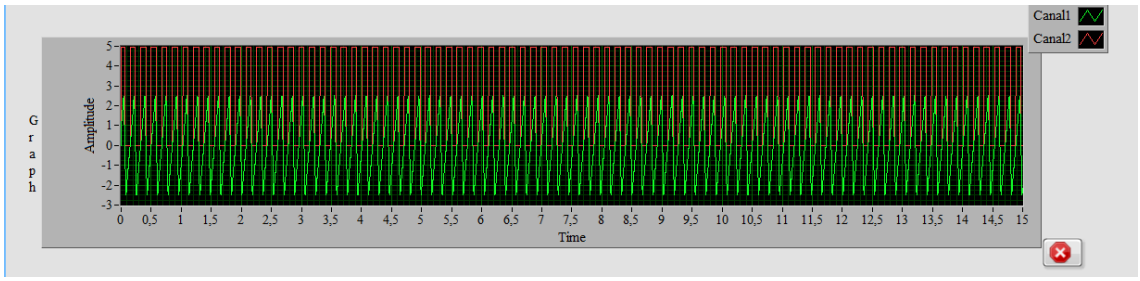


Figura 78. Lectura de un fichero con 2 canales.

5.2 **Pruebas integrales**

Las pruebas modulares recién expuestas se fueron haciendo, bien con programas idependientes, que luego se incorporarían a la aplicación final, o con la aplicación final en distintas fases de desarrollo. Una vez finalizado el programa, se repitieron todas las pruebas comentadas en el apartado anterior, más aquellas que implican el funcionamiento integral de la aplicación. A continuación, con ayuda de la tabla de requisitos, se señalarán aquellas pruebas que se hicieron de forma análoga a las modulares y se mostrarán en este apartado solo los resultados de las pruebas adicionales no mostradas allí. También en esta fase se sustituyó el generador de funciones por el equipo de registro analógico LabLinc V, mostrándose también aquí alguna de las capturas realizadas con el sistema completo.

En la Tabla 5 quedan recogidas las validaciones de los requisitos que se plantearon en el capítulo de especificaciones (Tabla 3). Se indica en la columna 'Validación' un 'OK' en aquellos requisitos cumplidos y 'KO' en los no cumplidos. Como se puede ver, por falta de tiempo, quedó pendiente el desarrollo correspondiente a los requisitos relacionados

con el procesado de la señal (Adq_Procesado), dejando este trabajo como una posible linea futura.

Tabla 5. Validación de los requisitos funcionales.

Número	Nombre Corto	Validación
1	Identificar	OK
2	Acción	OK
2.1	Adquirir	OK
2.1.1	Adq_Conf_Canal	OK
2.1.2	Adq_Conf_Fm	OK
2.1.3	Adq_Grafica	OK
2.1.3.1	Graf_Tiempo	OK
2.1.3.2	Graf_Visible	OK
2.1.3.3	Graf_Leyenda	OK
2.1.4	Adq_Arr_Para	OK
2.1.5	Adq_Escribir_Fichero	OK
2.1.5.1	Nombre_Fich	OK
2.1.5.2	Cabecera_Fich	OK
2.1.6	Adq_Procesado	KO
2.1.6.1	Proc_FFT	OK
2.1.6.2	Proc_HR	KO
2.2	Recuperar	OK
2.2.1	Rec_Fichero	OK
2.2.2	Rec_Graf_tiempo	OK
2.2.3	Rec_Graf_Visible	OK
2.2.4	Rec_Graf_Leyenda	OK
2.3	Salir	OK
3	Warning	
3.1	Err_Conexión	OK
3.2	Err_Config	OK
3.3	Err_Path	OK
3.4	Err_Fichero	OK
4	Error	OK
5	Usabilidad	OK

Tras una pruebas previas pasamos a validar los requisitos funcionales indicados en la Tabla 3.

- Identificar. Una vez iniciemos nuestra aplicación se mostrará un popup donde el usuario deberá introducir su nombre. Se comprueba qua la aplicación no permite continuar hasta que no se rellena el campo (Figura 79).

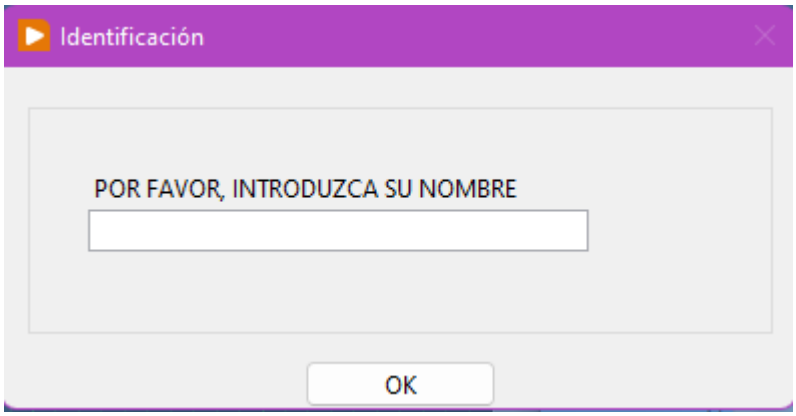


Figura 79. Popup de identificación.

- Acción. El usuario podrá comenzar la adquisición, guardar los datos de la adquisición en un fichero (Adq_Escribir_ Fichero), parar la adquisición (Adq_Arr_Para), leer un fichero o abandonar la aplicación (Salir). Permitiendo configurar hasta 8 canales, asignándole un nombre de forma identificativa (Adq_Conf_Canal), establecer una frecuencia de muestreo (Adq_Conf_Fm) y representar los datos en dos tipos de gráficas (Adq_Grafica). Los requerimientos Graf_Tiempo, Graf_Visible y Graf_Leyenda quedan representados en el apartado anterior, al igual que el requerimiento Adq_Escribir_ Fichero, incluidos el requerimiento Nombre_Fich y Cabecera_Fich, podemos verlo en la Figura 73.

  En la Figura 80 podemos ver las distintas acciones que puede realizar el usuario y la configuración de dos canales con sus nombres establecidos. También se comprobó que el terminal Reloj se pone a 0 cada vez que arranca una adquisición y, mientras que dicha adquisición no se detenga, se va incrementado cada segundo su valor.

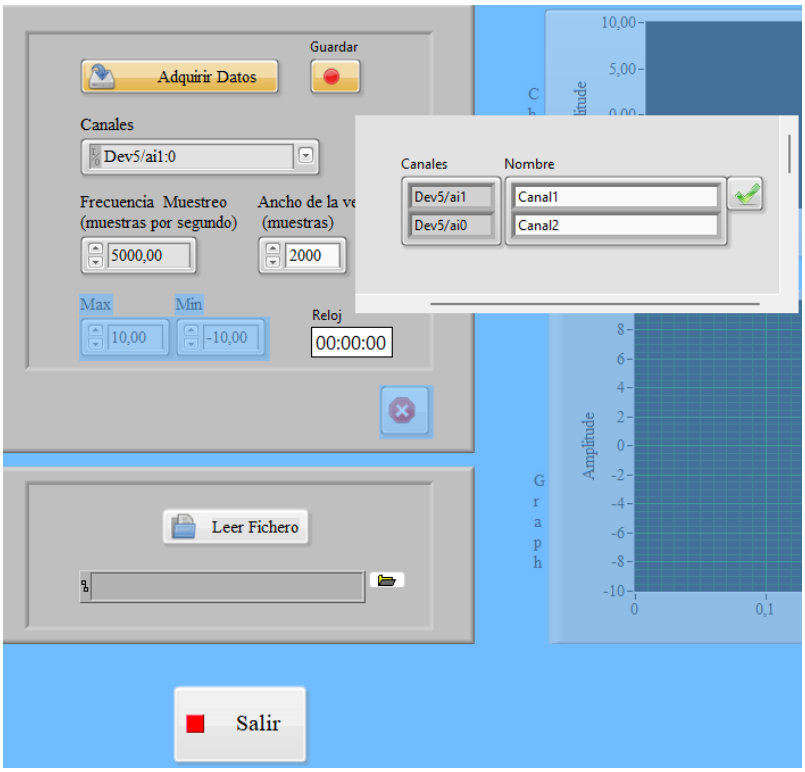


Figura 80. Requerimientos englobados en el número 2: Acción.

- Recuperar. Los requerimientos incluidos en este bloque 2.2 quedan representados en las pruebas unitarias del apartado 5.1.4.

- Warning. Mediante el panel informativo la aplicación avisará al usuario:
  - Cuando el dispositivo de adquisición no esté bien conectado (Err_Conexión). Se realizó la prueba iniciando el programa sin conectar el dispositivo USB-600 e iniciando la adquisición de datos. En la Figura 81 podemos ver el mensaje de advertencia.

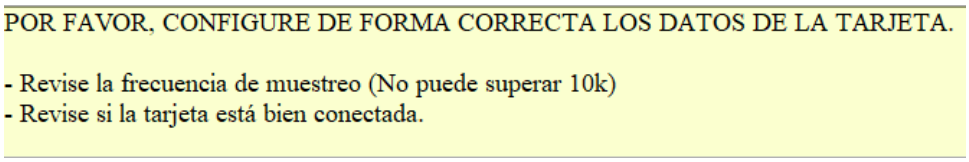


Figura 81. Advertencia cuando la tarjeta de adquisición no está bien configurada.

  - Cuando la tarjeta de adquisición no esté bien configurada (Err_Config). Se realizo la prueba escogiendo una frecuencia de muestreo superior a la que soporta el dispositivo, por ejemplo 10k muestras/s con más de un canal seleccionado. (Figura 82).

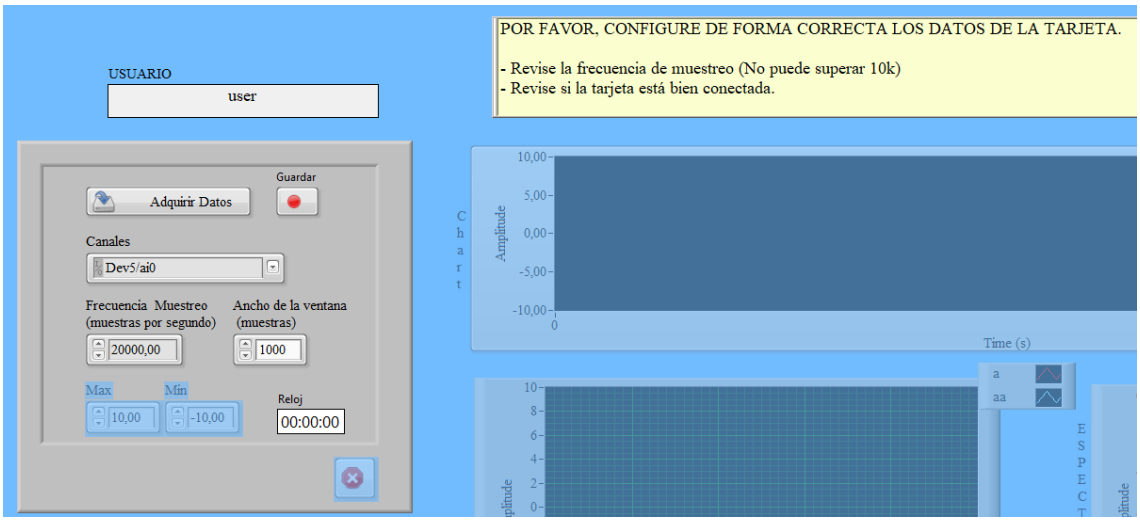


Figura 82. Mensaje informativo cuando la tarjeta de adquisición no está bien configurada.

