UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE INGENIERÍA

"Diseño y prototipo de un sistema ECG portátil utilizando el microchip AD8232 y plataforma de código abierto"

T E S I S

que presenta para obtener el grado de MAESTRO EN CIENCIAS

Daniel Cuevas González

DIRECTOR DE TESIS:
Dr. Miguel Bravo Zanoguera

Mexicali, B. C. **Agosto 2018**

AGRADECIMIENTOS

A mi familia, que me ha apoyado durante todo este trayecto, especialmente a mis padres que me han transmitido valores, estudios, y me han formado para bien.

A mis maestros, que me apoyaron durante esta investigación, especialmente a mi director de tesis que cumplió con su función para guiar, ayudar, y contagiar el interés en el desarrollo y estudio de nuevas tecnologías. Su compromiso con su trabajo y deseos de compartir y mejorar a los jóvenes mediante el conocimiento y formación

A mis compañeros y amigos de posgrado, con los que compartí experiencias, estudios, y conocimientos.

A la Universidad Autónoma de Baja California, Facultad de ingeniería, por permitirme ingresar al programa de posgrado.

Al Consejo Nacional de Ciencia y Tecnología (CONACYT), por tener (y brindarme) programas de apoyo económico para jóvenes con talento e interés en el desarrollo de ciencia y tecnología.

RESUMEN

El desarrollo de dispositivos médicos para el cuidado de la salud es uno de los campos de investigación con gran relevancia, se conoce la necesidad continua de actualizar y mejorar el cuidado y atención de la salud, con la aparición de enfermedades es fundamental desarrollar nuevos aparatos que sirvan para el tratamiento de estas y más importante aún, su prevención.

En las últimas décadas se ha observado un incremento alarmante en la población que presenta problemas y enfermedades cardiovasculares siendo esta la principal causa de muerte a nivel mundial, que por falta de conocimiento o identificación del problema suelen agravarse y en el peor de los caso provocar la muerte [1]. Una tendencia que ayuda a afrontar el problema es conocida como: "salud en el hogar" o "salud móvil", que entre sus servicios busca desarrollar dispositivos médicos de uso personal de bajo costo con el uso de la tecnología como: teléfonos inteligentes, sensores de monitoreo, y aplicaciones software, con el fin de monitorear, transmitir, o almacenar los datos del usuario y así tener acceso a su condición de salud en todo momento [2].

Actualmente las compañías de semiconductores han desarrollado microchips multifunción conocidos como AFEs ("*Analog Front-End*" por sus siglas en inglés, o "Etapa inicial Analógica" en español) para la adquisición y filtrado de señales electrofisiológicas, que proporcionan al usuario un amplio campo de aplicación en los diferentes métodos de adquisición de bioseñales, como son ECG, EMG, EEG, y EOG. Estos nuevos chips permiten el desarrollo de equipo de monitoreo de bajo costo, fácil de portar, y de acceso digital.

Durante esta investigación se evaluó el funcionamiento del microchip AD8232 con filtros hardware para sus posibles aplicaciones como: monitor cardíaco, monitor *fitness*, monitor cerca del pecho. Se desarrolló el circuito ECG en breadboard para implementar la interfaz digital, y un menú de operación para los modos de: transmisión de datos por cable serial, grabación por tiempo prolongado en SD, y transmisión bluetooth en PC, y teléfono inteligente utilizando una plataforma de código abierto. Además se evaluó la capacidad de la plataforma Arduino para la utilización de filtros digitales (software) mediante ecuaciones de diferencias.

El resultado obtenido de este trabajo fue el prototipo de un sistema ECG portátil de tiempo prolongado con transmisión serial, grabación en SD, y transmisión bluetooth. Que proporciona una señal ECG útil para aplicaciones de monitoreo, y cumple con la normativa de seguridad eléctrica y diseño de equipo médico.

Palabras clave—**Electrocardiógrafo, AD8232, Home Health, Mobile Health, Arduino.**

ABSTRACT

The development of medical devices for health care is one of the fields of research with great relevance, it's known the continuous need to update and improve the care and health care, with the appearance of diseases it's fundamental to develop new devices that they serve for the treatment of these and more importantly, their prevention.

In the last decades there has been an alarming increase in the population that presents cardiovascular problems and diseases, being these the main cause of death worldwide, which due to lack of knowledge or identification of the problem usually aggravate and in the worst case cause death [2]. A trend that helps to reduce the problem is known as: "health in the home" or "mobile health", which among its services is aimed to develop low-cost personal medical devices with the use of technology such as: smartphones, monitoring sensors, and software applications, for monitor, transmit, or store the user's data and thus have access to their health condition at all times [1].

Currently semiconductor companies have developed multifunction microchips known as AFEs ("Analog Front-End" for its acronym in English, or "Initial Analog Stage" in Spanish) for the acquisition and filtering of electrophysiological signals, which provide the user with a wide field of application in the different methods of acquisition of biosignals, such as ECG, EMG, EEG, and EOG. These new chips allow the development of low cost, easy to port, and digital access monitoring equipment.

During this investigation, the functioning of the AD8232 microchip with hardware filters was evaluated for possible applications such as: cardiac monitor, fitness monitor, and chest monitor. The ECG circuit was developed in breadboard to implement the digital interface, and an operating menu for the modes: data transmission by serial cable, long-time recording in SD, and bluetooth transmission in PC, and smart phone using a platform of Open Source. The capacity of the Arduino platform was evaluated for the use of digital filters (software) through differential equations.

The result obtained from this work was a prototype of a long-time portable ECG system with serial transmission, SD recording, and bluetooth transmission. It provides a useful ECG signal for monitoring applications, and complies with electrical safety regulations and medical equipment design.

Keywords- **Electrocardiograph, AD8232, Home Health, Mobile Health, Arduino.**

ÍNDICE TEMATICO

Página

ÍNDICE DE FIGURAS

Página

CAPÍTULO 1

CAPÍTULO 2

CAPÍTULO 3

CAPÍTULO 4

ÍNDICE DE TABLAS

CAPÍTULO 1

CAPÍTULO 2

CAPÍTULO 3

CAPÍTULO 4

ÍNDICE DE CÓDIGOS

INTRODUCCIÓN

El desarrollo de dispositivos médicos para el cuidado de la salud es uno de los campos de investigación con gran relevancia, se conoce la necesidad continua de actualizar y mejorar el cuidado y atención de la salud, con la aparición de enfermedades es fundamental desarrollar nuevos aparatos que sirvan para el tratamiento de estas y más importante aún, su prevención.

En las últimas décadas se ha tenido un incremento alarmante en la población que presenta problemas y enfermedades cardiovasculares siendo estas la principal causa de muerte a nivel mundial, que por falta de conocimiento o identificación del problema suelen agravarse y terminar incluso en muerte por infarto o afecciones cardíacas [1]. Una tendencia que ayuda a afrontar el problema es el desarrollo de dispositivos médicos de uso personal y de "salud en el hogar" o "salud móvil", de bajo costo y con el uso de la tecnología como: teléfonos inteligentes, sensores de monitoreo, y aplicaciones software registren transmitan, o almacenen los datos del usuario para tener acceso a su condición de salud en todo momento [2].

El desarrollo de este equipo personal es una alternativa al utilizado en el sector hospitalario, y puede otorgar un registro de la actividad y signos vitales del usuario a largo plazo o tiempo prolongado mientras realizan sus actividades diarias. Que pueden captar eventos que ocurren con poca frecuencia o que ocurren bajo circunstancias específicas, y con esto tener una perspectiva más realista y precisa al momento de realizar un diagnóstico.

Para el desarrollo exitoso de dispositivos de monitoreo personal deben ser de tamaño reducido, tener una calidad de la señal apropiada para la aplicación requerida, contar con suministro de energía a largo plazo, obtener registro de datos a largo plazo o transmisión en tiempo real, y ser de bajo costo de modo que sean más accesibles a la población. Esto ha sido considerado por las compañías de semiconductores para crear circuitos integrados de modo que sea más fácil producir este tipo de dispositivos.

Las compañías de semiconductores han desarrollado microchips multifunción conocidos como AFEs ("*Analog Front-End*" por sus siglas en inglés, o "etapa inicial analógica" en español) para la adquisición y filtrado de señales electrofisiológicas. La

compañía Analog Devices desarrolló el chip AD8232 con el cual se pueden sintetizar la etapa de filtrado y acondicionamiento de la señal de electrocardiograma, además de ser amigable con la etapa de procesamiento digital.

En este proyecto se propone el diseño de un monitor ECG portátil de uso personal y tiempo prolongado, utilizando el microchip AD8232 para la etapa inicial analógica. Con modo de operación para transmisión por cable serial, grabación de datos por tiempo prolongado en SD card, y transmisión bluetooth a PC y dispositivos inteligentes utilizando una plataforma de código abierto.

PLANTEAMIENTO DEL PROBLEMA

Hoy en día uno de los problemas de salud más frecuentes son las enfermedades cardiovasculares y problemas al corazón, siendo la principal causa de muerte a nivel mundial. La situación es bastante alarmante ya que no solo se ha visto un incremento en el número de pacientes que padecen estas enfermedades si no también un incremento en pacientes cada vez más jóvenes, que al no conocer su condición de salud o no tener acceso a la salud pública, no atienden esta problemática [2]. Una tendencia que ayuda a afrontar esta problemática con el uso de tecnología y dispositivos inteligentes es la “Salud en casa” o “Salud móvil”. Esta tendencia se ha generado por la necesidad de crear dispositivos de uso personal y portátiles para monitorear las señales biológicas que puedan ser utilizadas de manera continua para la prevención y atención a las enfermedades. Este tipo de dispositivos son una alternativa al del sector hospitalario, siendo su principal aplicación el monitoreo por tiempo prolongado y prevención de enfermedades. Entre sus características buscan un tamaño reducido, calidad de la señal, energía a largo plazo, obtener registro de datos a largo plazo o transmisión en tiempo real.

PREGUNTA DE INVESTIGACI ÓN

¿Puede proporcionar el circuito integrado AD8232, como etapa inicial analógica, una señal con la calidad y características necesarias para la elaboración de un sistema de electrocardiografía portátil de uso personal con plataforma de código abierto para registro de datos?

HIPÓTESIS

El circuito integrado AD8232 utilizado como etapa inicial analógica proporciona una señal amplificada, filtrada, y útil para un sistema de electrocardiografía portátil de uso personal con plataforma de código abierto con registro y transmisión de datos.

OBJETIVOS DE INVESTIGACIÓN

OBJETIVO GENERAL

Fabricar un prototipo de un sistema ECG portátil con registro y transmisión inalámbrica de datos, utilizando como etapa inicial analógica el chip AD8232 y plataforma de desarrollo de código abierto.

OBJETIVOS ESPECÍFICOS

- Evaluar y validar el funcionamiento, alcance y limitantes del microchip AD8232, para monitor de la señal ECG.
- Caracterizar la plataforma de código abierto Arduino con programación de bajo nivel (C, ensamblador), para aumentar su rendimiento en la aplicación de un sistema de adquisición de datos y aplicación de filtros digitales el tiempo real.
- Diseñar el circuito de un sistema de ECG portátil con los siguientes modos de operación: transmisión serial, grabación en SD card, y transmisión bluetooth.

JUSTIFICACIÓN

Con el apoyo de las nuevas tecnologías, uso de nubes en internet, y dispositivos inteligentes, se ha observado en estos últimos años una nueva forma monitoreo de señales biomédicas, además de la forma tradicional con instrumentos voluminosos se están empleando sistemas portátiles de uso continuo para adquisición y transmisión inalámbrica de datos, generando esta tendencia llamada "salud en casa" y "salud móvil" [2].

En respuesta a esta tendencia las empresas de semiconductores han desarrollado circuitos integrados, conocidos como AFEs ("*Analog Front-End*" por sus siglas en inglés y como etapa inicial analógica en español). Es importante hacer énfasis en la utilización de nuevas tecnologías como son los AFEs ya que integra las diferentes etapas que se utilizaban en módulos antiguos y voluminosos en una sola unidad miniatura (microchip), dando así facilidad en el diseño y fabricación de este tipo de dispositivos portátiles.

Esta investigación busca contribuir mediante el desarrollo de un dispositivo portátil, de fácil uso, y bajo costo que monitoree los signos vitales de electrocardiografía (ECG), que registre estos datos de manera que el usuario pueda tener acceso a su registro de forma fácil, y si lo requiere, llevar con un médico. Lo que permite tomar medidas preventivas de salud y mejorar la calidad de vida del usuario.

CAPÍTULO 1

ANTECEDENTES

1.1 Historia del Electrocardiograma.

Desde la antigüedad la adquisición de biopotenciales tuvo importancia iniciando en los siglos XVII y XVIII, donde los científicos comenzaron a desarrollar instrumentos para medir cosas que no se podían percibir con los sentidos normales. Entre los científicos importantes, Kölliker y Müller en 1856 registraron el potencial de acción del músculo cardíaco, demostrando al poner un nervio ciático sobre el corazón de una rana y los potenciales estimulaban el nervio demostrando así la existencia de biopotenciales del corazón.

En 1872 otro científico importante el físico francés Gabriel Lippman inventó un electrómetro capilar (ver Figura 1.1) con el cual le dieron el premio nobel de física en 1908, este instrumento consistió en un tubo fino de vidrio con una columna de mercurio bañada con ácido sulfúrico. Y al detectar potenciales eléctricos el mercurio representa ese potencial mediante movimiento. Este nuevo instrumento se utilizó por el fisiólogo francés Etienne-Jules Marey en 1876 para registrar por primera vez la actividad eléctrica de un corazón de rana [3].

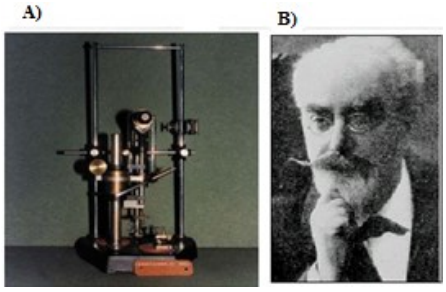


Figura 1.1 A) Electrómetro capilar. B) Gabriel Lippmann inventor del electrómetro capilar [3].
Fuente: Memoria Holter, pag. 12 Universidad Politécnica de Catalunya

1.1.1 Principio de funcionamiento ECG Augustus D. Waller.

En 1887 el médico y fisiólogo Augustus D. Waller perfeccionó el método utilizando un instrumento desarrollado a partir del electrómetro capilar de Lippmann llamado galvanómetro capilar y con este registró el primer electrocardiograma en humanos, demostró que la actividad eléctrica del corazón podía ser registrada desde la superficie del pecho del paciente [3].