  - Cuando la ruta path del fichero que desea leer no esté indicada (Err_Path). Se realizo la prueba pulsando Leer fichero sin haber seleccionado previamente el fichero. En la Figura 83 se muestra esta casuística.

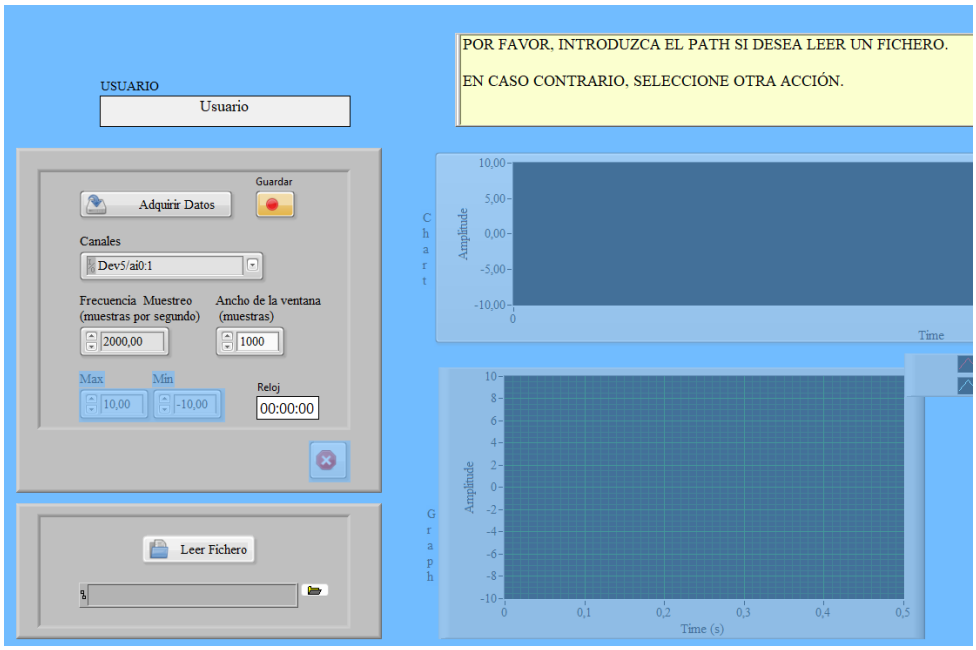


Figura 83. Mensaje informativo cuando seleccionamos Leer Fichero, pero la ruta está vacía.

- Cuando el fichero que desea no es el correcto o no contiene datos (Err_Fichero). Se realizo la prueba añadiendo un fichero con otro formato (Figura 84) y sin datos, solo la cabecera. (Figura 85)

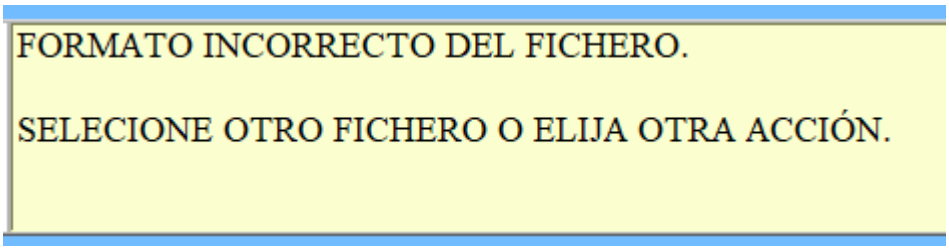


Figura 84. Panel informativo cuando el fichero seleccionado no es correcto.

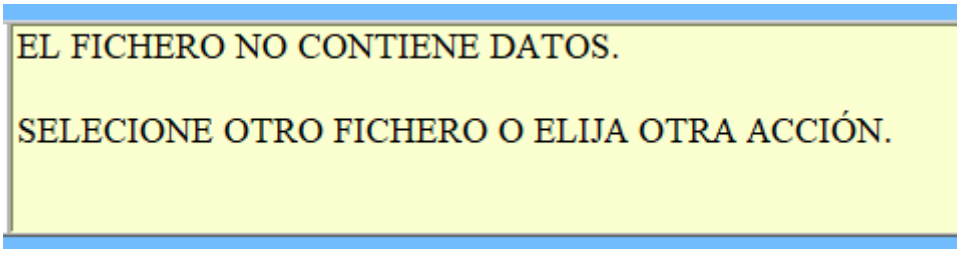


Figura 85. Panel informativo cuando el fichero no contiene datos.

- Error. En caso de que se produzca un error diferente a los mencionados se mostrará un mensaje de error y se apagará la aplicación. Se realizó la prueba colocando un punto de ruptura en tras la función de lectura de datos, para asegurar que el buffer se llenaba, hecho que provoca que la función de la DAQ genere un error que, efectivamente hace que la aplicación finalice. (Figura 86).

Figura 86. Mensaje de error.

- Usabilidad. Gracias a los property nodes se guiará al usuario las acciones que puede tomar en cada momento mostrando u ocultando los componentes de la interfaz de usuario tal y como hemos visto.

Para finalizar este apartado, se muestran algunas capturas de el funcionamiento del sistema completo, esto es, con la conexión del módulo de polifrafía Lab V al USB-6000.

En la Figura 87 podemos ver una captura sobre el resultado la adquisición la PPG. Como comentamos en el apartado 2.1.2, esta señal depende mucho del usuario. Gracias al módulo *V71-40* pudimos amplificar la señal como se puede observar en la figura.

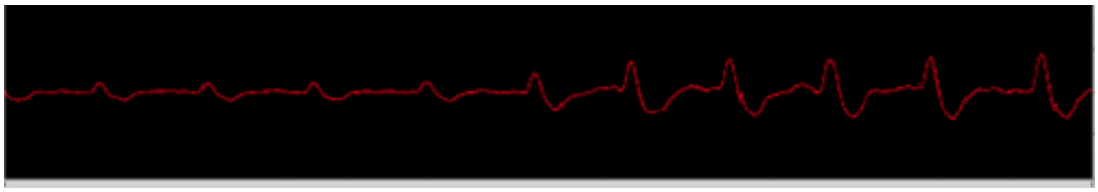


Figura 87. Adquisición de la señal PPG.

En la Figura 88 queda representando la toma de datos mediante los tres modulos facilitados. En la gráfica superior se ve solo la señal PPG, debido a que se puso dicho filtro de visibilidad en la leyenda, en la gráfica inferior podemos ver la señal PPG junto a la ECG y EEG. En la Figura 89 podemos ver las señales PPG y ECG.

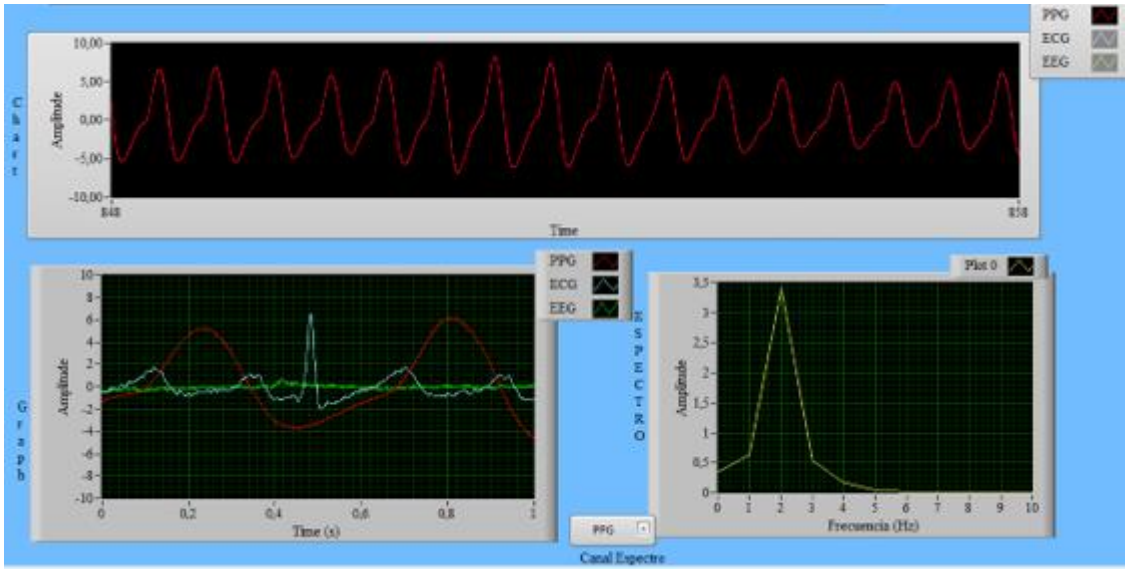


Figura 88. Representación de las señales PPG, ECG y EEG, con filtro de visibilidad en PPG.

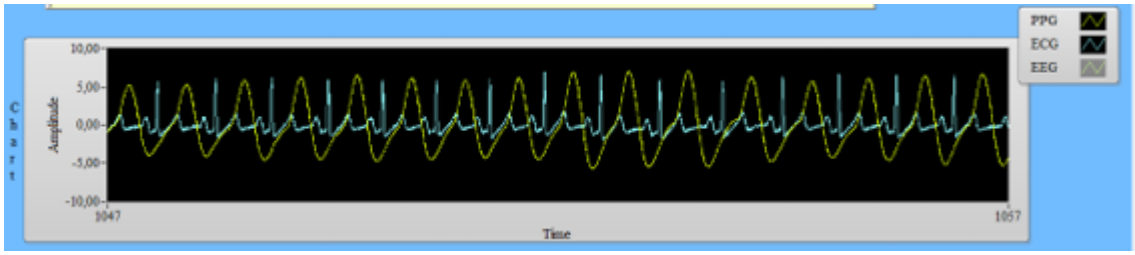


Figura 89. Representación de las señales PPG y ECG.

Respecto a la señal EEG, es una señal muy débil con los ojos abiertos y sin puntos característicos (como los máximos y los mínimos del ECG y la PPG) por lo que era dificil detectar si se estaba recogiendo correctamente la salida del módulo correspondiente. Optamos por posicionar los electrodos en la sién y verificar que la señal respondía a la EMG facial, así pudimos apreciar los cambios esperados en la señal siempre y cuando se pestañeara o se apretaran los dientes fuertemente , como podemos ver en la Figura 90.

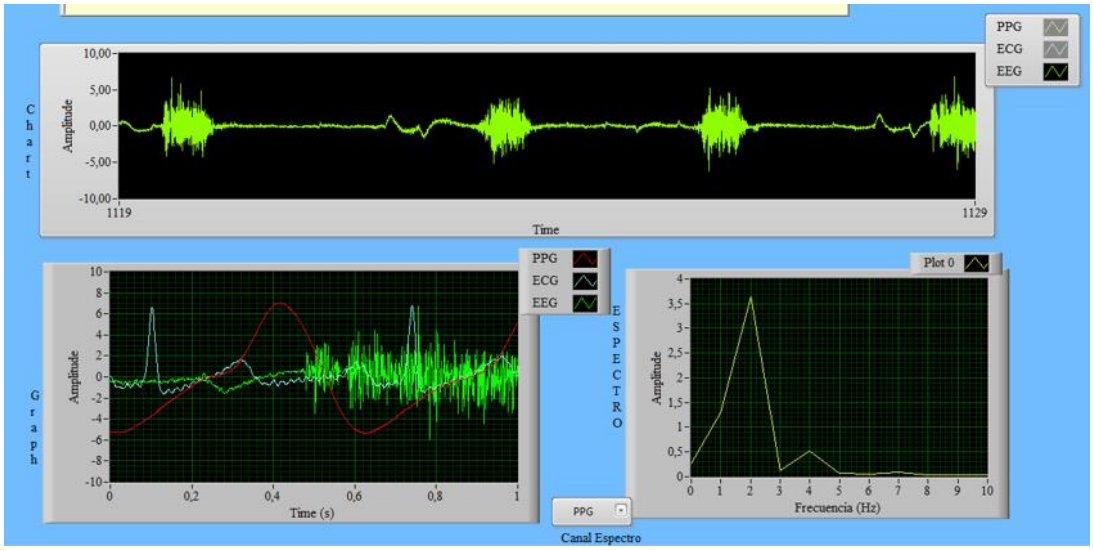


Figura 90. Representación de las señales PPG, ECG y EEG, con filtro de visibilidad en EEG.

En la Figura 91 dejamos una captura de la representación de las tres señales juntas en ambas gráficas, mostrandose en la gráfica del espectro la señal PPG, como se puede apreciar en la imagen.

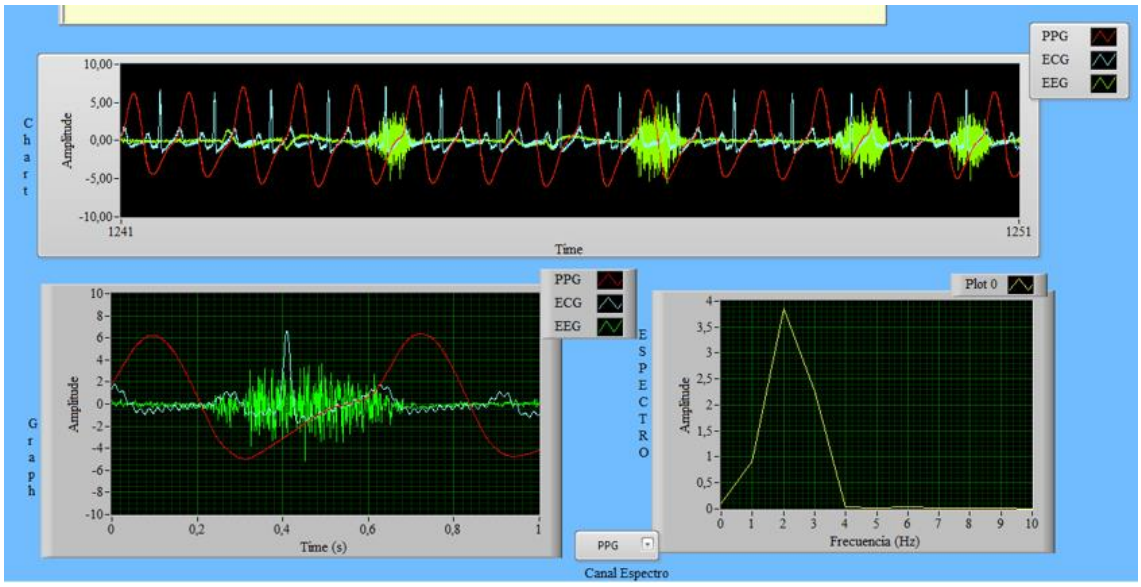


Figura 91. Representación de las señales PPG, ECG y EEG.

Como comentamos en el apartado 4.3.1.2, la gráfica del espectro fue añadida para detectar a tiempo real cualquier interferencia que se pudiese producir. Hicimos una prueba en el laboratorio que cuenta con tubos fluorescentes de 50Hz (estos invierten el valor negativo provocando una interferencia de 100Hz) con el transductor de *V71-40*. La prueba consistía en posicionar el transductor hacia el techo provocando una

saturación en la señal y un pico de 100Hz en la gráfica del espectro, como podemos ver en la Figura 92.

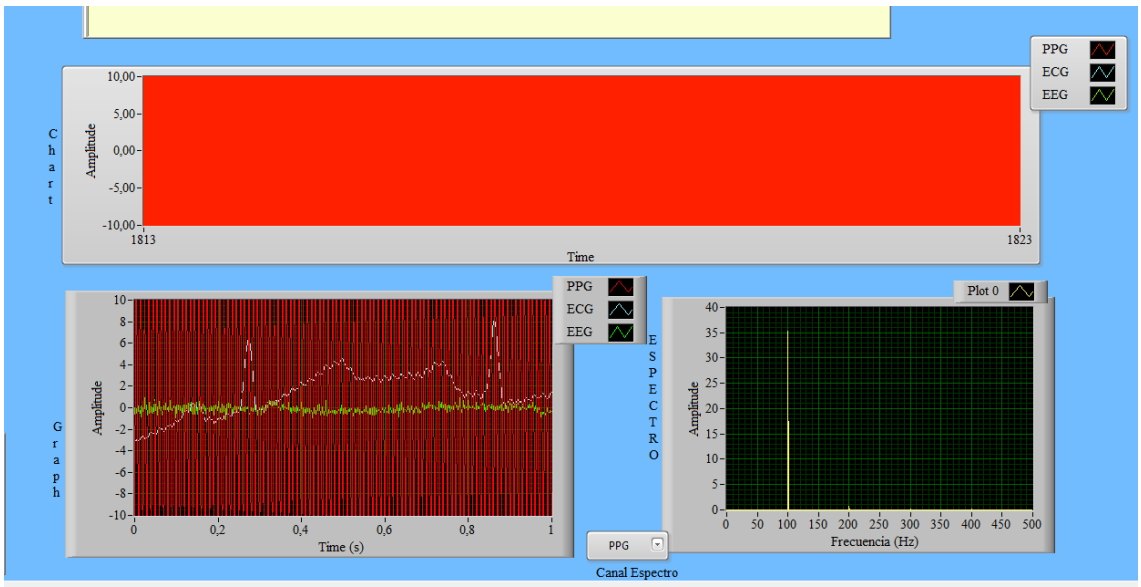


Figura 92. Prueba del transductor del *V71-40*.

También se llevaron a cabo pruebas típicas de una sesión de clase en la que se muestra el efecto del filtrado sobre la señal ECG, manipulando los correspondientes controles del panel frontal del módulo V75-04 (Figura 93).

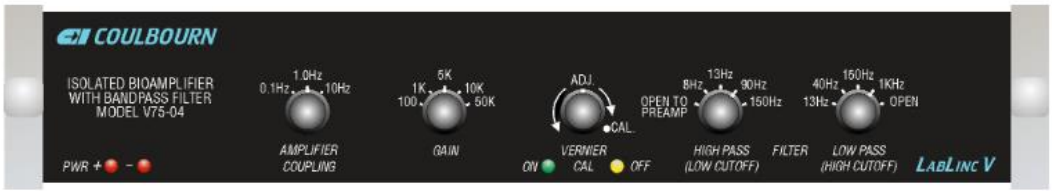


Figura 93. Controles módulo V75-04.

Podemos ver, en la Figura 94, la señal ECG tras realizar el mínimo filtrado que permite el módulo y, en la Figura 95, las modificaciones que sufre esta señal tras aplicarle una banda más estrecha de filtrado, que elimina el desplazamiento de línea de base y el rizado de red.

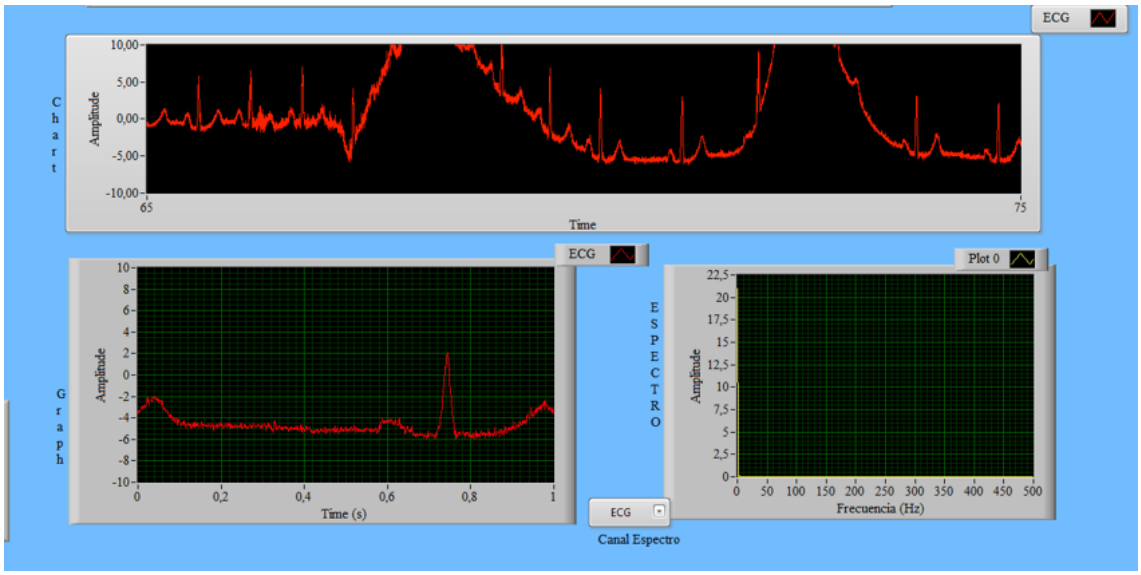


Figura 94. Señal ECG aplicando el mínimo filtrado posible [0,1-Open] Hz.