El galvanómetro capilar consistía en un tubo de vidrio terminado por una extremidad capilar muy fina, parcialmente lleno de mercurio sobre la cual reposaba una capa de ácido sulfúrico diluido (ver Figura 1.2).

Figura 1.2 Galvanómetro capilar creado a partir del electrómetro capilar de Lippmann [3].
Fuente: Memoria Holter, pag. 12 Universidad Politécnica de Catalunya

El ácido sulfúrico y el mercurio tenían contacto con los electrodos y al percibir la variación de los potenciales eléctricos sufría una modificación en la tensión superficial, y hacían que el menisco que separaba el mercurio y ácido sulfúrico se desplazara hacia arriba o abajo del tubo capilar. La zona de separación de ambos líquidos se iluminaba con una luz externa y la imagen era proyectada por una lente apocromática, la cual se proyectaba sobre una hendidura vertical y era grabada en una placa de papel fotográfico a una velocidad constante. Waller con su aportación de ECG pudo registrar 2 ondas como puede verse en la figura 1.3 y las nombró V1 y V2 para indicar la actividad ventricular [4].

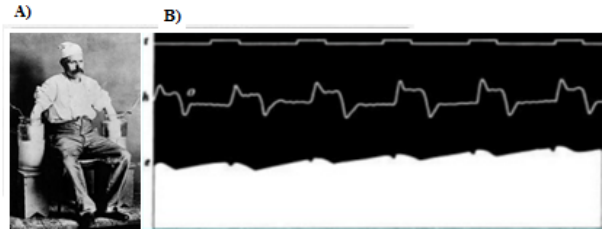


Figura 1.3 A) Electrodos de solución salina de experimentos con ECG por Augustus Waller y B) onda obtenida [4].
Fuente: Revista Médica Chile, Historia de la medicina. Einthoven. El hombre y su invento.

1.1.2 Principio de funcionamiento ECG Willem Einthoven

Einthoven estudió y perfeccionó el método de Waller, el electrómetro capilar fue utilizado cerca de 12 años y obtuvo resultados como la figura 1.4 imagen A y posteriormente al utilizar el galvanómetro de cuerdas lograría obtener la señal de la imagen B.

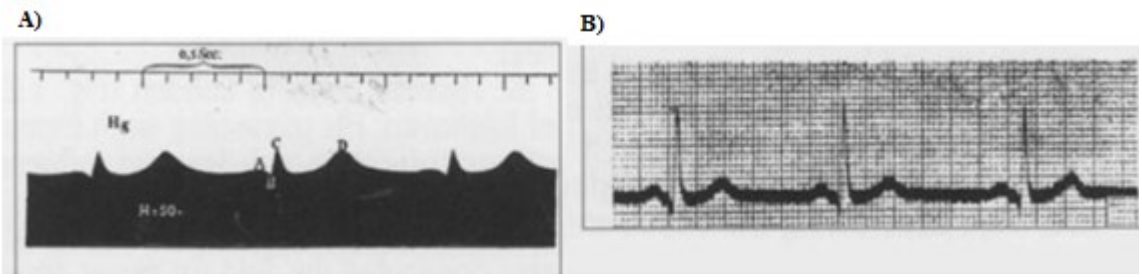


Figura 1.4 A) Señal de ECG con electrómetro capilar y B) Señal de ECG con galvanómetro de cuerdas experimentos de Einthoven [4].
Fuente: Revista Médica Chile, Historia de la medicina. Einthoven. El hombre y su invento. Pag 261-264

Einthoven sabía que las corrientes del corazón se movían a través del medio extracelular y encontró la forma de graficar estas corrientes, mediante la medición de diferencias de potencial entre partes diferentes del cuerpo se obtiene una corriente eléctrica cambiante y esta se puede representar gráficamente. Y así cada derivación mide la diferencia de voltaje, cuyo instrumento utilizado en esa época era el galvanómetro inventado por Galvani, pero esté no tenía la sensibilidad suficiente para detectar milivoltios que se generan en el cuerpo humanos a través del electrodo.

El galvanómetro se compone de 2 imanes permanentes, un conductor y una aguja. La corriente pasa a través del conductor, y causa una interacción con los magnetos por lo tanto la aguja se mueve y dicho movimiento es proporcional a la corriente que circula por el conductor. Dado que el sistema no es muy sensible como para detectar los cambios en las señales eléctricas del corazón, Einthoven tras años de estudio hizo un aparato con el mismo principio de funcionamiento pero sustituyó el conductor grueso por un hilo metálico muy delgado y liviano con el cual logró detectar corrientes muy pequeñas, el modelo de este aparato se muestra en la figura 1.5.

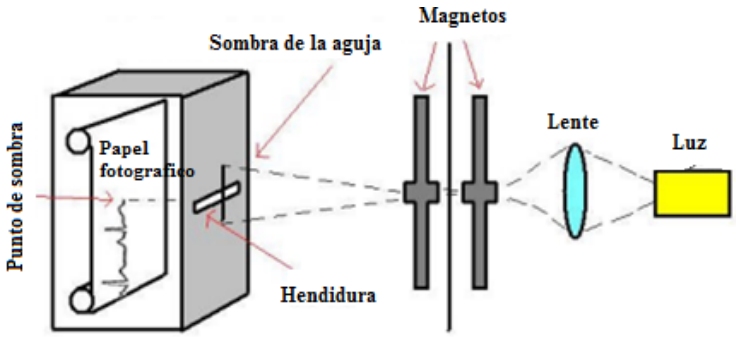


Figura 1.5 Modelo del primer electrocardiógrafo [4].
Fuente: Revista Médica de Chile, Historia de la medicina. Einthoven. El hombre y su invento.

La aguja del modelo de Einthoven se iluminaba por un haz de luz, la cual generaba una sombra que genera un barrido de un lado a otro dependiendo de la corriente que pasa por el hilo metálico y dicha sombra se expone por una ranura delgada produciendo un punto sobre un papel fotográfico registrando así la señal electrocardiográfica.

Hasta la modernización del electrocardiógrafo con la electrónica digital los electrocardiógrafos eran básicamente un galvanómetro que permitía registrar la actividad eléctrica a partir de electrodos conectados en la superficie del paciente, y esta señal amplificada y enviada a un oscilógrafo que al modificar la posición de un elemento de registro gráfico se movía sobre papel milimetrado. Cada diferencia de potencial es representada con movimientos de la aguja hacia arriba o abajo según la polaridad registrada y la magnitud del potencial, obteniendo así un trazo con ondas que refleja la actividad cardíaca.

1.1.3 Primer modelo ECG Willem Einthoven

Einthoven empezó a desarrollar su galvanómetro en el año 1900 luego de años de experimentar con el electrómetro capilar con resultados no satisfactorios, este nuevo aparato se llamó galvanómetro de cuerda. (Figura 1.6). Con este nuevo instrumento Einthoven asignó las letras P, Q, R, S y T a las diferentes deflexiones y describió las características electrocardiográficas de gran número de enfermedades cardiovasculares. En 1903, Einthoven inició la producción comercial del galvanómetro de cuerdas junto con la compañía inglesa *Cambridge Scientific Instruments* [5].

Figura 1.6 Primer electrocardiógrafo comercial [5].
Fuente: Acierno, L.J. The History of Cardiology.

Los dispositivos fueron reduciendo su tamaño, a pesar de conservar el principio de operación (ver Figura 1.7) se cambiaron los electrodos por placas de plata para situarlos directamente en la piel del paciente. El electrodo de succión lo diseño Rudolph Berger en 1932, para sujetar dicho electrodo al pecho [6].

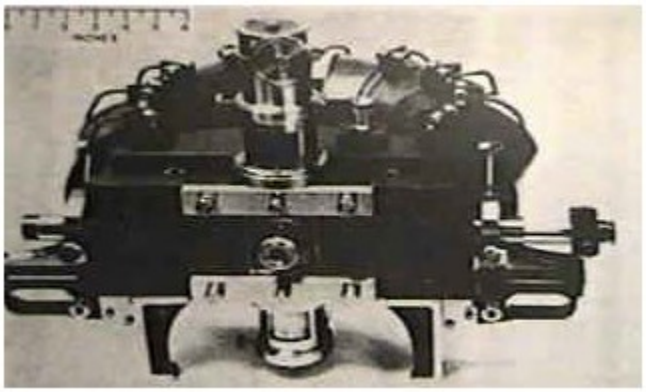


Figura 1.7 Galvanómetro de cuerdas creado por Willem Einthoven [6].
Fuente: Electrochemical and Electronics instruments version online.

Entre los trabajos importantes de Einthoven su artículo más famoso artículo *"Le telecardiogramme*" describe con detalle las aplicaciones clínicas del electrocardiograma. Describió además varios desordenes cardiovasculares como hipertrofia ventricular y auricular, la onda U descrita por primera vez, el complejo QRS, bloqueos auricular y ventricular. En esta publicación se establecieron las bases para futuros informes importantes sobre electrocardiogramas. Diseño además el papel de registro y estableció las derivaciones I, II, III, que constituyen el triángulo de Einthoven.

Le fue otorgado el Premio Nobel de Fisiología o Medicina en 1924 por su descubrimiento para la década de 1970 los sistemas de adquisición de 12 derivaciones ya eran una realidad y sistemas para el análisis automático de electrocardiografía que incorporaban microprocesadores se comenzaron a implementar [7].

1.1.4 Principio de funcionamiento ECG moderno.

Los electrocardiógrafos actuales reciben la señal proveniente de los electrodos, y dicha señal es amplificada por un amplificador de instrumentación el cual amplifica únicamente la diferencia de potenciales entre los arreglos de electrodos (derivación) esta etapa se conoce como etapa de pre-amplificación. Debido a que la intensidad detectable del corazón radica entre los .05Hz y 250Hz es necesario aplicar un filtro pasa banda, y utilizar un filtro Notch de 60Hz para eliminar la componente del suministro eléctrico. Luego de obtener la señal en el ancho de banda de interés y sin ruido, nuestra señal esta lista para ser amplificada hasta un nivel lógico ideal para la conversión analógico-digital mediante un microcontrolador, y posteriormente su visualización, registro, impresión, o trasmisión del electrocardiograma (ver Figura 1.8).

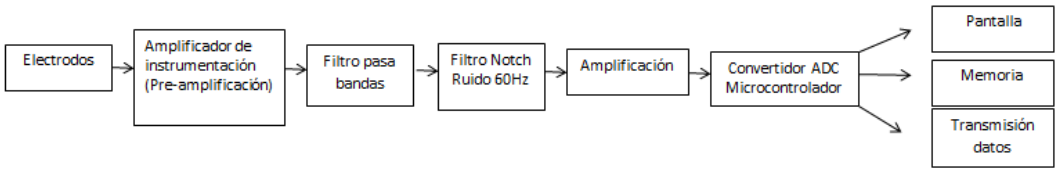


Figura 1.8 Diagrama a bloques de un electrocardiógrafo moderno.

1.1.5 Electrocardiógrafos modernos.

En la actualidad los electrocardiógrafos se pueden clasificar según su funcionamiento para recepción e impresión de datos o su aplicación clínica o no clínica. Según su funcionamiento se clasifican en monocanales, y multicanales. Los electrocardiógrafos monocanales registran e imprimen los reportes de actividad de una sola derivación (la actividad entre 2 electrodos) por registro. Entre las ventajas de estos equipos es su tamaño compacto, peso, y simplicidad de uso. En los electrocardiógrafos multicanal el usuario puede seleccionar el modo automático o manual y elegir las derivaciones, sensibilidad, rango de frecuencia de muestreo y velocidad de papel. La señal proveniente de 12 derivaciones (precordiales, unipolares, y bipolares) son registradas durante un intervalo de 12 segundos simultáneamente e imprimen 3 o 6 al mismo tiempo. Algunos modelos permiten observar la señal en tiempo real antes de realizar la impresión. Son los más completos para las distintas aplicaciones hospitalarias, (ver Figura 1.9) pero a diferencia de monocanal su interfaz de uso es más compleja, así como dimensiones y peso.

Según su aplicación se pueden clasificar en ambulatorios, de uso clínico, o monitoreo (Holter). Entre sus principales ventajas y desventajas los de uso ambulatorio cumplen la función de ser portátiles monocanal debido a su diseño y simplicidad. Los de uso clínico por lo general son multicanal, son los más precisos y claros con la calidad para proporcionar un diagnóstico adecuado, deben portar con baterías y conexión a suministro eléctrico por lo que presenta desventajas claras en aspectos de portabilidad. Esto se compensa con su confiabilidad y calidad en la labor de diagnóstico haciéndolo útil en clínicas y hospitales. Finalmente los de monitoreo son electrocardiógrafos de tipo Holter los cuales cumplen con la función de portabilidad deben monitorear al usuario durante tiempos prolongados, utilizan baterías, con dimensiones y peso limitados para la comodidad del usuario. Entre sus desventajas es presentan un diagnóstico no tan completo como los ambulatorios o clínicos, pero sirven para proporcionar un cuidado esencial en pacientes que requieran monitoreo constante.

Entre las mejoras adicionales se encuentra la elaboración de módulos multifunción que además de ofrecer un Electrocardiógrafo, cumplen con la función de frecuencia respiratoria, pulsioxímetro, termómetro entre otras variables fisiológicas. Además de contar con transmisión de datos alámbrica o inalámbricamente 3G, Wifi, Bluetooth para desplegar la información no solo en el papel registro o pantalla si no en dispositivos portátiles como son teléfonos inteligentes, tablets o computadoras [8].

Figura 1.9 Electrocardiógrafo moderno [9].
Fuente: http://Equimeed.com

1.1.6 Primer modelo ECG portátil y trasmisión de datos (Holter).

Por otro lado un aspecto importante de este proyecto es la portabilidad del electrocardiógrafo Norman Holter en 1949 desarrolló una especie de mochila de unos 37 Kg con la que se puede registrar el electrocardiograma de quien la porta y transmitirlo por ondas de radio, lográndose el primer registro ambulatorio de electrocardiograma (ver Figura 1.10). Años después fue mejorado y comercializado por la compañía Frank Sanborn con un peso de 25Kg. Y funcionaba con una batería de automóvil de 6V. [10].