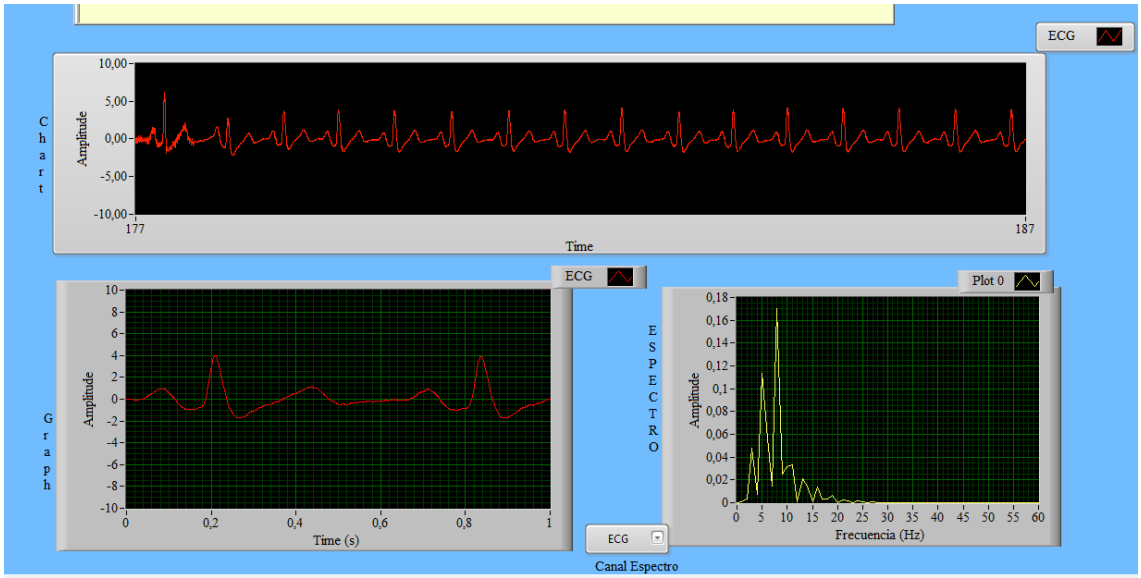


Figura 95. Señal ECG aplicando un filtrado de [1-40] Hz.

6

Conclusiones y líneas futuras

6.1 **Conlusiones**

Una vez finalizada nuestra aplicación podemos decir que hemos cumplido los objetivos propuestos al principio de nuestro desarrollo: se ha obtenido una aplicación capaz de adquirir, representar, almacenar en fichero y aplicar un cierto procesado a las señales fisiologicas provinientes del módulo LabLinc V, usando el dispositivo USB-6000 de NI. Si las señales se guardaron, la aplicación también permite recuperarlas para visualizarlas. Se ha usado el dispositivo de adquisición de menores prestaciones de los que disponía el DTE, tras asegurar que era adecuado para realizar la etapa de digitalización mediante un estudio previo tanto de las señales fisiologicas como de los distintos modelos de dispositivos de adquisición disponibles.

Se trata de una aplicación docente, así que la mayor parte de los requistos del interfaz de usuario fueron dictados por los tutores, añadiendo la autora de este trabajo algunos requisitos adicionales como la posibilidad de dar un nombre a los canales asociados con la señal del polígrafo al que están conectados o todos los relacionados con los posibles errores de la aplicación y su usabilidad. Entendemos que el cumplimiento de todos los requisitos mínimos planteados ha sido verificado mediante el plan de pruebas detallado en el capítulo correspondiente.

La aplicación ha sido desarrollada en LabVIEW y, al principio de este documento, comentabamos que el factor determinante para elegir este sistema de desarrollo era mi interés por aprender sobre la programación gráfica, ya que durante la carrera no se han impartido asignaturas con este tipo de programación. Los resultados de este trabajo muestran la potencialidad del lenguaje de programación gráfica para la implementación de sistemas inteligentes capaces de adquirir bioseñales en tiempo real. Durante toda la carrera hemos estado practicando con otro tipo de programación y puedo decir que es

a la que estoy más acostumbrada, pero gracias al desarrollo de este trabajo he adquirido unos nuevos conocimientos sobre una programación que desconocia en la que puedo decir que me habría gustado conocer antes para poder explorar todas las herramientas y funciones que contiene. Considero que los conocimientos adquiridos me han ayudado a aumentar mis herramientas para enfrentarme al futuro laboral.

6.2 Líneas futuras

Una de las consideraciones de diseño que hemos tenido en cuenta durante todo el desarrollo es que nuestra aplicación sea facilmente ampliable para que puedan ofrecerse nuevas líneas de trabajo futuras.

Por una parte, y tal y como hemos indicado en el apartado 5.2, tenemos algunos requisitos de la Tabla 3 que no hemos podido cubrir. Estos requisitos están relacionado con el procesado de la señal, en concreto, el cálculo de la tasa cadiaca partiendo del ECG o del PPG. La implementación de un algoritmo de umbral se sugiere como inmediata ampliación de la aplicación.

Aunque no figura en los requistos, nos planteamos que el usuario pudiera definir el tiempo total que se representa en la gráfica que visualiza el histórico del registro. Como ya se explicó, no ha sido posible porque LabVIEW no permite variar este parámetro desde el programa. Proponemos estar atentos a posibles cambios de este hecho en futuras versiones de LabVIEW para incluir esta funcionalidad o, más complicado, hacer la programación pertinete para gestionar esta historia y representarla en una gráfica que no sea de tipo chart.

Por otro lado, durante las pruebas integrales, estuvimos evaluando con los distintos módulos del equipo de poligrafía , añadiendo o quitando ganancia, añadiendo filtros de paso-alto o paso-bajo, y valoramos que tomar nota en la propia aplicación de estas modificaciones pueden ser relevante a la hora de interpretar los datos recogidos. Nuestra aplicación cuenta con un reloj que nos puede ayudar a registrar de forma manual cuándo se realizan las modificaciones, pero, una idea para nuevas líneas futuras, seria añadir la posibilidad de registrar el tiempo y la modificación aplicada en la señal, para cada manipulación, para que quedase todo registrado en el fichero de datos.

Por último, y también durante las pruebas, observamos que, si se están representando varios canales (en tiempo real o provinientes del fichero), la superposición de todos ellos en la gráfica puede entorpecer el análisis de las señales. Si bien la aplicación permite ocultar los canales, podría añadirse también la posibilidad de mover en la gráfica su línea de base (de forma análoga a como se hace en un osciloscopio) para poder visualizar varios canales y evitar su superposición.

7

Referencias

A. Tiwari, R. Cassani, J. -F. Gagnon, D. Lafond, S. Tremblay, & T. H. Falk. (2020). *Prediction of Stress and Mental Workload during Police Academy Training Using Ultra-Short-Term Heart Rate Variability and Breathing Analysis*.

Allen, J., Overbeck, K., Stansby, G., & Murray, A. (2006). Photoplethysmography assessments in cardiovascular disease. *Feature Www.Instmc.Org.Uk*, *80*(3). www.instmc.org.uk

BIOPAC. (n.d.). *Psychophysiology*. Retrieved September 1, 2022, from https://www.biopac.com/application/psychophysiology/

Cooking Hacks. (n.d.). *e-Health Sensor Shield V2.0 for Arduino, Raspberry Pi and Intel Galileo [Biometric / Medical Applications]*. Retrieved September 1, 2022, from https://www.cooking-hacks.com/ehealth-sensor-shield-biometric-medical-arduino-raspberry-pi.html

COULBOURN. (n.d.). *Lab Linc V - Coulbourn Instruments*. Retrieved August 15, 2022, from https://www.coulbourn.com/v/vspfiles/assets/manuals/LabLincVHardwareUserGuide.pdf

Coulbourn Instruments. (n.d.). *Lab Linc V System*.

coulbourn intruments. (n.d.). *LabLincV*. Retrieved August 15, 2022, from http://www.coulbourn.com/v/vspfiles/assets/manuals/LabLincVCatalog.pdf

Hassan, M. M., Alam, M. G. R., Uddin, M. Z., Huda, S., Almogren, A., & Fortino, G. (2019). *Human emotion recognition using deep belief network architecture. Information Fusion*. *51*, 10–18.

Koudelková, Z., Strmiska, M., & Jašek, R. (2018). *Analysis of brain waves according to their frequency*. *12*, 202–207.

L Sörnmo, & P Laguna. (2005). *Bioelectrical signal processing in cardiac and neurological applications*.

Li, S., Liu, L., Wu, J., Tang, B., & Li, D. (2018). Comparison and Noise Suppression of the Transmitted and Reflected Photoplethysmography Signals. *BioMed Research International*, *2018*. https://doi.org/10.1155/2018/4523593

Liu, S. H., Li, R. X., Wang, J. J., Chen, W., & Su, C. H. (2020). Classification of Photoplethysmographic Signal Quality with Deep Convolution Neural Networks

for Accurate Measurement of Cardiac Stroke Volume. *Applied Sciences 2020, Vol. 10, Page 4612*, *10*(13), 4612. https://doi.org/10.3390/APP10134612

National Instruments. (n.d.-a). *Dispositivo de E/S Multifunción*. Retrieved September 19, 2022, from https://www.ni.com/es-es/shop/hardware/products/multifunction-io-device.html

National Instruments. (n.d.-b). *USB-6000 Specifications*. Retrieved September 19, 2022, from https://www.ni.com/docs/en-US/bundle/usb-6000-specs/page/specs.html

NI. (n.d.). *USB-6000 Specifications*. Retrieved August 15, 2022, from https://www.ni.com/docs/en-US/bundle/usb-6000-specs/page/specs.html

Park, J., Seok, H. S., Kim, S. S., & Shin, H. (2022). Photoplethysmogram Analysis and Applications: An Integrative Review. *Frontiers in Physiology*, *12*, 2511. https://doi.org/10.3389/FPHYS.2021.808451/XML/NLM

Stéphane, M. (2009). Discrete Revolution. *A Wavelet Tour of Signal Processing*, 59–88. https://doi.org/10.1016/B978-0-12-374370-1.00007-0

UNE. (n.d.). *UNE-EN 60601-2-47:2015 (Ratificada)*. Retrieved August 31, 2022, from https://www.une.org/encuentra-tu-norma/busca-tu-norma/norma?c=N0055015

Webster, J. G. (2009). *Medical instrumentation : application and design* (4th ed.).