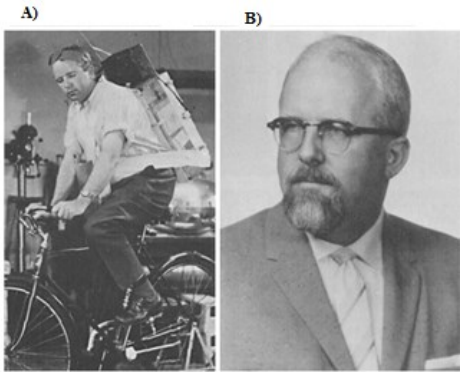


Figura 1.10 A) Primer electrocardiógrafo portátil inventado por B) Norman Holter [11].
Fuente: http://www.bcmj.org/sites/default/files/BCMJ_56_Vol2_holter.pdf

Con el avance de los transistores, el equipo fue progresivamente miniaturizándose, reduciendo el consumo, y la radiotransmisión fue reemplazada por un grabador de cinta electromagnético. A partir de 1952, el engorroso y pesado sistema Holter, fue miniaturizado y pasó de 37kg de peso a tan sólo 1,2kg. En la actualidad, los registradores Holter son dispositivos mucho más compactos, ligeros, ya que no graban en cinta magnética, sino de manera digital en memoria y en modo de transmisión inalámbrica puede variar por bluetooth, wifi, entre otras vías aun en desarrollo (ver Figura 1.11) [10].

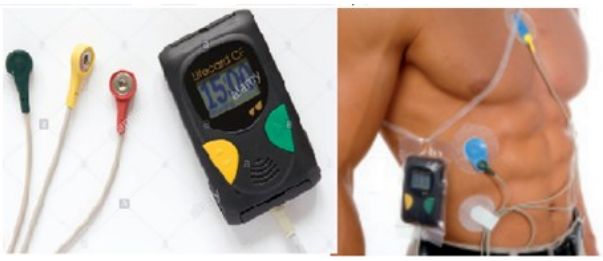


Figura 1.11 Electrocardiógrafo Holter moderno modelo Lifecard CF [12].
Fuente: Spacelabs healtcare manual Holter lifecard-CF.

1.2 Uso del microchip AD8232 para diseño y fabricación de un ECG portátil.

En los últimos años la tendencia del diseño y fabricación de equipo médico portátil usando nuevas tecnologías ha llamado la atención de un gran número de investigadores y científicos, por lo que se tienen algunas referencias (ver Tabla 1.1 y 1.2) de la implementación de estos sistemas, en la cual en algunos casos no se ha completado la fase de visualización, interfaz de usuario, o aplicación de filtros para la calidad de la señal dejando un gran campo de oportunidad y mejora.[13-20] Uno de los aspectos importantes en la realización de este proyecto es la evaluación y utilización del microchip AD8232 ofrecido por la compañía Analog Devices.

En [13] se realizaron pruebas con el chip AD8232, pero menciona que al cometerse un error en el diseño e implementación la placa fue descartada optando por la placa Olimex ECG *Shield*. En este proyecto se propuso trabajar inicialmente con la placa integrada Eval-AD8232 de la compañía Analog Devices la cual sirve para la evaluación y pruebas de este microchip. También en [13], se utilizó Arduino Uno como convertidor A/D por el libre acceso a sus librerías, dando buenos resultados la utilización de su convertidor analógico digital para la digitalización de la señal. Para esta investigación se utilizó una placa similar de esta familia llamado Arduino Nano la cual se basa en el mismo microcontrolador, pero en dimensiones y consumo energético reducido con el fin de aplicaciones donde sea importante la portabilidad.

Para el filtrado de la señal [13] realizó varias pruebas comenzando por simulación de filtros RC los cuales documentó no obtuvieron resultados fructuosos, seguidos de aplicar transformada de Fourier para filtrar la señal sin mejoría alguna, finalmente optó por un filtro de media móvil que toma las 19 muestras anteriores a una frecuencia de muestreo de 1kHz, obteniendo resultados excepcionales ya que según documenta se eliminaron las componentes ruidosas, eliminando así la más molesta que es el ruido eléctrico. Para fines de esta investigación se simularon diferentes filtros analógicos, y digitales mediante ecuaciones de diferencia para para eliminación de ruido, y la obtención de la señal deseada.

En [13] se incorporó también un sistema para la detección de anomalías y diagnóstico de ECG adaptado a la plataforma Android, este software de análisis tiene el nombre de CLIPS

y consiste en un sistema de inferencia que analiza los datos obtenidos para brindar un diagnóstico pero no se obtuvieron muy buenos resultados. En lo personal pienso que hasta donde la tecnología lo permite, ninguno de estos sistemas puede superar años de preparación y experiencia por parte de un personal médico de cardiología. Aun así con el avance de la tecnología y el desarrollo de este tipo de software en desarrollo (*open source*) para diagnóstico pueden ser útiles para tener presente el estado de la salud cotidiano y así promover la cultura para el monitoreo constante y personal de la salud.

Para fines de esta investigación debido a que la aplicación que se busca no es el diagnóstico, si no el monitoreo de la actividad del corazón no se utilizó un software o sistema de diagnóstico, aunque se utilizó una interfaz de visualización de libre acceso para que el usuario decida utilizar el software de su preferencia (por ejemplo Excel, THEREMINO).

Otra investigación donde se aplicó el uso del chip AD8232 con fines de monitoreo para el cuidado de la salud, es [14] cual incorporó filtros RC con frecuencias de corte de 0.34Hz a 150Hz para filtrar el ruido de la señal. Y para trasmisión de datos vía bluetooth utilizó el módulo nRF8001 el cual utiliza protocolo de comunicación SPI, además de tener compatibilidad con la aplicación “salud” de IOS Apple.

Otros artículos de trabajos similares [15,16] utilizan como microcontrolador y convertidor el módulo Arduino uno, con el microcontrolador ATmega128, trabajan con un ancho de banda de 0.3 a 100Hz, con una frecuencia de muestreo de 200Hz, y [16] utilizó conexión serial alámbrica además de bluetooth para visualizar la señal.

Existen otros módulos para la conversión y transmisión de una señal, [17] utilizó el módulo STM32L151 el cual incorpora un convertidor, microcontrolador y bluetooth integrado, pero limitó la transmisión a solo este protocolo de comunicación. Con este módulo obtuvo resultados satisfactorios, e implementó el diseño de una camisa con electrodos para la facilidad de uso.

1.2.1 Tabla resumen de trabajos previos utilizando chip AD8232 de Analog Devices.

Tabla 1.1 Trabajos de tesis con aplicación del chip AD8232.

Nombre de la tesis [Ref.]	Tipo de microcontrolador y uso	Autor	Institución	Frecuencia de operación Ancho de banda	Tipo de comunicación de datos. (No. De bytes)	Análisis o procesamiento de datos	Presentación de datos (bluetooth, LCD, Memoria) Tiempo real.	Comunicación inalámbrica
Analog Integrated Circuit Applications [18].	nRF51-DK nRF51422 (módulo transmisor Bluetooth con ADC y microcontrolador incluido).	Long Nguyen, et.al.	Worcester Polytechnic Institute 2013.	0.2-200Hz.	Serial 10 bits.	Señal medida Osciloscopio, App Android (trabajo futuro).	Señal digital medición Osciloscopio.	Nordic Semiconductors nRF51422.
Extending Sensing Capabilities and Modalities of Mobile Devices [19].	Atmel Arduino Mega2560-based USB Bridge.	Rohit Chaudhri.	University o f Was hington 2014.	N/A	Comunicación serial procesador (ATMega328p a 8MHz).	Según su uso, se puede transmitir la información adquirida en app Android, o guardar los registros de temperatura de las vacunas o leche para ver la gráfica de tiempo en la PC.	Los datos se presentan según la aplicación, se pueden guardar datos vía USB, transmitir vía bluetooth.	Módulo Bluetooth RN42.
Un electrocardiógrafo inteligente de bajo coste [13].	Arduino Uno R3 Chip ATmega328 de Atmel.	A. Mendiguren.	Universidad del país de Vasco 2014.	0.5 a 100Hz.	Comunicación serial chip ATmega328, 10bites 1024 bytes.	Realización de diagnóstico a partir de la información recibida mediante aplicación Android.	Visualización tiempo real, por transmisión de datos vía Bluetooth en Tablet.	Módulo bluetooth. JY-MCU, basado en el chip HC-06.
Wireless Hybrid Bio-Sensing with Mobile based Monitoring System [14].	ATmega328P.	Linlin Xu.	School of Information and Communicat ion Technology Kungliga Tekniska Högskolan 2013.	0.34-150Hz.	Comunicación serial ATmega328P ADC has a 10-bit.	El análisis de datos se realiza mediante la App del iPod, se mide la señal obtenida del ADC, y una vez transmitida los datos en el computadora se grafica para corroborar.	Señal visualizada en osciloscopio, iPod touch con App, y graficada a partir del vector de datos.	Bluetooth bajo consumo. Conexión IC nRF8001.

Tabla 1.2 Artículos relacionados con el chip AD8232.

Nombre de la tesis	Tipo de microcontrolador y uso	Autor	Institución	Frecuencia de operación Ancho de banda	Tipo de comunicación de datos. (No. De bytes)	Análisis o procesamiento de datos	Presentación de datos (bluetooth, LCD, Memoria) Tiempo real.	Comunicación inalámbrica
A Health Shirt with ECG Real-time Display on Android Platform [17].	Module is STM32L151. It uses an ultra-low-power Cortex-M3 core.	Zixiao Shen et. al.	Shenzhen Institutes of Advanced Technology, Chinese Academy of Science 2014.	0.5-150Hz.	Serial 12-bit ADC.	Análisis de datos de matrices de datos por MATLAB.	Visualización en app plataforma Android.	Módulo bluetooth CC2540.
A Portable ECG Monitor with Low Power Consumption and Small Size Based on AD8232 Chip [15].	ATmega328p chip.	Tancheng Lu et. al.	Department of Electronic Engineering, Fudan University *2014.*	0.5-100 Hz.	Comunicación serial chip ATmega328, 10bites 1024 bytes.	Se graba mediante APP en móvil un minuto de datos ECG cuando se detecta arritmias.	Visualización aplicación celular.	Módulo bluetooth TX-RX No especifica.
Design of ECG Homecare:12-Lead ECG Acquisition using Single Channel ECG Device Developed on AD8232 Analog Front End [20].	10-bit ATmega en Arduino Uno R3.	Muhammad Wildan Gigari et. al.	KK Teknik Biomedika Sekolah Teknik Elektro dan Informatika ITB 2015.	0.05-100 Hz.	Comunicación serial arduino uno 10-bit.	Análisis de datos mediante app Android, y de manera alámbrica al PC según los resultados la conexión alámbrica es más clara.	Bluetooth Display grafico en tiempo real con Android.	Módulo bluetooth HC-05.
Medical Monitoring: Empowered By Integrated Analog Technology [16].	Módulo BLEN micro contiene Atmel ATmega32U4 Bluetooth Nordic nRF8001.	Vijay Laxmi Kalyani, et. al.	*Faculty of Electrical Engineering Brno University of Technology* 2014.	N/A	Comunicación SPI Serial Interface arduino uno 10-bit ADC.	N/A	Visualización aplicación celular y datos en PC vía serial.	Módulo bluetooth.

1.3 Propuesta de proyecto

Para fines de este proyecto se propuso un electrocardiógrafo portátil de uso personal de una derivación, que permita el registro y transmisión de datos, durante tiempo prolongado mediante la implementación del chip AD8232, y el microcontrolador ATmega328 siendo este el utilizado en la plataforma de código abierto Arduino (ver Figura 1.12 y 1.13). Se busca además que el sistema sea de bajo costo, se puede ver en el presupuesto de este proyecto en la tabla 4.8 en la sección 4.11 que el costo fue de 20 USD.

Se utilizó un ancho de banda de 0.5Hz-40Hz el cual es el estándar para la aplicación de monitoreo y sistemas ambulatorios holter de acuerdo con la norma de diseño para sistemas de electrocardiografía ambulatoria, IEC 60601-2-47:2012 atendiendo las recomendaciones de la AHA (*American Heart Association*) [21]. Los requerimientos para la frecuencia de muestreo debe cumplir el teorema de *Nyquist* siendo más del doble o triple de la frecuencia de corte, la normativa indica que para monitoreo basta con una frecuencia de muestreo 120Hz, y para cubrir un uso clínico 500Hz. El convertidor analógico digital debe ser mayor a 8 bits para la aplicación de monitoreo, siendo el del microcontrolador ATmega328 de 10 bits.

Para la sección de adquisición de la señal se utilizaron electrodos desechables de parche, debido a que son los más utilizados para la obtención del ECG mediante triángulo de Einthoven y cumplen con la normativa ANSI/AAMI EC12:2000/(R) 2010 para electrodos en sistemas de electrocardiografía [22]. Se utilizó también el tablero de evaluación EVAL-AD8232 inicialmente y luego de desarrollar un circuito propio que se ajuste a la normativa, con requerimientos acreditados por el chip AD8232 como un CMRR $\geq$80dB, impedancia de entrada $\geq$ 5MΩ, ruido de la señal $\leq$ 30μV, y voltaje offset DC de $\pm$300mV. Para la digitalización de la señal se utilizó el módulo Arduino Nano (con microcontrolador ATmega328) debido a que es una plataforma de código abierto y se ha reportado buenos resultados en la mayoría de los trabajos. Se utilizó el módulo data logger que incorpora una ranura SD con conexión SPI y un reloj de tiempo real 1307 incluido que funcionan en conjunto con una batería CR1220 para el almacenamiento de la información una vez apagado o desenergizado dicho módulo. Para transmisión de datos el módulo bluetooth HC-06 es una excelente opción ya que se ha utilizado en un gran número de trabajos y ha mostrado buenos resultados y estabilidad en su funcionamiento. Cada elemento se validó y caracterizó para la aplicación de desarrollo de un sistema ECG para monitoreo de uso personal portátil.