Anexo

Manual de Instalación y Uso

Manual de Instalación

Hardware

La aplicación ha sido desarrollada y probada con un ordenador de modelo Asus A543UA-GQ1693 (procesador Intel Core i5-8250U de 4 núcleos, 8GB de memoria RAM y un SSD de 256GB), una tarjeta de adquisición de National Instrumets (USB-6000) y tres módulos del equipo de poligrafía Lab Linc V: *V75-08 4-CHANNEL EEG AMPLIFIER , V75-04 ISOLATED BIOAMPLIFIER WITH BANDPASS FILTER y V71-40 PULSE MONITOR/OPTICAL DENSITOMETER*, identificados en orden en la Figura 96.

Para la conexión del dispositivo USB-6000 con el equipo Lab Linc V se debe usar un cable de par trenzado, con un mínimo de pares igual al número de canales del USB-6000 que se quieran adquirir. En el extremo del dispositivo USB-600 se conecta cada hilo del par a una salida analógica y el otro a tierra, y, en el otro extremo del cable, cada par al módulo del equipo de poligrafía que se quiera usar, mediante un conector compatible con las salidas de los módulos. Es importante identificar el número del canal al que está conectado cada uno ya que tendremos que indicarlo en nuestra aplicación.

A modo de ejemplo, en las siguientes figuras aparece el conexionado para el caso de querer trabajar con tres módulos del equipo de poligrafía. En la Figura 97 podemos ver la conexión al canal siete (identificado con el número uno en la figura), canal cuatro (numero dos) y canal cero (número tres). En la Figura 98 podemos ver mejor los canales de la tarjeta de adquisición. En el otro extremo del cable, se deben conectar cada uno de los dos pares del cable a un conector (Figura 99 y Figura 100), que se usará para realizar la conexión con a la salida analógica de cada módulo (Figura 101).

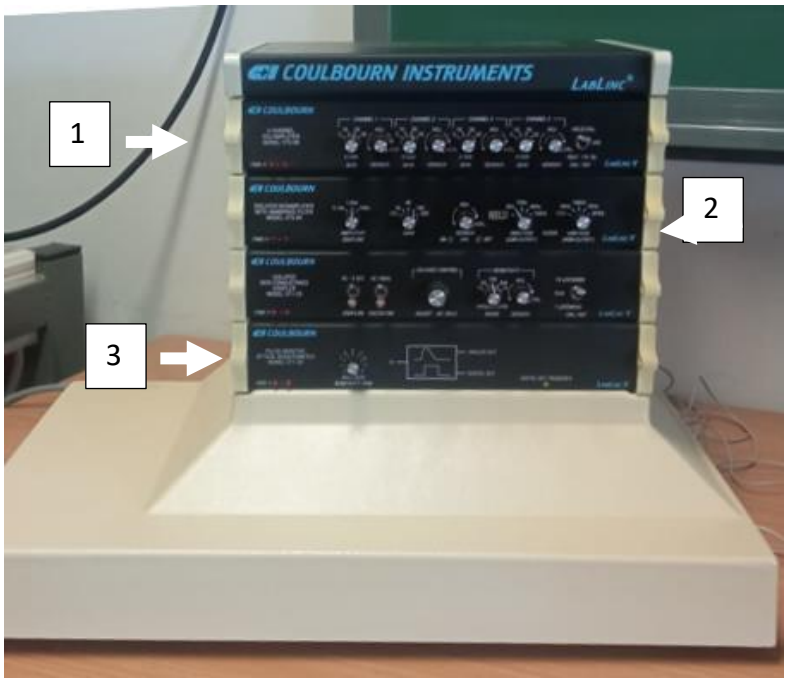


Figura 96. Equipo de poligrafía. [1] V75-08 [2] V75-04 [3] V71-40

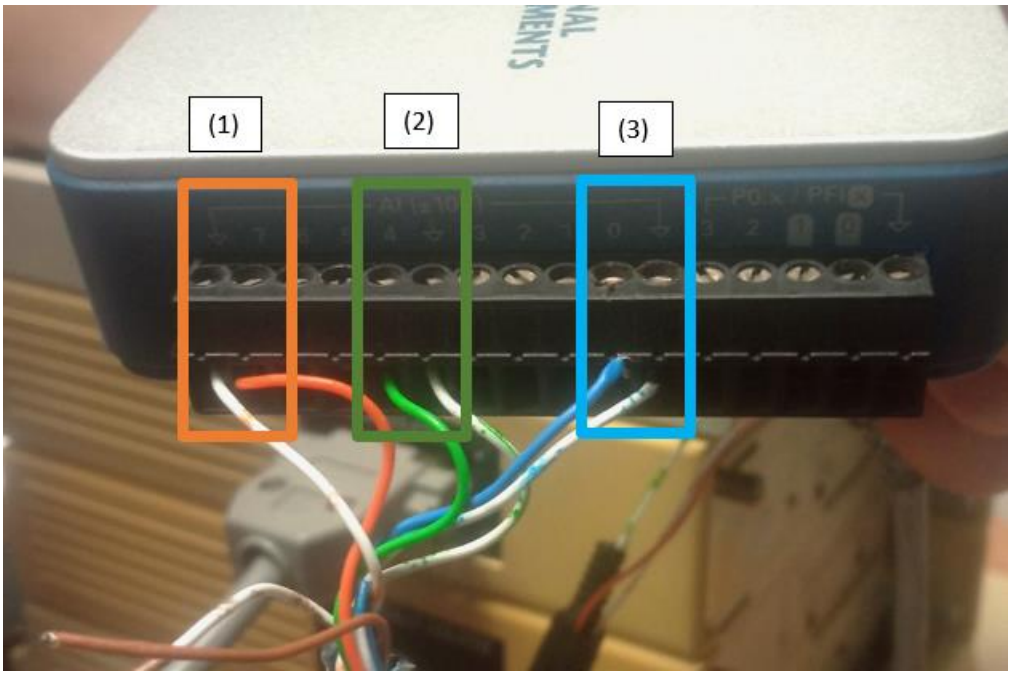


Figura 97. Conexión a los canales de la tarjeta de adquisición.

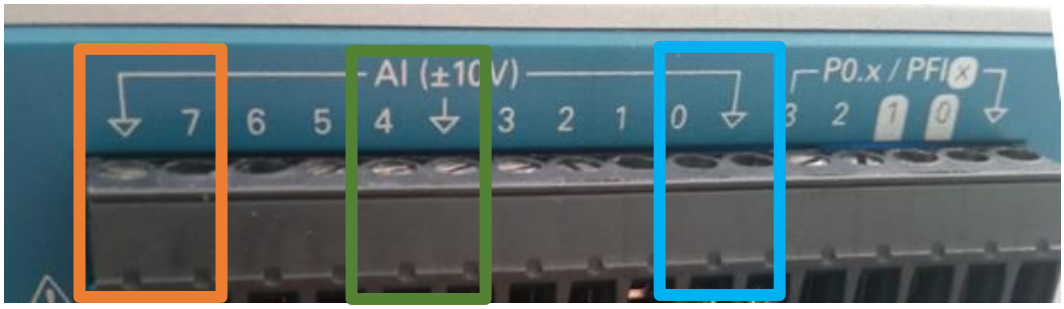


Figura 98. Canales de la tarjeta de adquisición.

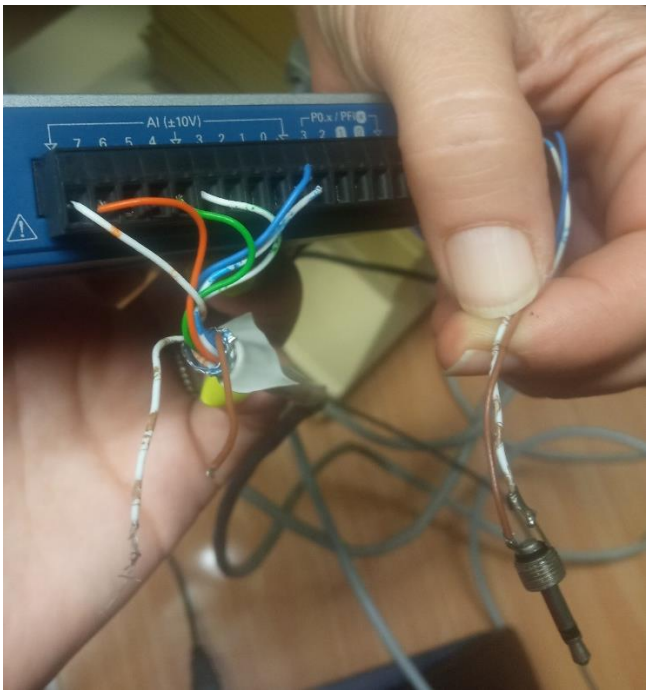


Figura 99. Extremo del cable con el cabezal.

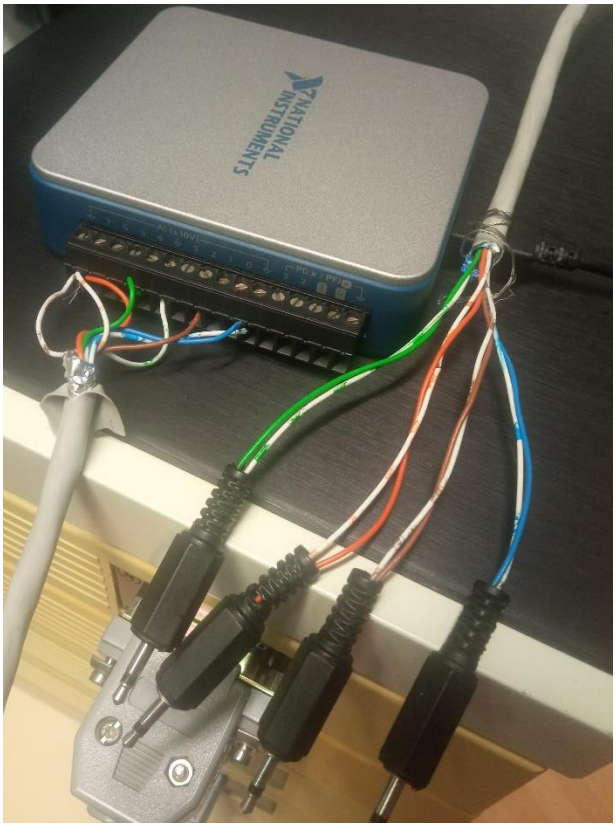


Figura 100. Resultado final de la conexión a la tarjeta de adquisición.