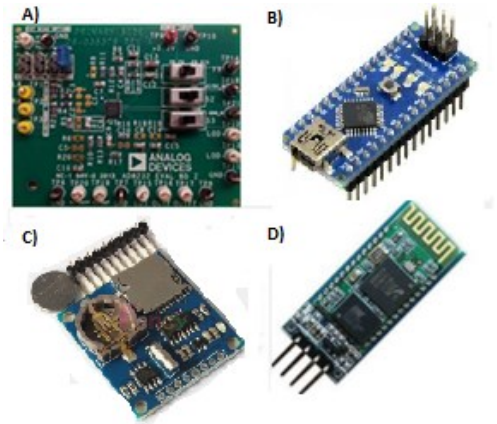


Figura 1.12 A) Tablero de evaluación AD8232 de Analog Devices, B) Microcontrolador basado en plataforma de código Abierto (Arduino Nano). C) Data logger shield, y D) Módulo Bluetooth HC-06.

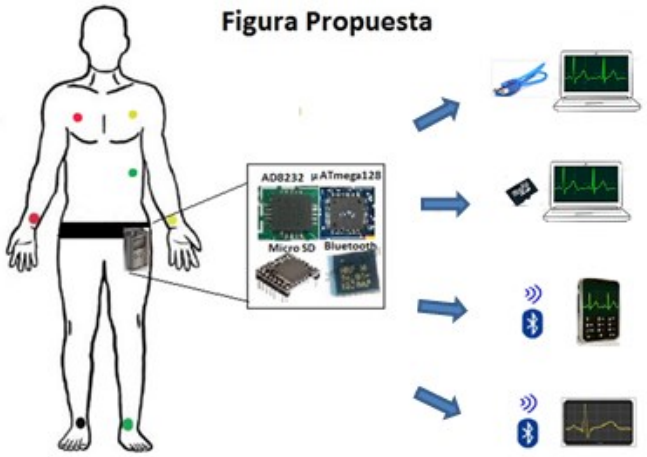


Figura 1.13 Propuesta del prototipo ECG portátil para monitoreo personal y tiempo prolongado con chip AD8232 y plataforma de código abierto.

CAPÍTULO 2

MARCO TEÓRICO.

2.1 Potencial de acción del corazón.

El electrocardiograma representa la actividad eléctrica del corazón, que se capta al establecer la interfaz electrodo-electrolito cuando el electrodo hace contacto con la piel. Estas diferencias de potenciales se generan por intercambio iónico entre células y se da a partir de la membrana celular la cual modifica su permeabilidad a iones Na^+, y K^+ (ver Figura 2.1).

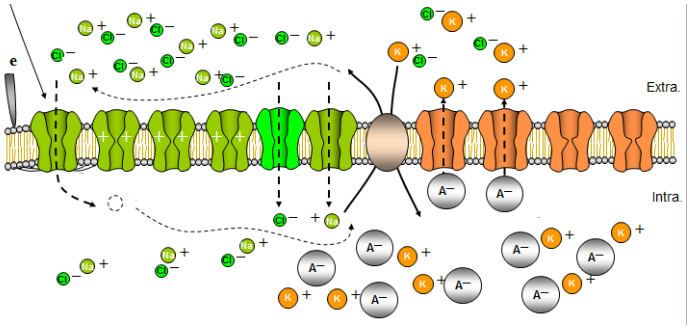


Figura 2.1 Membrana celular e intercambio iónico para generar un potencial de acción donde los iones A- son aniones, K^+ es potasio, Na^+ es sodio, y Cl^- es cloro [23].
Fuente: Netter's Essential Physiology Susan E. Mulroney, PhD & Adam K. Myers, PhD

El potencial de membrana es el que presenta la membrana celular en modo de reposo, para células excitables y en células no excitables se conoce como potencial de membrana. En células excitables puede presentarse un estímulo subumbral en el cual se abren muy pocos canales de Na^+ y solo se presenta una despolarización parcial por lo que no se logra generar un potencial de acción. Cuando se presenta ese estímulo umbral, se despolariza la membrana celular y se provoca la apertura de canales para el Na^+ dependiente de voltaje lo que a su vez, abre más canales permitiendo la entrada rápida de Na^+ a la célula. Y permitiendo que se genere el potencial de acción (ver Figura 2.2). Una vez que se alcanza el potencial de estabilidad del Na^+ se cierran los canales para el Na^+ y se abren los canales dependientes de voltaje para el K^+ permitiendo la salida rápida de este y provocando una hiperpolarización, luego que se llegó al potencial de estabilidad de K^+ se cierran los canales para el K^+, activándose la bomba sodio potasio para establecer el equilibrio iónico de la célula en estado de reposo y así recuperar su polaridad negativa, por cada 3 iones de Na^+ que saca mete 2 iones de K^+ y vuelve al potencial de reposo.

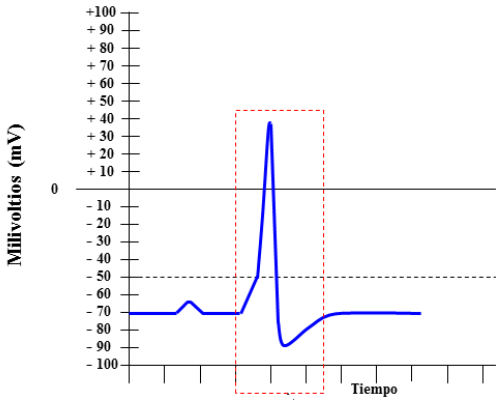


Figura 2.2 Potencial de acción generado por el intercambio iónico de una célula excitable. [23]
Fuente: Netter's Essential Physiology Susan E. Mulroney, PhD & Adam K. Myers, PhD

2.2 Electrocardiograma (ECG)

Durante los latidos del corazón se generan diferencias de potencial que se propagan y transmiten a través del medio extracelular a través de células, tejidos y órganos hasta la piel, y estos biopotenciales o diferencias de potencial pueden ser utilizadas como herramienta diagnóstica para revisar el funcionamiento del corazón. A esta técnica que mide la diferencia de potencial entre ciertas partes del cuerpo (según la técnica) generados por la actividad del corazón se le conoce como electrocardiografía.

Esta actividad puede ser representada por un vector y por lo tanto es importante conocer su amplitud, comportamiento en el tiempo y ubicación. El modelo del corazón para fines de análisis puede considerarse como un dipolo eléctrico, sabemos que potencial de acción puede tener polaridad positiva o negativa, y así es como se puede medir a través de las manos por ejemplo (derivaciones de Einthoven) tomando en cuenta su polaridad y magnitud para dar origen a la onda de ECG (ver Figura 2.3).

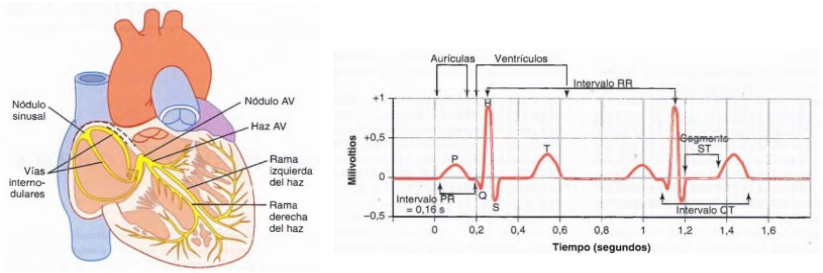


Figura 2.3 Anatomía del corazón, y onda ECG normal [24].
Fuente: Guyton y Hall, Tratado de fisiología médica, 12va. Edición. 1112 págs. Año 2011

La frecuencia cardíaca normal de una persona se encuentra alrededor de 70 latidos por minuto (BPM) y puede variar entre 30 y 200 BPM. La amplitud de las señales de electrocardiografía puede variar entre 0.1 mV y 5 mV y depende de la distancia que se encuentren los electrodos del corazón, entre más distanciados menor será la amplitud de la señal. La mayor parte de la energía de la señal de electrocardiografía se encuentra en un ancho de banda entre 0.05 y 250 Hz [25].

Como se mencionó la actividad del corazón se puede representar mediante un vector el cual representa esa diferencia de potencial que corresponde a cada una de las etapas del ciclo cardíaco como son: despolarización auricular en las que las aurículas envían sangre a los ventrículos al contraerse, la despolarización ventricular donde se lleva a cabo la expansión de los ventrículos para recibir la sangre de las aurículas y posteriormente su contracción para la eyección de la sangre al organismo, y una vez eyectada la sangre, la repolarización ventricular para la relajación del músculo cardíaco y comenzar así un ciclo vital para la vida.

La onda P: Nos indica que se está realizando la contracción auricular. La duración de esta onda suele ser inferior a los 100ms y su amplitud está entre 0,1mV y 0,5mV. Las ondas P irregulares o inexistentes pueden provocar una arritmia.

El complejo QRS: Representa la despolarización ventricular antes de su contracción. Es la onda de mayor Amplitud y su duración puede variar entre 60ms y 100ms. La onda P y complejo QRS son llamadas ondas de despolarización.

La onda T es producida por los potenciales que se generan cuando los ventrículos se recuperan del estado de despolarización. Éste proceso aparece normalmente en el músculo ventricular entre 0.25 y 0.35 segundos después de la despolarización, es una onda positiva de una amplitud que no supera los 0.6mV. Se conoce como onda de repolarización [25].

2.2.1 Principio de análisis de vectores cardíacos para registro del electrocardiograma.

El ciclo cardíaco inicia con la despolarización auricular la cual se genera en el nodo seno auricular (ver figura 2.4) donde se genera el potencial de acción para iniciar la despolarización auricular. Inicia el flujo de corriente en el nodo sinusal en el tiempo 0 llega al nodo aurículo ventricular en el tiempo 0.02-0.04 segundos, y finalmente despolariza completamente ambas aurículas a los 0.09 segundos. El vector resultante de la

despolarización auricular se puede ver en la figura 2.4, y analizando el vector con las derivaciones del triángulo de Einthoven podemos observar cómo se genera la onda P la cual corresponde a la despolarización auricular [25].

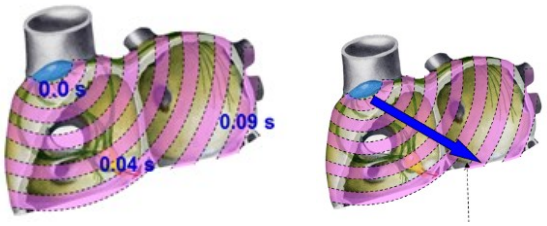


Figura 2.4 Despolarización auricular y vector generado por acción de la misma [23].
Fuente: Netter's Essential Physiology Susan E. Mulroney, PhD & Adam K. Myers, PhD

La proyección del vector cardíaco de despolarización auricular generan el registro correspondiente a la imagen I, donde sí se expusiera un haz de luz desde la punta contraria a la derivación I la proyección seria el vector superior al triángulo, desde negativo hasta positivo así pues el registro de la onda observado en la derivación I aplicando el mismo principio a la derivación II y derivación III las señales se pueden observar en la Figura 2.5, este análisis se conoce como triángulo de Einthoven.

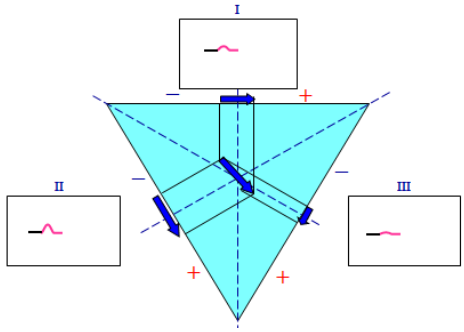


Figura 2.5 Vector cardíaco de despolarización y análisis mediante triángulo de Einthoven y la onda P resultante de la despolarización auricular [23].
Fuente: Netter's Essential Physiology Susan E. Mulroney, PhD & Adam K. Myers, PhD

Siguiendo con el ciclo cardíaco el evento de despolarización ventricular se inicia su registro al llegar la despolarización del nodo aurículo ventricular pero es tan rápido que no genera un registro detectable hasta que llega al ventrículo por las fibras de Purkinje. Y obtenemos un vector llamado septal. La despolarización sigue propagándose desde el endocardio hacia epicardio obteniendo el vector apical, y finalmente se despolarizan ambos

ventrículos obteniendo el vector basal (estos 3 vectores de la Figura 2.6 corresponden al complejo QRS) [23].

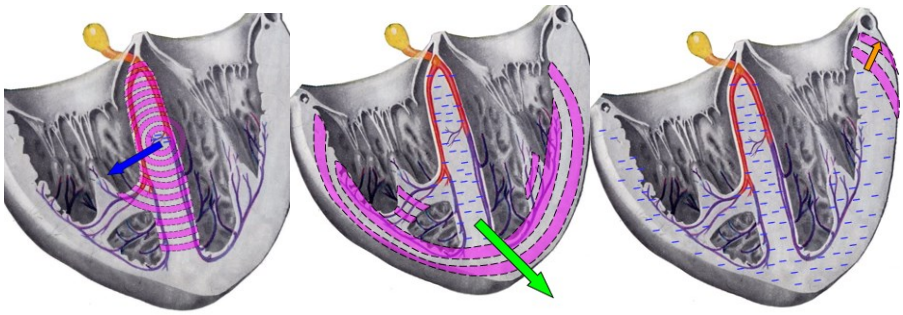


Figura 2.6 Despolarización ventricular y vectores generados por acción de la misma.
a) Vector septal, b) Vector apical, c) Vector basal [23].
Fuente: Netter's Essential Physiology Susan E. Mulroney, PhD & Adam K. Myers, PhD

Aplicando el mismo análisis de proyección de los vectores por el triángulo de Einthoven obtenemos las ondas Q, R, S que representan la despolarización ventricular (Figura 2.7).

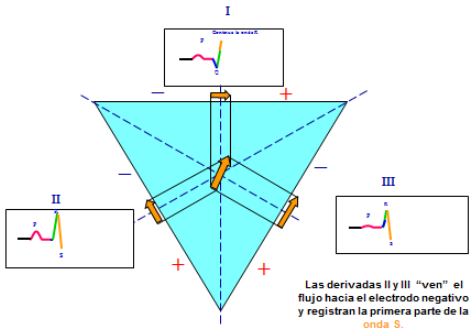


Figura 2.7 Proyecciones de vectores de despolarización ventricular y análisis mediante derivaciones de Einthoven para formar el complejo QRS [23].
Fuente: Netter's Essential Physiology Susan E. Mulroney, PhD & Adam K. Myers, PhD

Una vez despolarizados ambos ventrículos y alcanzada la eyección de la sangre hacia el cuerpo, el corazón comienza su repolarización hasta llegar al nodo aurículo ventricular (ver figura 2.8).