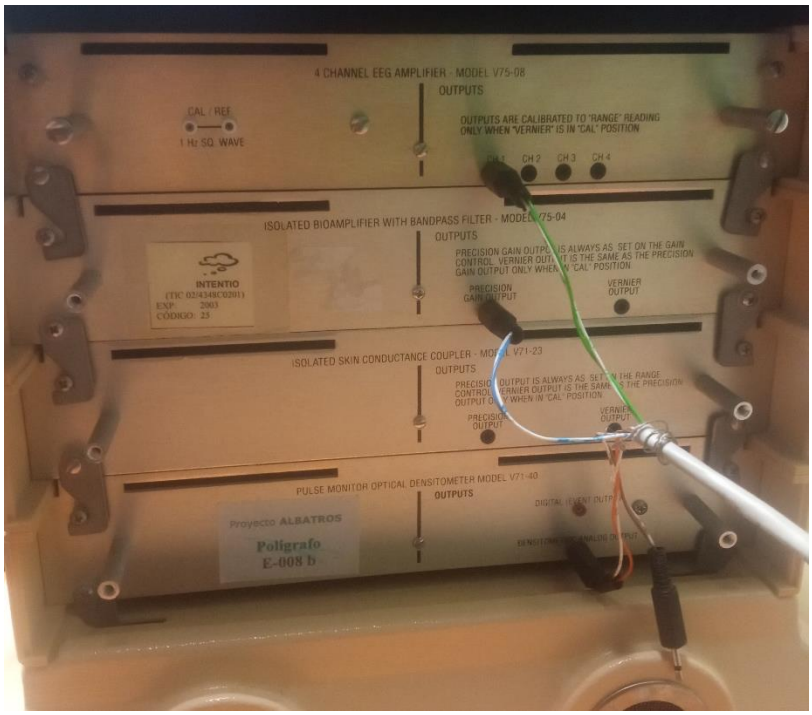


Figura 101. Conexión a la salida analógica de los módulos de LabVic.

Una vez conectada la tarjeta de adquisición a los módulos del equipo de poligrafía, se deben conectar a conectar los sensores. En el lateral de cada modulo se encuentra la ranura destinada a esta conexión tal y como vemos en la Figura 102.

Figura 102. Conexión de los sensores al equipo de poligrafía.

Por último, el dispositivo USB-6000 se conecta al ordenador mediante un cable USB.

Software

La aplicación se ha desarrollado en Windows 10, usando la versión 2020 del sistema de desarrollo LabVIEW de NI, que deberá estar instalado en el ordenador.

Se suministra, por un lado, el código fuente con el formato de un proyecto de LabVIEW (lvproj). Una vez abierto desde el sistema de desarrollo, la aplicación arrancará al ejecutar el vi localizado en la carpeta Main (Figura 103).

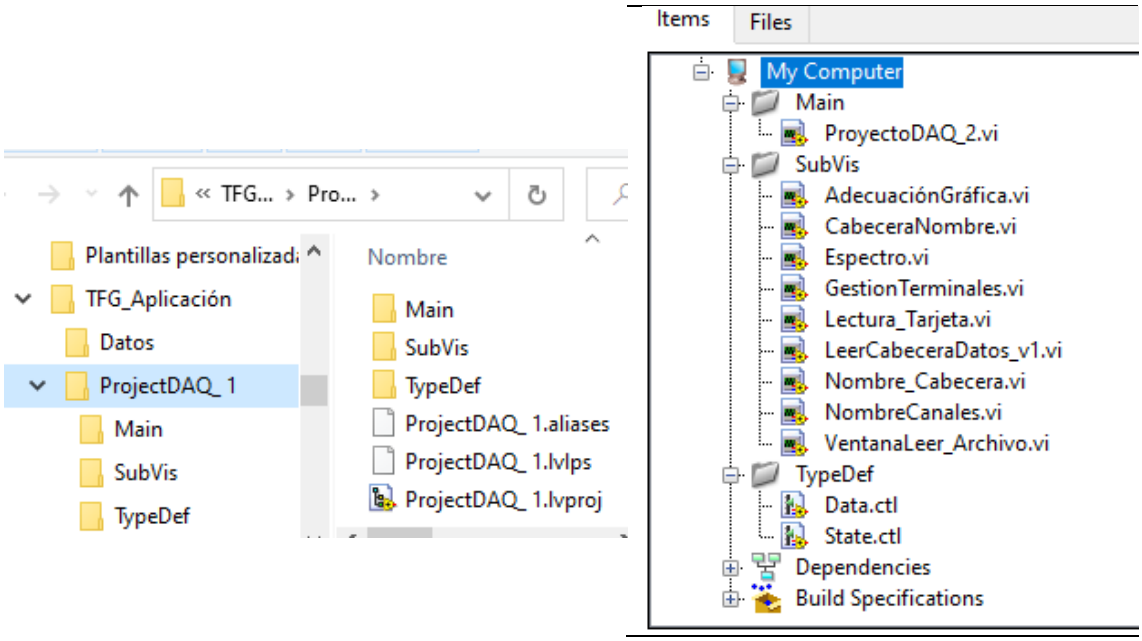


Figura 103. Código fuente: ventanas del explorador de Windows y del proyecto abierto con LabView

Se suministra también un ejecutable (Figura 104, desde el que se podrá arrancar también la aplicación, pero sin que sea necesario tener abierto el sistema de desarrollo (útil si se desea que se use la aplicación sin tener acceso al código fuente).

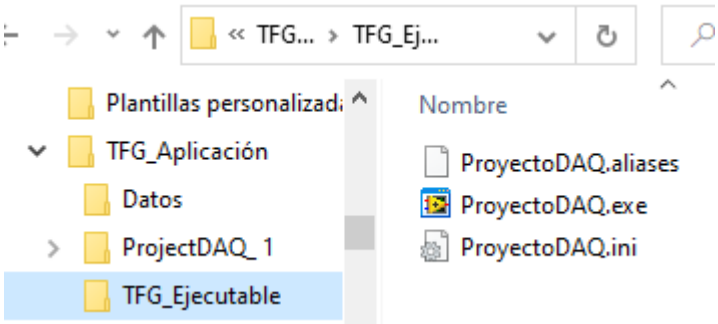


Figura 104 Ejecutable: Ventana del explorador de Windows

La carpeta Datos, en la que el programa automáticamente almacena los datos adquiridos (ver Manual de uso), deberá estar previamente creada en la misma carpeta en la que se haya colocado la carpeta del código fuente o del ejecutable, tal como aparece en las Figura 103 y Figura 104.

Manual de uso

La aplicación, tras arrancar, le permitirá:

1. Adquirir las señales provinientes del equipo Lanlinc V previamente conectadas al dispositivo USB-6000 (ver Instalación), visualizarlas y si así lo desea, almacenarlas en un fichero. Vea los detalles en el apartado **Adquiriendo**.
2. Visualizar las señales adquiridas y almacenadas en esta u en otras sesiones anteriores. Vea los detalles en el apartado **Visualizando desde fichero**.
3. Salir de la aplicación. Vea los detalles en el apartado **Saliendo.**

La aplicación ofrece dos paneles Informativos (Vea los detalles en el apartado Paneles Informativos). Uno permite manipular ciertos parámetros de las gráficas donde se visualizan las señales (Vea los detalles en el apartado **Paneles informativos**) y el otro avisa de los posibles errores que puedan ocurrir durante la ejecución (Vea los detalles en el apartado **Errores**).

Arrancando:

- Identificación. Se debe añadir el nombre del usuario para poder continuar.

Adquiriendo:

En la Figura 105 aparece la sección del interfaz de usuario relacionada con la adquisición de datos. A continuación se muestra la operatoria a seguir haciendo referencia a los controles enumerados en la figura.

- Adquirir datos (1). Botón que ejecuta la adquisición de datos. Antes de pulsar este botón se debe configurar las siguientes caracteristicas según el interes:
    - (2) Guardar. El botón de guardar vendrá marcado por defecto, en caso de no querer crear un fichero con los datos adquiridos, deseleccionar dicho botón.
    - (3) Canales. Por lo general saldrán los ocho canales disponibles para el puerto que esté conectado la tarjeta. Seleccionar el canal o canales de interes.
    - (4) Frecuencia de muestreo y ancho de ventana. Configurar según el interés teniendo en cuenta las especificaciones de la tarjeta de adquisición.
    - (5) Max/Min. No pueden ser modificados. Definidos con el valor ± 10V correspondiente al rango de la tarjeta de adqusición USB-6000.
    - (6) Reloj. Se iniciará una vez comience la adquisición y se parará en cuanto se pare. Sirve para tomar registro de la adquisición.
    - (7) Parar adquisición. Se habilitará una vez se inicie la adquisición.

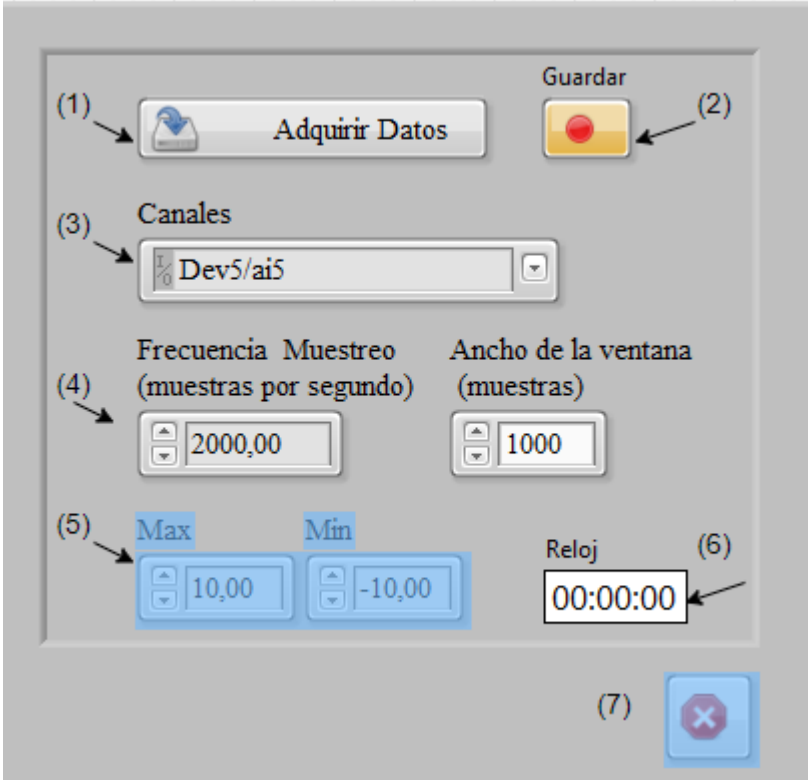


Figura 105. Sección para la adquisición de datos.