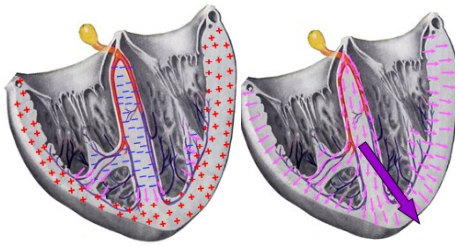


Figura 2.8 Repolarización ventricular y vector generado por acción de la misma [23].
Fuente: Netter's Essential Physiology Susan E. Mulroney, PhD & Adam K. Myers, PhD

Luego de despolarizarse los ventrículos y alcanzada la eyección de la sangre hacia el cuerpo, el corazón comienza su repolarización generando el vector de repolarización y originando la onda T. Obteniendo así un registro exitoso de ECG en condiciones normales (ver Figura 2.9).

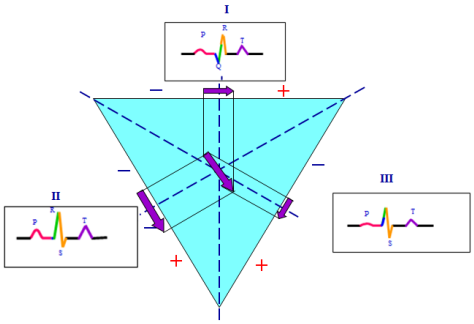


Figura 2.9 Análisis de vector de repolarización ventricular mediante derivaciones de Einthoven formando la onda T [23].
Fuente: Netter's Essential Physiology Susan E. Mulroney, PhD & Adam K. Myers, PhD

2.2.2 Derivaciones del electrocardiograma.

La forma de medir los potenciales de acción, por la actividad del ciclo cardíaco es midiendo la diferencia de potencial entre dos electrodos en la superficie del cuerpo. En estos arreglos se generan diferentes voltajes por la dependencia espacial del campo eléctrico del corazón y cada arreglo de electrodos en posiciones específicas se conoce como derivación. Existen tres derivaciones básicas conocidas como I, II y III, que expresadas como vectores forma un triángulo en el plano frontal de cuerpo, denominado el triángulo de Einthoven (ver Figura 2.10). La derivación I, se mide posicionando electrodos en la mano derecha (RA) y en la mano izquierda (LA), la derivación II se obtiene con electrodos entre el pie izquierdo (LL) y la mano derecha y la derivación III se obtiene posicionado electrodos entre el pie izquierdo y la mano izquierda. También existen otras derivaciones denominadas derivaciones precordiales (V1, V2, V3, V4, V5 y V6) son derivaciones empleadas para precisar con exactitud perturbaciones miocárdicas del lado izquierdo y derecho del corazón, para distinguir las lesiones de la pared anterior y de la pared posterior. Estas derivaciones permiten el registro de potenciales que no son posibles registrar en las derivaciones de Einthoven ni las aumentadas derivaciones aumentadas (AVL, AVR y AVF).

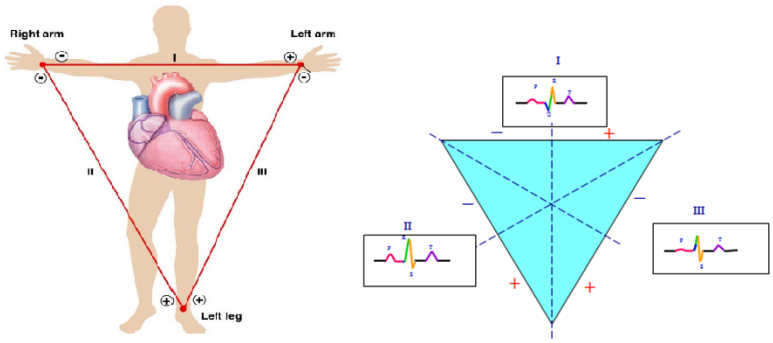
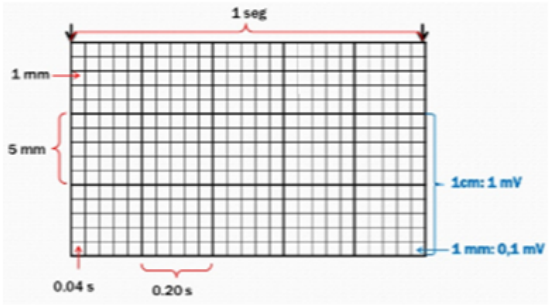


Figura 2.10 Triángulo de Einthoven, y diagrama de análisis vectorial [23].
Fuente: Netter's Essential Physiology Susan E. Mulroney, PhD & Adam K. Myers, PhD

Uno de los aspectos más importantes del ECG de 3 derivaciones (sobre todo en DII) son el ritmo cardíaco, puede determinarse la posición del corazón, las medidas como amplitud y duración de las ondas y segmentos que sirven para determinar patologías, diagnóstico diferencial de las arritmias y la frecuencia cardíaca. La importancia de la derivación 2 es que siguiendo una relación matemática enunciada por el mismo Einthoven se tiene que DII=DI+DIII por lo que sí es posible detectar alguna patología en cualquiera de las 3 derivaciones podrá ser apreciable de forma más significativa en la derivación II por lo que es la más utilizada para diagnóstico e interpretación del ECG [26].

2.3 Electrocardiograma: Interpretación y reporte

A pesar del amplio estudio que se tiene en el campo de la cardiología, un electrocardiógrafo por si solo aun no es capaz de interpretar las patologías que pueden representarse en el electrocardiograma por lo que es necesaria la interpretación de un médico cardiólogo que tome en cuenta los parámetros del papel registro (ver Figura 2.11).



Papel de registro:
Papel milimetrado cuadriculado: cada 5 mm línea más gruesa.
Tiempo en eje de abscisas: edad de 25 mm/s (la habitual):
1 mm= 0,04 seg.
5 mm=0,2 seg.
Voltaje en ordenadas :
1 mV=10 mm
0,1 mV= 1 mm.

Figura 2.11 Papel registro y parámetros a considerar durante el diagnóstico [27].
Fuente: Medicine Plus Enciclopedia médica "Electrocardiograma".

Dentro de la utilidad del electrocardiograma esta la detección de patologías o irregularidades como son: daños en el miocardio, la frecuencia cardíaca (taquicardia o

bradicardia), bloqueos de cavidades o conductos del corazón, tamaño y posición de las cámaras del corazón. Ataque cardíaco anterior en evolución o inminente, alteraciones en la cantidad de electrolitos en la sangre [27].

2.4 Diagrama a bloques de un ECG.

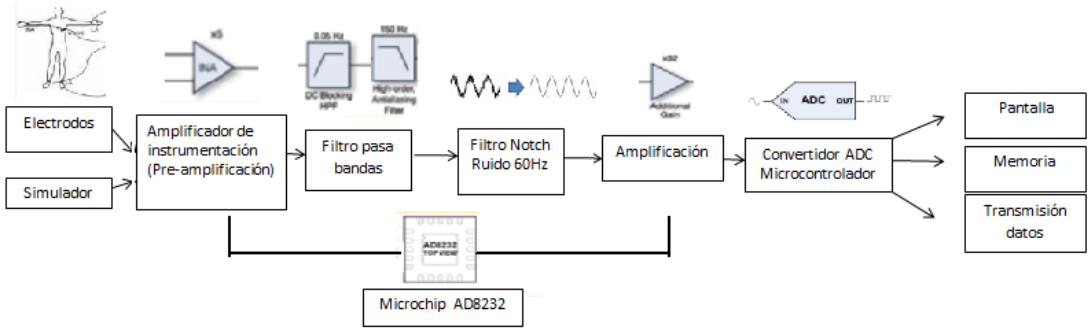


Figura 2.12 Esquema a bloques de sistema de adquisición de datos de un ECG [28].

2.4.1 Electrodos

La primera etapa del diagrama a bloques de un ECG (ver Figura 2.12) son los electrodos, utilizados para transmitir los biopotenciales generados en la interacción entre el cuerpo y el equipo de medición. Al entrar en contacto la piel y el elemento metálico del electrodo se por ellos circulará una corriente producto del intercambio iónico e interacción de electrones entre la solución y iones del electrolito se combinaran con el elemento metálico del electrodo, los cuales darán como resultado una reacción de óxido-reducción (Ver figura 2.13) y como resultado generará un potencial llamado de media celda [29].

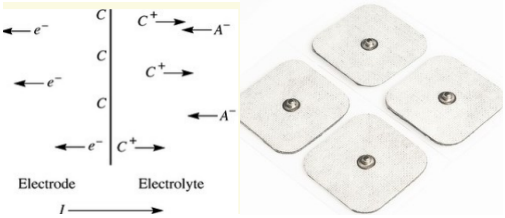


Figura 2.13 Interfaz electrodo electrolito. Y electrodo Ag/AgCl [29,30].
Fuente: *Introducción a la Bioingeniería capítulo 4 Transductores bioeléctricas*, Barcelona, España, Marcombo.
Grupomedicastore/ilustraciones/tipos de electrodos para ECG

2.4.2 Amplificador de instrumentación de alto CMRR

El amplificador de instrumentación se rige por la ecuación 1 y 2 Siendo la función de transferencia la ganancia. Las entradas al circuito es V1 y V2; estas son las conexiones de los electrodos. Su ganancia se puede calcular por la Ec. 2 con el arreglo de resistencias y amplificadores que se puede ver en la Figura 2.14.

$$Vout = (V2 - V1) * (1 - \frac{2 * R_1}{R_g}) \quad . \tag{1}$$

Ec.1 Se muestra Ecuación de Vout de un amplificador de instrumentación [31].

La entrada común será el ruido blanco proveniente del exterior o del ambiente que rodea al sistema es por eso el factor CMRR del amplificador instrumental tiene que ser lo más alto posible.

$$G = (1 - \frac{2 * R_1}{R_g}) \tag{2}$$

Ec.2 Se muestra Ecuación de Ganancia del amplificador de instrumentación [31].

El factor de rechazo en modo común CMRR debe ser alto, lo cual nos indica que hay un buen rechazo al ruido [31].

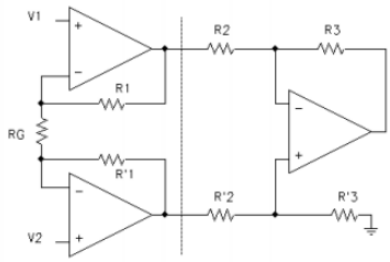


Figura 2.14 Diagrama de un amplificador de instrumentación [31].
Tesis, Carlos A. Alva, Wilfredo Reaño, Joel O. Castillo, Universidad Ricardo Palma, Lima-PERÚ

.

2.4.3 Filtro pasa-bajo de 4to orden

El filtro pasa bajo de 4to orden se compone de 2 filtros de 2do orden en serie tipo Butterworth (ver Figura 2.15). Este tiene como fin ser un filtro anti-aliasing.

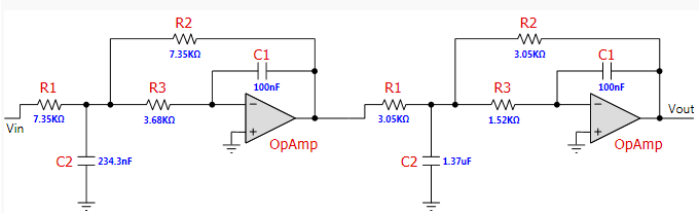


Figura 2.15 Circuito de filtro pasa bajas 200Hz. [32]
Fuente: Programa Filter PRO diseño de Circuitos

2.4.4 Filtro pasa-alto de 4to orden

El filtro pasa alto de 4to orden se compone de 2 filtros de 2do orden tipo Butterworth. Este tiene como fin, ser un filtro anti señal DC de 0Hz por lo que la frecuencia

de corte del filtro es de 0.5Hz. En unión el filtro pasa bajas y pasa altas dan como resultado un filtro pasa banda de 4to orden (ver Figura 2.16).

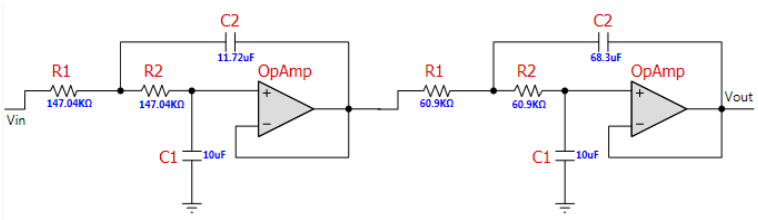


Figura 2.16 Circuito de filtro pasa altas de 0.5Hz [32].
Fuente: Programa Filter PRO diseño de Circuitos

2.4.5 Filtro Notch de 4to orden

El filtro notch está centrada en la frecuencia de rechazo de 60Hz. Se sabe que es el ruido de suministro de corriente eléctrica 110/220AC. Es importante tener un factor de calidad Q elevado para la eliminación del ruido eléctrico de 60Hz. A continuación se muestra en la figura 2.17 el circuito de un filtro notch de 4to orden [31].

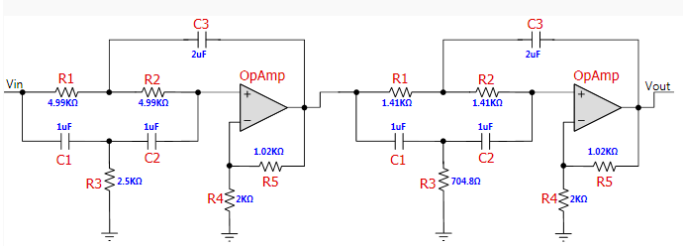


Figura 2.17 Circuito de filtro rechaza banda 60Hz [32].
Fuente: Programa Filter PRO diseño de Circuitos

2.4.6 Convertidor ADC

Los dispositivos ADC convierten un nivel de tensión analógico a valor digital expresado en bits (ver Figura 2.18). Todo convertidor ADC debe procurar que el conjunto de bit obtenidos a la salida sea un reflejo lo más exacto posible del valor analógico correspondiente. Hoy en día la mayoría de los microcontroladores poseen un ADC interno, de no ser así existen convertidores independientes para cumplir dicha función [33].