Una vez pulsado el botón de adquisición (1) se mostrará una pantalla para añadir los nombres identificativos de los canales seleccionados. **Es obligatorio rellenar estos nombres**. En la Figura 106 se muestra un ejemplo para dos canales. Para continuar con la adquisición, pulsar el check verde.

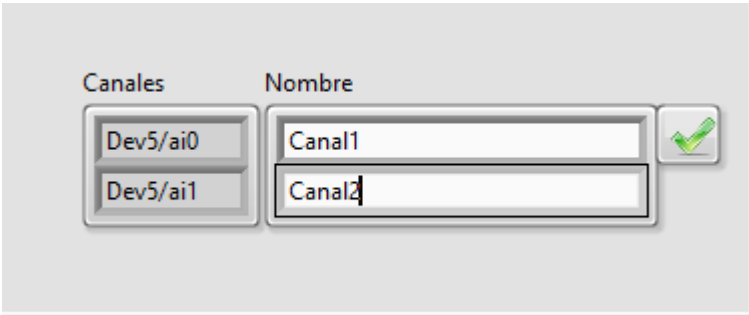


Figura 106. Subpanel para nombrar los canales.

Los datos que se están adquiriendo y procesando aparecerán representados en las gráficas que se muestran en la Figura 107.

- Gráfica Chart (8). Muestra el historial de registro. Viene definido un número de muestras de 10.000 (el tiempo total representado vendrá dado por tanto por la configuración de la adquisición). En esta versión este parámetro no es modificable por el usuario
- Gráfica Graph (9). Muestra cada registro. El tiempo de esta gráfica dependerá de los valores introducidos en el panel de adquisición (4).
- Espectro (10). Se mostrará el espectro de la señal adquirida en la misma ventana temporal que la programada para la Graph.

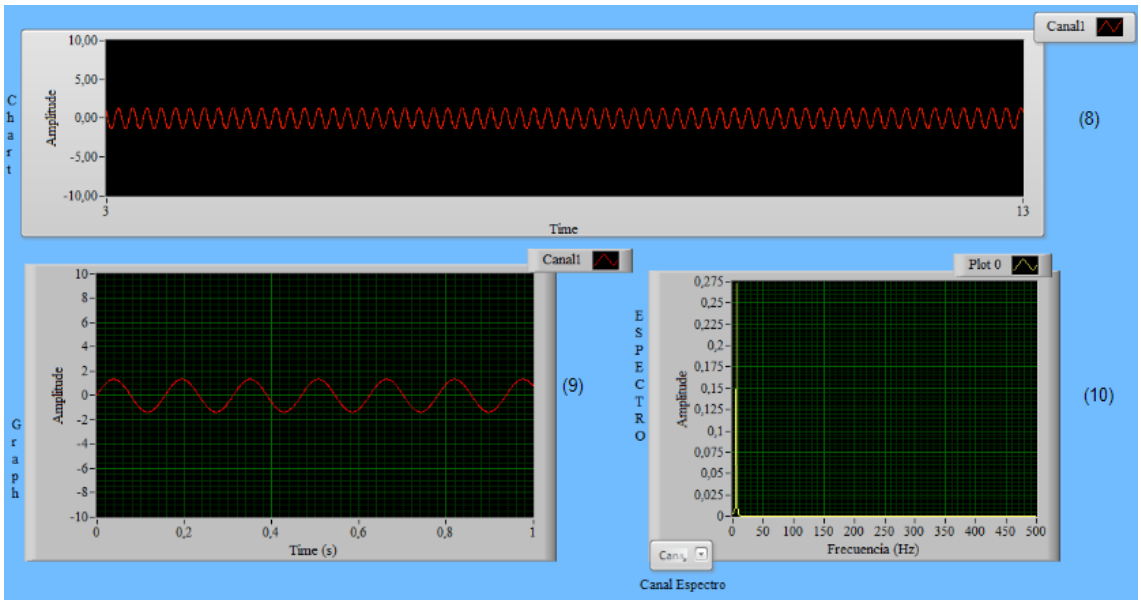


Figura 107. Gráficas de la interfaz de usuario.

Si se ha pulsado el botón (2) de la Figura 105, Guardar, los datos se iran almacenando en un fichero. Tanto la carpeta destino (Datos), que debe estar previamente creada en paralelo con la carpeta del proyecto, como el nombre del fichero viene predefinido por el programa. El nombre del fichero es el resultado de la concatenación del día, mes, hora, minutos y nombre introducido en la identificación.

El fichero está formado por una cabecera que contiene el nombre del usuario, la fecha de la adquisición, nombre de los canales y la frecuencia de muestreo, seguido por los datos adquridos organizados por columnas por canal. En la Figura 108, podemos ver un ejemplo de una adquisición de un canal realizada el diecinueve de septiembre por el usuario de nombre "Nombre Usuario", tal como aparece cuando se abre el fichero con el Bloc de notas de Windows.

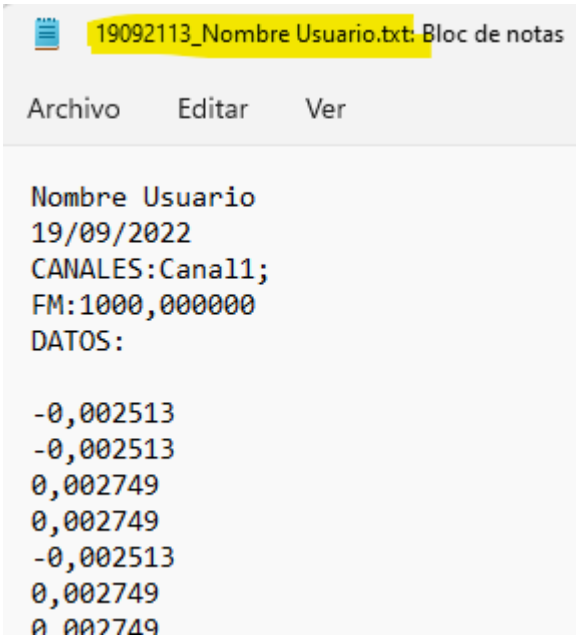


Figura 108. Nombre y cabecera del fichero de almacenamiento.

Visualizando desde fichero:

En la Figura 109 y Figura 110 aparece la sección del interfaz de usuario y la gráfica relacionada con la visualización desde fichero.

- Leer Fichero (11). Botón que ejecuta la lectura de un fichero. Antes de pulsar este botón se debe introducir la ruta path del fichero que se desea leer (12).

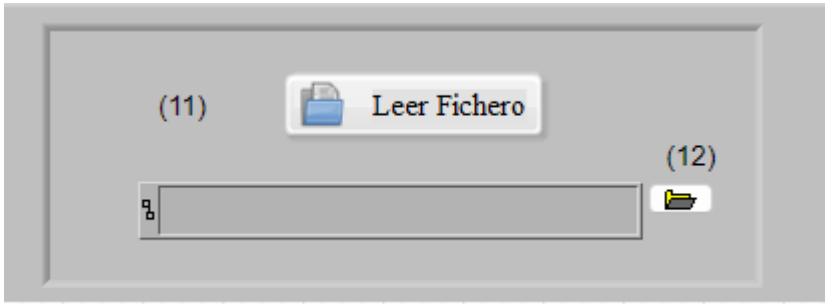


Figura 109. Sección para la lectura de un fichero.

- Gráfica Graph de Lectura (13). Esta gráfica aparecerá en un subpanel cuando se accione el botón de Leer Fichero (8). Una vez se quiera continuar con otra acción se debe cerrar el subpanel dándole al botón (14).

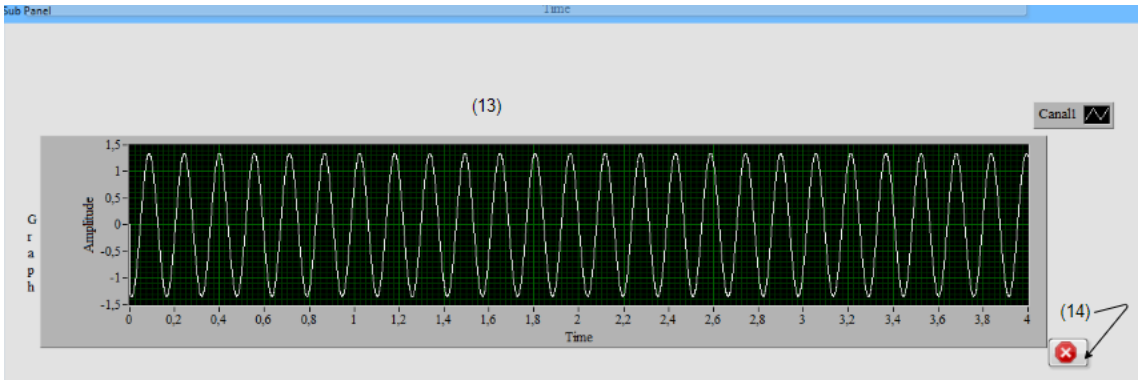


Figura 110. Gráfica Graph de lectura.

Saliendo:

- Salir. Botón que apaga la aplicación. Una vez se pulse este botón se mostrará una ventana de confirmación para apagar la aplicación o volver (Figura 111).

Figura 111. Botón Salir

Paneles informativos:

- Usuario (15) viene el nombre de identificación y Panel informativo (16) el programa va advirtiendo y guiando al usuario durante el uso de la aplicación.

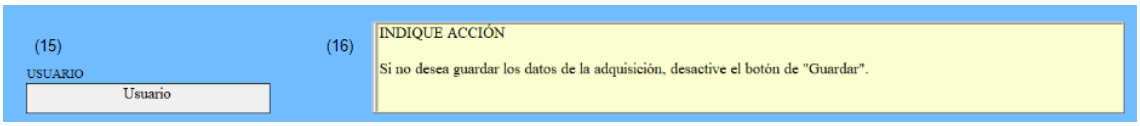


Figura 112. Cabecera de la interfaz de usuario.

- Leyenda de las gráficas (17). En el lado superior derecho de cada gráfica podemos ver un panel sobre los canales que está siendo representados, indicando el color, forma y nombre de cada uno. En el siguiente punto hablaremos de acciones que se pueden realizar sobre estos paneles (Figura 113).

- Canal actual de la gráfica del Espectro. En el caso de representar más de un canal podemos ver el espectro de cada una de ellas cambiando en la sección (18) que se muestra en la Figura 113.