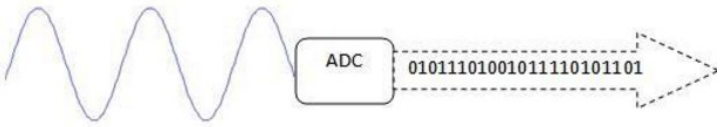


Figura 2.18 Esquema de un convertidor analógico digital (ADC) [34].
Frecuencia fundamental, Glosario Convertidores Analógico Digital

2.5 Desarrollo e innovación en ECG tecnología Analog Front-End (AFE)

En las últimas décadas, avances en el área de microelectrónica han permitido el desarrollo de sistemas portátiles para aplicaciones no clínicas, con el propósito de mejorar la calidad de vida de las personas. Algunos ejemplos de avances importantes son los dispositivos integrados de aplicación específica para la adquisición de biopotenciales desarrollados por las empresas IMEC, Texas Instrument*s* y Analog Devices que han abierto las puertas a varios tipos de aplicaciones portátiles.

Estos nuevos circuitos integrados, conocidos como AFEs ("*Analog Fron-End*" por sus siglas en inglés y como etapa inicial analógica en español), están dirigidos al acondicionamiento de biopotenciales en general. Desde el año 2011, las compañías Analog Devices y Texas Instruments introdujeron circuitos integrados AFE para la aplicación de ECG [35]. En 2012 el microchip AD8232 de la compañía Analog Devices ganó el premio al mejor diseño electrónico de AFE *single-lead heart-rate monitor*, en la categoría de innovación médica publicado en la revista *Electronics Design* [36].

Es importante resaltar que el microchip AD8232 (ver Figura 2.19) es un AFE dirigido a la construcción de un ECG portátil y de bajo costo, ofreciendo facilidades para el diseño ofreciendo así un producto de amplia aplicación entre estas puede utilizarse como monitor portátil (Holter), monitoreo durante el ejercicio (*fitness*), uso en el hogar, y sistemas de ECG ambulatorio y clínicas.

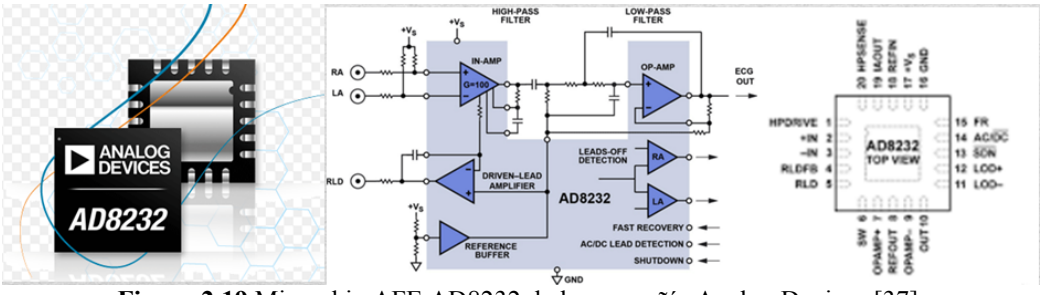


Figura 2.19 Microchip AFE AD8232 de la compañía Analog Devices [37].
Fuente: www.analog devices.com/Datasheet.

Estos circuitos AFE presentan entre las siguientes ventajas, principalmente la reducción de 50 componentes discretos a un microchip con el cual se reduce hasta un 80% del espacio en tableros y módulos, reducen también la captación de ruido, presentan un bajo consumo energético y uso de baterías permitiendo el desarrollo de equipos portátiles de larga vida, y ligeros [36, 37].

2.6 Evaluación de calidad para el desarrollo de un electrocardiógrafo.

Para definir los aspectos de calidad que se evaluarán para este proyecto se puede ver la Tabla 2.1 con las normas IEC, AAMI, ANSI, que describen los requerimientos de diseño para el desarrollo de equipo médico y seguridad básica, estos son estándares internacionales que deben de cumplirse para utilizar un dispositivo médico en el sector hospitalario, uso en el hogar, y comercialización.

Tabla 2.1 Normativas de estándares de calidad y seguridad para el desarrollo de un electrocardiógrafo.

Norma	Regulación
IEC 6060-1 parte 1 [38].	Equipos electromédicos partes 1: Requisitos generales para la seguridad básica y funcionamiento esencial.
ANSI/AAMI/IEC 60601-2-47:2012 [21].	Requisitos particulares para la seguridad básica y el rendimiento esencial de los sistemas electrocardiográficos ambulatorios.
ANSI/AAMI/IEC 60601-2-27:2011[39].	Requisitos particulares para la seguridad básica y el rendimiento esencial del equipo de monitorización electrocardiográfica (no incluye ECG domésticos)
ANSI/AAMI/IEC 60601-2-25:2011[40].	Requisitos particulares para la seguridad básica y el rendimiento esencial de los electrocardiógrafos. (No incluye ECG ambulatorios, pero incluye registros y análisis de electrocardiógrafos monocanal y multicanal.
ANSI/AAMI C12:2000/ (R) 2010 [22].	Electrodos desechables

Entre los requerimientos planteados en estas normas son un ancho de banda de 0.5-40Hz con aplicación para electrocardiógrafos Holter o ambulatorios, que puede ser flexible hasta 0.67-40Hz, a pesar de presentar una distorsión en el complejo QRS está permitido por la AHA, y normativas ANSI, AMMI. Establece también que para monitoreo en modo manual el ancho de banda es 0.67-40Hz, mientras que para monitoreo en tiempo real el indicado es 0.5-40Hz. Para el uso clínico es de 0.05-150Hz. Y para aplicación de pediatría y neonatos 0.05-250Hz (Ver Figura 2.20) [41].

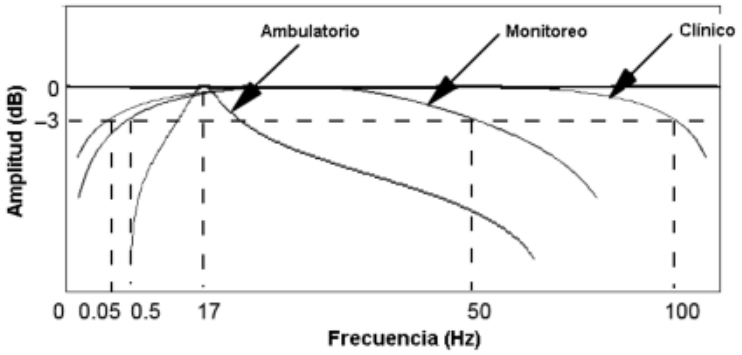


Figura 2.20 Se muestra ancho de banda para aplicación de electrocardiografía [42].

Otras mediciones de seguridad incluidas en las normativas de la tabla 2.1 son: rango dinámico de entrada= 0.5-5mV, exactitud en la ganancia +/- 2%, impedancia de entrada ≥ 2.5 MΩ, ruido del sistema ≤30µV, corrientes de fuga a paciente ≤10µA, voltaje de offset de ±300 mV, CMRR ≥80dB, uso de resistencias de +/- 1%. Los estándares de la IEC, difieren un poco con los de la ANSI y AAMI en algunas mediciones y tolerancias.

Otros aspectos de calidad a considerar en el desarrollo de un equipo son:

CMRR: La ecuación para calcular el CMRR es la siguiente (ver Ec. 3) donde se divide la ganancia diferencial entre la ganancia común [43].

$$CMRR = 20\log_{10}\left(\frac{A_d}{A_s}\right) \tag{3}$$

Ec. 3. Ecuación para calcular CMRR [43].

Relación señal-ruido: La relación señal ruido (S/N) es la diferencia entre el nivel de la señal y el nivel de ruido. Se entiende como ruido cualquier señal no deseada, en el sistema. El ruido se mide sin ninguna señal a la entrada del equipo. Una forma efectiva de calcular la calidad de la señal es mediante el factor de ruido indicado en la Ec.4 que relaciona la S/N de entrada y salida para determinar la atenuación del ruido después de etapas de filtrado o procesamiento o transmisión de una señal.

$$F = \frac{(S/R)ent}{(S/R)sal} \tag{4}$$

Ec. 4. Ecuación para calcular factor de ruido [43].

Relación de rechazo de la fuente de alimentación: Se usa para describir la cantidad de ruido de la fuente de alimentación que un dispositivo puede rechazar.

El PSRR se define como la relación entre el cambio en la tensión de alimentación en el opam a la tensión equivalente (diferencial) de salida que produce. Se expresa en decibeles.

Ruido de interferencia del suministro eléctrico 60Hz: Es el más común y difícil de eliminar, el filtro Notch de 60Hz es el más utilizado para eliminar este ruido. Para poder determinar que el ruido de 60Hz ha sido eliminado es necesario obtener un factor de calidad Q alto, entre mayor sea significa que es mejor calidad de rechazo de banda, la ecuación 5 se utiliza para calcular el factor de calidad [43].

$$Q = \frac{f_0}{f_2 - f_1} = \frac{f_0}{\Delta f} \tag{5}$$

Ec. 5 Ecuación para calcular factor de calidad Q del filtro Notch 60Hz [43].

Ruido offset: es una señal DC de amplitud hasta 300mV causada por la interfaz electrodo-piel, se puede disminuir con el uso de electrodos de mayor calidad o utilización de geles de electrolitos, puede eliminarse por hardware en el diseño del circuito (frecuencia 0Hz) o filtrado digital mediante software [43].

Ruido causado por contracción muscular, otros órganos, y movimiento del usuario: Este tipo de ruido es identificado dentro de las frecuencias cercanas al ancho de banda de 0.05-250Hz, generadas por el movimiento muscular, actividad de otros órganos y movimiento del usuario. Para esto se utilizan filtros definiendo un ancho de banda según la aplicación del ECG [43].

Desviaciones Basales debidas a respiración del paciente representada como señal de corriente alterna de baja frecuencia: este tipo de ruido es difícil de eliminar debido a que se genera por la respiración del usuario y se encuentra dentro del ancho de banda de interés, pero no interfiere con la señal original si no se aprecia una elevación y disminución ligera en el registro del ECG completo por lo que según el interés y aplicación del ECG en ocasiones es despreciable [43].

Ruido electromagnético por interferencia de artefactos electrónicos: este tipo de ruido normalmente se representa por altas frecuencias por lo que al definir nuestro ancho de banda con buena calidad y un blindaje apropiado para los electrodos, y el chasis del instrumento, es suficiente para proteger este tipo de ruido [43].

2.7 Corrientes de fuga.

La mayoría de los regímenes de pruebas de seguridad para equipos médicos eléctricos implican la medición de ciertas "corrientes de fuga", ya que el nivel de estos puede ayudar a verificar si un equipo es eléctricamente seguro. Las corrientes que fluyen desde o entre los conductores que están aisladas de la tierra y entre sí se denominan corrientes de fuga, y normalmente son pequeñas. Sin embargo, dado que la cantidad de corriente requerida para producir efectos fisiológicos adversos también es pequeña, tales corrientes deben estar limitadas por el diseño del equipo a valores seguros (ver Figura 2.21).

Corriente de fuga a tierra: Es la corriente que normalmente fluye en el conductor de tierra de una pieza de equipo conectada a tierra de forma protectora. En condiciones normales, una persona que está en contacto con la carcasa de metal con conexión a tierra del equipo y con otro objeto conectado a tierra no sufrirá efectos adversos, incluso si fluyera una corriente de fuga a tierra bastante grande. Esto se debe a que la impedancia a tierra del recinto es mucho más baja a través del conductor de tierra de protección que a través de la persona.

Corriente de fuga del recinto o corriente táctil: Es la corriente que fluye desde una parte conductora expuesta del gabinete a la tierra a través de un conductor que no sea el conductor de protección a tierra.

Corriente de fuga del paciente: Es la corriente de fuga que fluye a través de un paciente conectado a una parte o partes aplicadas. Puede fluir desde las partes aplicadas a través del paciente a la tierra o desde una fuente externa de alto potencial a través del paciente y las partes aplicadas a la tierra.

Corriente de fuga auxiliar del paciente: Es la corriente que normalmente fluye entre las partes de la parte aplicada a través del paciente, que no está destinada a producir un efecto fisiológico [38,39].

Corriente de fuga (µA)		Corriente de fuga a tierra mA	Corriente de contacto (µA)	Corriente de fuga del paciente CA (µA)	Corriente de fuga del paciente CC (µA)	Corriente de fuga del paciente con aplicación de la tensión de línea (µA)	Corriente auxiliar del paciente (µA)	Corriente auxiliar del paciente (µA)	Corriente auxiliar del paciente (µA)
Tipo B	NC	5	100	100	10	–	100	10	100
	SFC	10	500	500	50	–	500	50	500
Tipo BF	NC	5	100	100	10	–	100	10	100
	SFC	10	500	500	50	5000	500	50	500
Tipo CF	NC	5	100	10	10	–	10	10	10
	SFC	10	500	50	50	50	50	50	50

Figura 2.21 Estándar de corrientes de fuga máximo permitido IEC 60601-1[39,44].

IEC60601 AAMI/NFPA 99

La principal norma de seguridad eléctrica para los equipos médicos es la IEC 60601. En esta norma, cada instrumento tiene una clase:

A) Clase I: Parte con corriente, recubierta de aislamiento básico y tierra de protección.

B) Clase II: Parte con corriente recubierta de aislamiento doble o reforzado.

C) Clase IP: Fuente de alimentación interna.

Cada pieza o terminal aplicado al paciente es de un tipo determinado.

Tipo B1: Pieza aplicada al paciente puesta a tierra.

Tipo BF: Pieza aplicada al paciente flotante (conductor de superficie).

Tipo CF: Pieza aplicada flotante para su uso en contacto directo con el corazón [39,44].

2.8 Estabilidad de un instrumento de medición.

2.8.1 Fiabilidad vs Robustez.

Uno de los aspectos importantes a considerar para el diseño de un equipo es la confiabilidad del parámetro medido, y la robustez del mismo, puede ser un aparato confiable sin llegar a ser robusto debido a que al funcionar bajo las condiciones ambientales como temperatura, ruido, presión, el aparato será completamente fiable pero alterando estas condiciones de trabajo puede presentar inestabilidad, o debido al desgaste de los componentes por el tiempo pueden llegar a tener un porcentaje de variación mayor al establecido en la hoja de datos por consiguiente pueden ocasionar la variación de mediciones y por ende inestabilidad del sistema, por lo que al diseñar un equipo se busca que este sea fiable y robusto es decir que su rango de trabajo u operación se mantenga

siempre fiable aun presentes dichas variaciones. Para el diseño de un electrocardiógrafo es muy importante la precisión de sus componentes por normativa de la IEC 60601-2-47, 60601-2-25, y 60601-2-27 el porcentaje de tolerancia permitido en los componentes es de 1%, por lo que es importante utilizar técnicas de análisis como Monte Carlo, Worst Case, y Variación de temperatura para predecir el comportamiento del sistema y evitar errores y condiciones de falla en el equipo. También se utilizó Análisis Fourier para observar la señal ECG del sistema en términos de frecuencia [45, 46, 47].

2.8.2 Análisis Monte Carlo.

El análisis de Monte Carlo se utiliza para simulaciones con un error dado en diferentes componentes. Esta prueba es muy útil para visualizar cómo el circuito se ejecutará con componentes imperfectos como se utilizan en la realidad. El análisis de Monte Carlo permite al usuario introducir dos cosas:

1.- El porcentaje de error de los componentes.

2.- El tipo de error, *Gauss* o *Uniforme*. La simulación en el modo Gauss (más realista que el uniforme) utiliza un enfoque de curva de campana, que escoja al azar valores en el rango de error especificado [48].

2.8.3 Análisis Worst Case.

Es una técnica estadística que utiliza un enfoque de "qué pasaría si" para proporcionar un resultado práctico y su objetivo principal es identificar los componentes más críticos de un circuito. Este análisis ayuda a responder a la pregunta: ¿Cuáles serán los peores efectos posibles de las variaciones de los componentes?

El análisis del Worst case nos permite conocer el valor crítico de los componentes, y la respuesta de la señal de salida del sistema para hacer consideraciones en el diseño del instrumento y así asegurar la estabilidad del sistema [48].

2.8.4 Análisis Fourier.

El análisis Fourier es un método que permite analizar una señal de un sistema para saber el comportamiento de este sistema en el dominio de la frecuencia, consiste en descomponer la señal de estudio en un número infinito de señales seno y coseno (armónicos) con frecuencia, magnitud, y fase que al recuperar o sumarse estas señales se obtiene como resultado la señal original. Esta técnica tiene muchas aplicaciones para

razones del ECG mediante el análisis de Fourier se puede ver en qué frecuencias la magnitud de la señal es mayor y con esto se pueden hacer consideraciones para el desarrollo de los filtros y acondicionamiento de la señal mediante software [48].

2.8.5 Análisis Barrido de temperatura.

Esta técnica de análisis nos ayuda a conocer la respuesta que tendrán los componentes electrónicos ante variaciones o un rango de temperatura, con el fin de conocer la señal de salida y tener consideraciones sobre los rangos mínimos y máximos de temperatura para la operación de un circuito o sistema en desarrollo. Cuando se ejecuta una simulación en Multisim, se asumen todos los elementos del circuito a medir a la temperatura nominal de 27°C. Esta temperatura se puede cambiar mediante la personalización de las opciones [48].

2.9 Plataforma Arduino y código abierto.

En la actualidad una tendencia que ha permitido un incremento en el desarrollo tecnológico es la utilización de tecnologías *open source*, y *open hardware*, Arduino es un ejemplo de estas tecnologías, al tener una plataforma de código abierto diseñada para crear prototipos utilizando electrónica libre tanto en hardware como en software. Esta plataforma consiste en una placa de circuito impreso de tamaño reducido con elementos electrónicos de bajo costo y bajo consumo energético, así como un ambiente de desarrollo abierto (programación) con librerías para el control de la placa. Esta herramienta enriquece el proceso de enseñanza-aprendizaje mediante la experimentación, es accesible para estudiantes, profesores e investigadores, de tal manera que puedan contar con una plataforma de desarrollo de proyectos en ciencias, tecnologías e ingeniería. [49] Algunas de las placas de esta plataforma son: Arduino UNO, Arduino Nano, Arduino Mega, y Arduino DUE con características específicas, para diferentes aplicaciones.

2.10 Arduino Nano

2.10.1 Plataforma de código abierto (Arduino Nano).

2.10.1.1 Arduino

La placa Arduino Nano es un dispositivo que integra elementos de hardware, software y lenguaje de programación de libre acceso (Figura 2.22). Los elementos básicos que constituyen la placa Arduino Nano son: un microcontrolador ATmega 328 que trabaja con un voltaje de operación de 5 volts, un reloj de 16MHz, memoria EEPROM de 1KB, y un regulador de voltaje integrado por lo que se puede alimentar externamente con una fuente de alimentación que opere en un rango de 7-12 volts con un consumo de corriente de 15mA. Tiene también seis entradas analógicas, y 14 puertos digitales los cuales puedes configurar como entrada o salida, una memoria flash de 32 KB para alojar el programa (s*ketch*) que se ejecutará una vez grabado, de los cuales 0.5KB son usados por el programa llamado *bootloader* el cual sirve para poder cargar nuestros *sketch* directamente del PC sin utilizar hardware adicional también permite utilizar un sistema de *reset* mediante el botón integrado en la placa. Otras características pueden verse en la Tabla 2.2 [50].

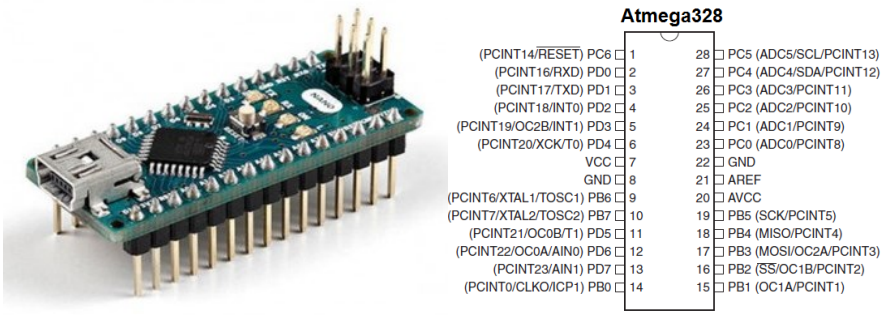


Figura 2.22 Secciones de Arduino Nano conexiones y microcontrolador ATmega328 pines.

Tabla 2.2 Especificaciones técnicas Arduino Nano.

Microcontrolador	ATmeta328
Tensión de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (limite)	6-20V
E/S digitales TTL	14 (6 con PWM)
PWM digital pines I/O	6
Pines de entrada analógica	6
Corriente continua para pin I/O	20mA
Corriente CC para Pin 3.3V	50mA
Memoria flash	32KB (ATmega328)
SRAM	2KB
EEPROM	1KB (ATmega328)
Velocidad de reloj	16MHz

2.10.1.2 IDE interfaz de desarrollo.

El entorno de desarrollo integrado también llamado IDE (sigla en inglés de *Integrated Development Environment*), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios lenguajes. IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación; es decir, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (Ver figura 2.23). Además, en el caso de Arduino incorpora las herramientas para cargar el programa ya compilado en la memoria del hardware [51].

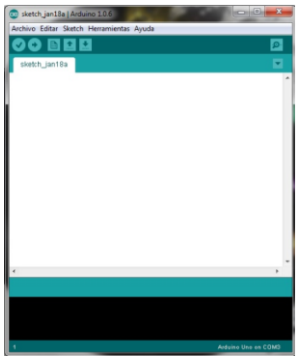


Figura 2.23 Entorno de programación de Arduino software IDE.

2.11 Modo de operación grabación en micro SD data logger.

2.11.1 Modo de Comunicación serial.

La comunicación serial es un protocolo muy común para comunicación entre dispositivos que se incluye de manera estándar en prácticamente cualquier computadora. La comunicación serial es también un protocolo común utilizado por varios dispositivos para instrumentación que incluyen un puerto RS-232 (estándar ANSI/EIA-232), además puede ser utilizada para adquisición de datos si se usa en conjunto con un dispositivo remoto de muestreo.

El concepto de comunicación serial es sencillo, el puerto serial envía y recibe bytes de información un bit a la vez (Figura 2.24). Aun y cuando esto es más lento que la comunicación en paralelo, que permite la transmisión de un byte completo por vez, este método de comunicación es más sencillo y puede alcanzar mayores distancias. Por ejemplo, la especificación IEEE 488 para la comunicación en paralelo determina que el largo del

cable para el equipo no puede ser mayor a 20 metros, por el otro lado, utilizando comunicación serial el largo del cable puede llegar a los 1200 metros [52].

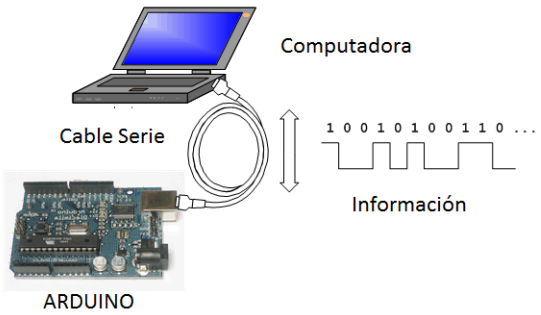


Figura 2.24 Comunicación en serie de PC a Arduino.

Típicamente, la comunicación serial se utiliza para transmitir datos en formato ASCII. Para realizar la comunicación se utilizan 3 líneas de transmisión: (1) Tierra (o referencia), (2) Transmitir, (3) Recibir. Debido a que la transmisión es asincrónica, es posible enviar datos por una línea mientras se reciben datos por otra. Las características más importantes de la comunicación serial son la velocidad de transmisión, los bits de datos, los bits de parada, y la paridad. Para que dos puertos se puedan comunicar, es necesario que las características sean iguales.

Velocidad de transmisión *(baud rate)*: Indica el número de bits por segundo que se transfieren, y se mide en baudios (*bauds*). Por ejemplo, 300 baudios representan 300 bits por segundo. El estándar de velocidad entre dispositivos es de 9600 baudios, es posible tener velocidades más altas, pero se reduciría la distancia máxima posible entre los dispositivos. Las altas velocidades se utilizan cuando los dispositivos se encuentran uno junto al otro.

Bits de datos: Se refiere a la cantidad de bits en la transmisión. Cuando la computadora envía un paquete de información, el tamaño de ese paquete no necesariamente será de 8 bits. Las cantidades más comunes de bits por paquete son 5, 7 y 8 bits. El número de bits que se envía depende en el tipo de información que se transfiere. Por ejemplo, el ASCII estándar tiene un rango de 0 a 127, es decir, utiliza 7 bits; para ASCII extendido es de 0 a 255, lo que utiliza 8 bits. Si el tipo de datos que se está transfiriendo es texto simple (ASCII estándar), entonces es suficiente con utilizar 7 bits por paquete para la comunicación. Un

paquete se refiere a una transferencia de byte, incluyendo los bits de inicio/parada, bits de datos, y paridad.

Bits de parada: Usado para indicar el fin de la comunicación de un solo paquete. Debido a la manera como se transfiere la información a través de las líneas de comunicación y que cada dispositivo tiene su propio reloj, es posible que los dos dispositivos no estén sincronizados. Por lo tanto, los bits de parada no sólo indican el fin de la transmisión sino además dan un margen de tolerancia para esa diferencia de los relojes.

Paridad: Es una forma sencilla de verificar si hay errores en la transmisión serial. Existen cuatro tipos de paridad: par, impar, marcada y espaciada. La opción de no usar paridad alguna también está disponible [52].

2.12 Modo de operación grabación en micro SD data logger.

2.12.1 Construcción de un data logger.

Un data logger es un dispositivo electrónico que registra mediciones ordenadas en el tiempo, provenientes de uno o más sensores. Cada medición proveniente del sensor es almacenada en una memoria, junto con su respectiva fecha y hora. En general los data logger son pequeños, y alimentados por baterías independientes, están conformados por un sensor o transductor que nos proporcionara la señal analógica, un microcontrolador con convertidor analógico digital interno, o en caso de no incluirlo se requerirá uno externo, y una memoria para el almacenamiento de los datos [53].

2.12.2 Almacenamiento en memoria para un data logger.

2.12.2.1 SD card y formato FAT16.

El Arduino no está dotado con memoria RAM ni memoria masiva para almacenar datos en grandes cantidades. En efecto, la memoria EEPROM integrada en el microprocesador solo puede almacenar unos pocos cientos de bytes en el mejor de los casos. Para almacenar volúmenes considerables de datos, lo cual es requerido la mayor parte de las aplicaciones debemos recurrir a tarjetas de memoria SD o SDHC.

Una memoria SD (por sus siglas "*Secure Digital*") es una tarjeta de memoria basada en la tecnología Flash la cual es de tipo no volátil, y permiten conservar la información guardada sin necesidad de alimentación eléctrica hasta por 10 años, y soportan ciclos de escritura de hasta 10,000 ciclos. Tienen sus variantes de tamaño como son micro SD card pero operan

bajo la misma tecnología, protocolo de información (SPI), y con sistemas de codificación de ficheros FAT16 y FAT32.

Para entender un sistema de archivos, véanse los clusters como páginas de memoria. Para optimizar su acceso, al igual que en un libro, se dispondrá de un índice, llamado FAT (*File Allocation Table*); con punteros que señalen de algún modo a los clusters. Según estos punteros sean de 12, 16 ó 32 bits, se llamará al sistema de archivos FAT12, FAT16 o FAT32 respectivamente. Para utilizar la librería SPI debes considerar que es compatible con ficheros FAT16 y FAT32, dependiendo de la memoria que utilizaras por ejemplo las memorias SD son compatibles para ambos formatos, mientras que las SDHC únicamente FAT32. En el caso de nuestro data logger al utilizar micro SD es indistinto el formato que se le dé, aunque para optimizar la velocidad es preferente realizar formato en FAT16 ya que a pesar de tener un pequeño desperdicio de información entre sus desventajas lo compensa al ser más rápido y en el caso contrario el formato FAT32 al contener ficheros más grandes optimiza el desperdicio de información contra un poco de tiempo más al realizar la función de acceder a ella [54].