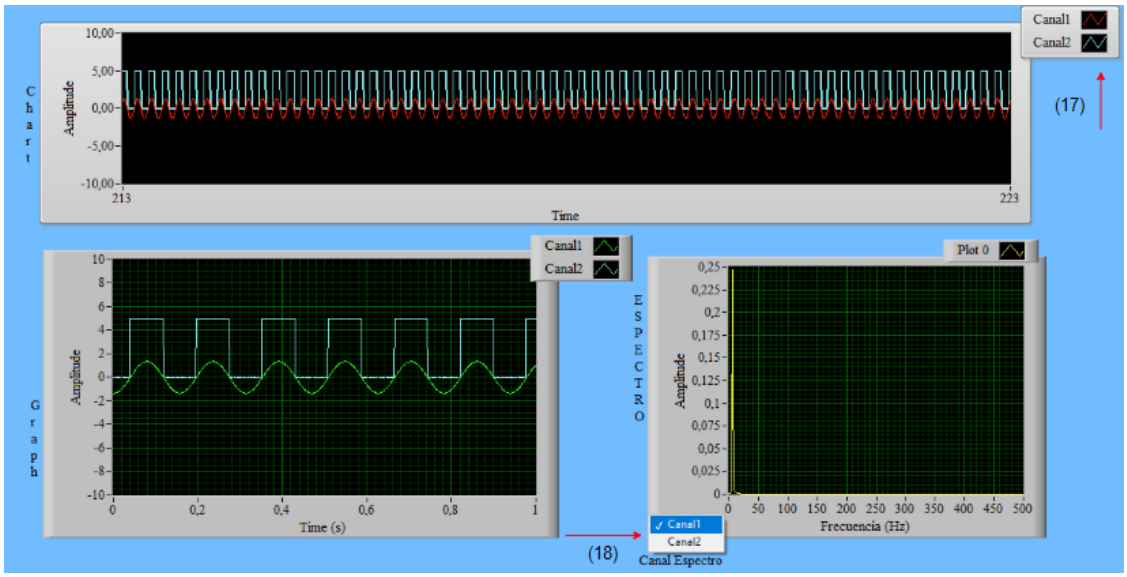


Figura 113. Información sobre las gráficas.

Sobre las gráficas:

Algunas acciones que se pueden hacer sobre las gráficas son:

- Cambiar la visibilidad de las señales. Si estamos mostrando más de una señal, durante la adquisición podemos cambiar la visibilidad de cada una de ellas, es decir, si estamos viendo dos canales podemos indicar que solo se muestre una. Esto se hace pulsando el botón derecho sobre el canal que quieras cambiar y dándole a "Plot Visible". En el caso que se muestra en la Figura 114 si le damos desaparecería el canal1 de la gráfica como podemos ver en la Figura 115.

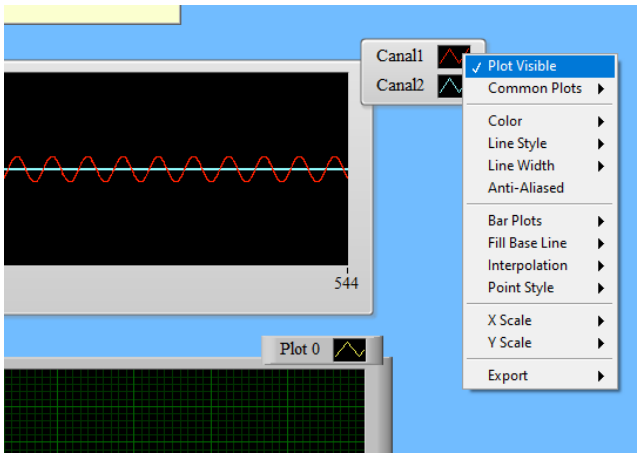


Figura 114. Cambiar visibilidad de los canales.

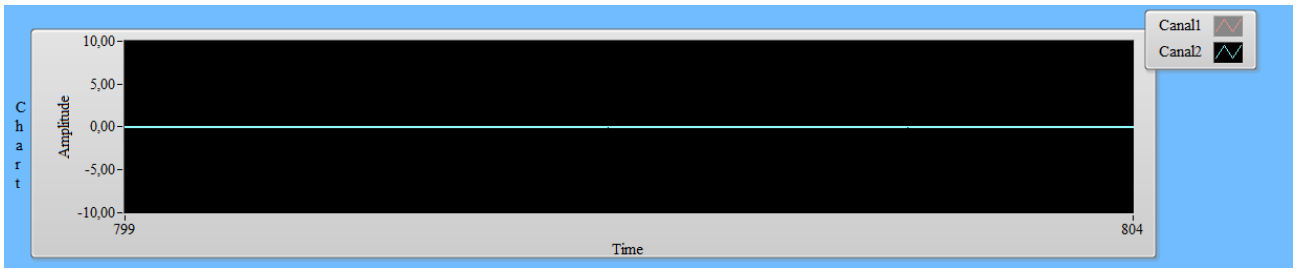


Figura 115. Visibilidad de un canal.

- Ampliar datos en la gráfica. Si desea ver con más detalle los valores obtenidos en un tiempo concreto, puede modificar el eje X de la gráfica según su interés. Tan solo debe seleccionar los valores de los extremos e introducir el valor que le interesa. En la Figura 116 podemos ver un ejemplo, donde se ha modificado el rango menor de tiempo a dos. La gráfica origen es la representada en la Figura 110.

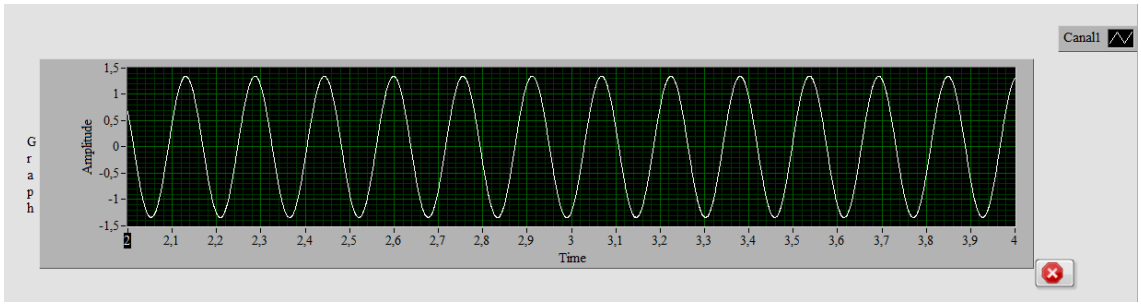


Figura 116. Ampliación de los datos en una gráfica.

- Cambiar la forma de las ondas. Se puede seleccionar distintas formas de representación de los datos. Para ello hay que pulsar el botón derecho sobre el canal que desea cambiar y elegir la forma que desee. (Figura 117)

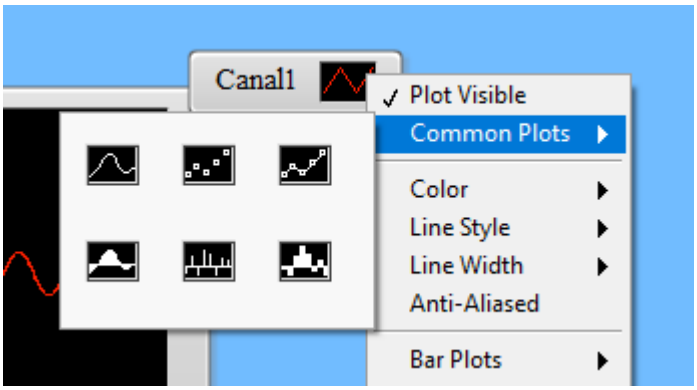


Figura 117. Cambiar forma de la señal.

Errores:

Durante el uso de la aplicación pueden saltar distintas alertas en el panel informativo (16) por los siguientes motivos:

- Mala configuración de la tarjeta. En el caso que se añada una frecuencia de muestreo superior a la velocidad máxima para el número de canales indicados.
- Mala conexión de la tarjeta. En el caso que la tarjeta de adquisición no esté bien conectada al PC o se hayan seleccionado canales que no están conectados.
- Cancelación de la operación de adquisición. Cuando se abra la ventana Windows que muestra la creación del fichero y, en vez de continuar, se cancele cerrándola o dándole al botón de cancelar.
- Error en el formato del fichero de lectura. Se mostrará un aviso cuando el formato del fichero no sea el correcto y cuando el fichero no contenga datos. En caso de añadir un fichero con otro formato se mostrará el mensaje que aparece en la Figura 118 y en caso de que el fichero este vacío se mostrará el mensaje de la Figura 119.

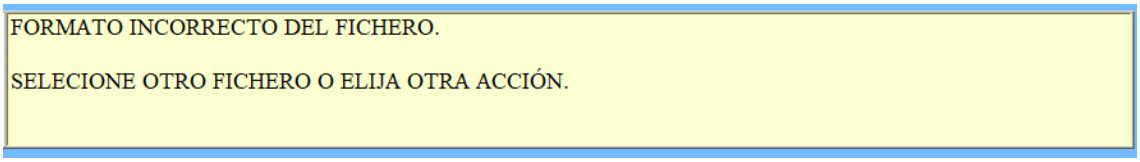


Figura 118. Aviso sobre el formato del fichero.

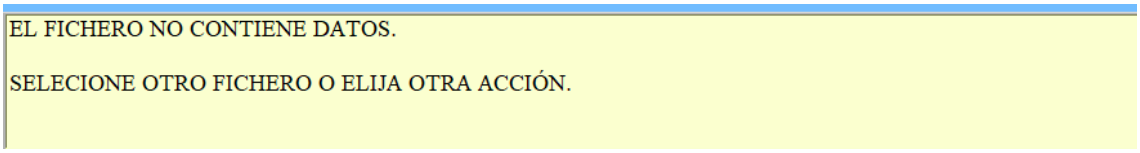


Figura 119. Aviso sobre el contenido del fichero.

Podemos ver los demás mensajes que se muestran en la sección *Warning* del apartado 5.2.

Si saltan algunas de estas alertas, el usuario podrá modificar los datos configurados y continuar con la acción, pero, en caso de que se muestre un mensaje de error (Figura 120), el programa se apagará.

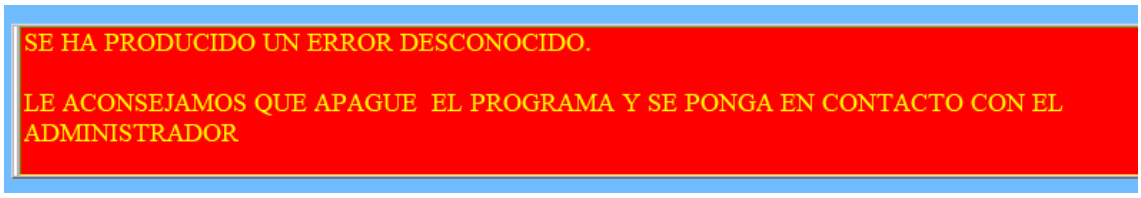


Figura 120. Mensaje de error.