2.12.2.2 Módulo data logger shield.

Para realizar el almacenamiento de información utilizó el módulo data logger, (Figura 2.25) el cual trabaja mediante el protocolo de comunicación SPI, y se utilizan 4 entradas digitales que van del microcontrolador a este módulo los cuales son SCLK, MISO, MOSI, y SS o CS. También tiene incorporado un reloj de tiempo real (RTC) DS1307 y una batería 1220 con el cual se puede programar para mantener la hora y fecha hasta por 10 años. Este RTC trabaja con el protocolo de comunicación I2C, que funciona con solo 2 líneas SDA y SCL que transportarán la información entre los dispositivos conectados a este bus, este bus permite la comunicación entre múltiples dispositivos conectados y cada dispositivos es reconocido por un código (dirección).

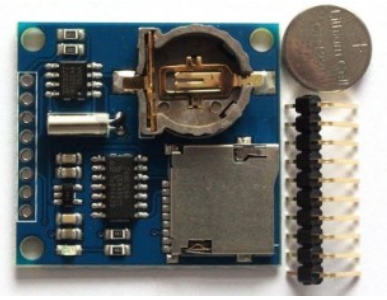


Figura 2.25 Módulo data logger shield.

2.12.2.3 Protocolo de comunicación SPI.

Para el desarrollo de un data logger con aplicación de ECG con Arduino se utilizó el protocolo de comunicación SPI del microcontrolador ATmega328. El bus o protocolo SPI (*Serial Peripheral Interface*) es básicamente un bus de comunicación a nivel de circuitos integrados. La transmisión de datos se realiza en serie, es decir un bit después de otro (Figura 2.26). El bus SPI se define mediante 4 pines:

- SCLK o SCK: Señal de reloj del bus. Esta señal rige la velocidad a la que se transmite cada bit.
- MISO (*Master Input Slave Output*): Es la señal de entrada a nuestro dispositivo, por aquí se reciben los datos desde el otro integrado.
- MOSI (*Master Output Slave Input*): Transmisión de datos hacia el otro integrado.
- SS o CS: *Chip Select* o *Slave Select*, habilita el integrado hacia el que se envían los datos. Esta señal es opcional y en algunos casos no se usa [55].

Figura 2.26 Conexiones y la transmisión típica de un byte por SPI.

El funcionamiento para un envió de un master es el siguiente: Se habilita el chip al que hay que enviar la información mediante el CS.

1. Se carga en el buffer de salida el byte a enviar.
2. La línea de Clock empieza a generar la señal cuadrada donde normalmente por cada flanco de bajada se pone un bit en MOSI.
3. El receptor normalmente en cada flanco de subida captura el bit de la línea MISO y lo incorpora en el buffer.

Se repite el proceso 8 veces para transmitir un byte (Figura 2.26). Una vez transmitido el byte se vuelve a poner la línea CS en reposo. Hay que tener en cuenta que a la vez que el Master está enviando un dato también lo recibe así que si el Slave ha depositado algún byte en el buffer de salida, este también será enviado y recibido por el Master. Visto cómo se envía un byte en SPI vamos a ver sus principales ventajas.

1. Comunicación *Full-Duplex*, es decir es capaz de enviar y recibir datos al mismo tiempo.
2. Muy simple y usado en los integrados.
3. Velocidades de comunicación relativamente elevadas.

También tiene sus desventajas que son las siguientes:

1. No hay control del flujo de datos por hardware.
2. Carece de confirmación de la recepción como si ocurre en I2C. Es decir no sabemos si el mensaje ha llegado al destino, es necesario utilizar hardware si se requiere confirmar la información.
3. Usa más pines que otros buses, ya que necesita uno por cada esclavo.
4. Funcionamiento a distancias cortas [55].

2.12.2.4 Reloj de Tiempo Real (RTC).

Como ya se mencionó el módulo data logger (figura 2.25) tiene incluido un reloj DS1307, el cual funciona protocolo I2C, y básicamente consiste en un reloj programable, al cual el usuario mediante código BCD de bits especificado en la hoja de datos, sube una hora y fecha que el usuario especifique. Y posteriormente con otro código esta información puede ser recuperada para vaciar la información (hora y fecha) a otro dispositivo o visualizarla según la aplicación. A continuación se muestra el código para subir y vaciar la hora y fecha.

2.12.2.5 Protocolo de comunicación I2C.

El bus I²C, un estándar que facilita la comunicación entre microcontroladores, memorias y otros dispositivos con cierto nivel de "inteligencia", sólo requiere de dos líneas de señal y un común o masa. Fue diseñado a este efecto por la compañía *Philips* y permite el intercambio de información entre muchos dispositivos a una velocidad aceptable, de

unos 100 Kbits por segundo. La metodología de comunicación de datos del bus I²C es en serie y sincrónica. Una de las señales del bus marca el tiempo (pulsos de reloj) y la otra se utiliza para intercambiar datos [56].

Como dijimos, las líneas SDA y SCL transportan información entre los dispositivos conectados al bus. Cada dispositivo es reconocido por su código (dirección) y puede operar como transmisor o receptor de datos. Además, cada dispositivo puede ser considerado como *Master* o *Slave*.

El Master es el dispositivo que inicia la transferencia en el bus y genera la señal de Clock. El Slave (esclavo) es el dispositivo direccionado. Las líneas SDA (*serial data*) y SCL (serial clock) son bidireccionales, conectadas al positivo de la alimentación a través de las resistencias de *pull-up*. Cuando el bus está libre, ambas líneas están en nivel alto. La transmisión bidireccional serie (8-bits) de datos puede realizarse a 100Kbits/s en el modo estándar o 400 Kbits/s en el modo rápido. La cantidad de dispositivos que se pueden conectar al bus está limitada, solamente, por la máxima capacidad permitida: 400 pF.

Las características más importantes del bus I2C son:

- Se necesitan solamente dos líneas, la de datos (SDA) y la de reloj (SCL).
- Cada dispositivo conectado al bus tiene una dirección seleccionable por software. Teniendo una relación entre dispositivos *Master*/*Slave*.
- El bus permite la conexión de varios masters, ya que incluye un detector de colisiones.
- Los datos y direcciones se transmiten en paquetes de 8 bits [56].

2.12.3 Comandos para escritura en SD card.

Para el desarrollo del código de un data logger o registro de datos en micro SD card es necesario hacer consideraciones al momento de acceder a la memoria física SD para transcribir la información, así como el método en que se realiza el muestreo de la señal de interés. Esto debido a que esto lleva tiempo y puede causar perdida o disminuir el flujo de información. Los métodos empleados en este proyecto para realizar el muestreo de la señal de entrada son: utilizar el comando **delay** nulo, y mediante interrupciones. Y para grabar la información en la memoria física SD se utilizaron 2 comandos: **myfile.print** y **myfile.write**. En el código A8 de la sección de anexos se utilizó el método de muestreo con **delay** nulo, y el comando **myfile.print** para la escritura en la memoria SD, este método por

lo general es utilizado para sistemas de adquisición de datos o data logger con tasas de muestreo muy bajas, por ejemplo sistemas donde se mide temperatura, o niveles de humedad de 1 o 2 muestras por minuto. Esto se debe a que el comando **myfile.print** fuerza la escritura en el medio físico cada vez que se ejecutan, es decir accede a la memoria SD para escribir la información y esto puede tardar hasta 10 ms, causando un retardo entre cada dato escrito y limitando el número de datos por segundo.

En el código A9 (ver sección de anexos) se utilizó el método de interrupciones y para escribir los datos en la memoria SD se utilizó el comando **myfile.write** a diferencia del comando anterior este utiliza la memoria RAM del microcontrolador como un espacio destinado a utilizarse como buffer de bloque, este espacio se encarga de retener los datos recibidos, y al llenarse se transcriben en la memoria física es decir espera a llenar todo el bloque antes de acceder a la memoria física SD reduciendo así el tiempo de escritura y priorizando el muestreo por las interrupciones siendo más eficiente para sistemas de adquisición con tasas elevadas de datos.

2.13 Modo de operación transmisión datos por bluetooth.

2.13.1 Comunicación bluetooth.

Las comunicaciones inalámbricas están presentes en muchas de nuestras actividades diarias asimismo, la creación de estándares de comunicaciones inalámbricas en las redes de transmisión de datos ha abierto oportunidades de desarrollo de estas tecnologías.

Bluetooth forma parte de las tecnologías creadas para proveer comunicación inalámbrica en áreas de uso personal. Sin embargo, su uso va más allá de la eliminación de cables, ya que es lo suficientemente flexible para permitir la creación de aplicaciones en el desarrollo de nuevos dispositivos. El estándar bluetooth está dado por especificaciones técnicas como la velocidad de transmisión estándar entre dispositivos es de 9600 bauds, y otras especificaciones técnicas que son utilizadas para la interoperabilidad de las compañías que fabrican dispositivos con comunicación bluetooth, y también tiene perfiles específicos para aplicaciones como son: acceso genérico, identificación de servicio, puerto serial, acceso a LAN sincronización y el de dispositivo de información móvil (MIDP) [57].

Características

Tecnología inalámbrica. Reemplaza la conexión alámbrica en distancias que no exceden los 10 metros, alcanzando velocidades del rango de 1Mbps.

- **Bajo consumo de potencia.** Lo pequeño de los dispositivos y su portabilidad requieren de un uso adecuado de la energía, el cual provee esta tecnología.
- **Bajo costo.** Los dispositivos de comunicación con comunicación bluetooth son de aplicación personal por lo que el costo debe ser bajo.
- **Integración de servicios.** Puede soportar transmisiones de voz y datos de manera simultánea.
- **Seguridad.** Utiliza *Spread Spectrum Frequency Hopping* como técnica de multiplexaje, lo que disminuye el riesgo de que las comunicaciones sean interceptadas o presenten interferencia con otras aplicaciones. [57]

CAPÍTULO 3

EXPERIMENTACIÓN

En esta sección se describe el procedimiento de cada etapa de la experimentación del proyecto desde la evaluación y configuración del microchip AD8232, simulaciones, implementación del circuito de prueba, adquisición y validación de la señal ECG, caracterización de la plataforma Arduino y digitalización de la señal así como su grabación y transmisión a la interfaz de visualización.

3.1 Evaluación del microchip AD8232.

3.1.1 Estructura del circuito integrado AD8232.

Este microchip desarrollado por la compañía Analog Devices pertenece a la familia de microchips multifunción AFE. Su configuración interna (ver Figura 3.1) ofrece un amplificador de instrumentación con ganancia fija de 100 y un arreglo de amplificadores para acondicionar la señal con filtros hardware, en un encapsulado con dimensiones de 4x4 mm. Ofrece además un bajo consumo de energía, reducción de ruido, larga vida y bajo costo [35]. Para el arreglo de amplificadores se utilizó un filtro pasa altas de 0.5Hz y un filtro pasa bajas de 40Hz con una amplificación de 11, siendo la ganancia total de 1100.

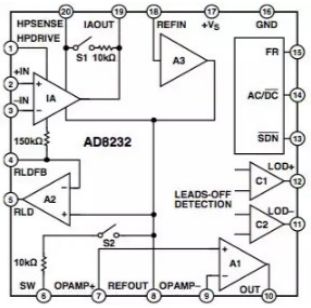


Figura 3.1 Configuración interna del microchip AD8232 [35].

3.1.2 Simulador cardíaco SimCube SC-5

El simulador cardíaco SimCube (Figura 3.2) es una herramienta que ofrece simulador de ECG, respiración, arritmias, y marcapasos para pruebas biomédicas precisas. Para la obtención de la señal se utilizaron las derivaciones del triángulo de Einthoven RA, LA, y RL. Se conectaron a las terminales del tablero de evaluación AD8232.

Figura 3.2 Simulador SimCube SC-5 y sus derivaciones.

3.1.3 Adquisición de la señal

La adquisición de la señal se realizó mediante electrodos con cableado snap y el simulador SimCube, posteriormente se realizó la conexión con un paciente sin antecedentes de problemas cardíacos utilizando electrodos desechables Ag/AgCl en condiciones de reposo, con la posición indicada en la figura 3.3.

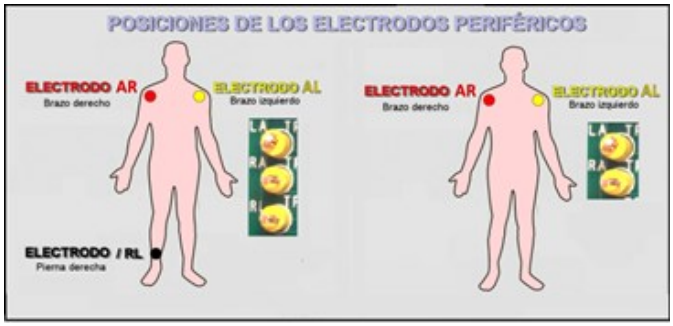


Figura 3.3 Posición de los electrodos y conectores para la etapa de adquisición de la señal.

3.1.4 Tablero EVAL-AD8232 configuración 2 y 3 electrodos.

El tablero de evaluación AD8232 tiene 2 configuraciones para la adquisición de la señal de ECG la cual se puede configurar mediante 3 interruptores y 1 *jumper* (ver figura 3.4). Para la configuración de 3 electrodos es necesario colocar los interruptores del tablero de evaluación como lo indica la tabla 3.1, y para la configuración de 2 electrodos deben colocarse en la posición indicada por la tabla 3.2 [37].

Tabla 3.1 Configuración para 3 electrodos.

Nombre	Posición	Configuración
S1	FR_EN	Fast restore enabled
S2	DC	DC leads off detection
S3	EN	Operación enabled
INPUT BIAS SELECTION	P3	Input bias set by RLD and VS

Tabla 3.2 Configuración para 2 electrodos.

Nombre	Posición	Configuración
S1	FR_EN	Fast restore enabled
S2	AC	DC leads off detection
S3	EN	Operación enabled
INPUT BIAS SELECTION	P4	Input bias set by RLD and VS

En la Figura 3.4 se puede apreciar el tablero de evaluación en sus bloques:

1 Alimentación (3.3V, GND).
2 Interruptores (FR, S2, S3).
3 *Jumper input Bias* (P1, P2, P3, P4).
4 Señal entrada (LA, RA, RL).
5 Señal salida (Out, GND).
6 Microchip AD8232

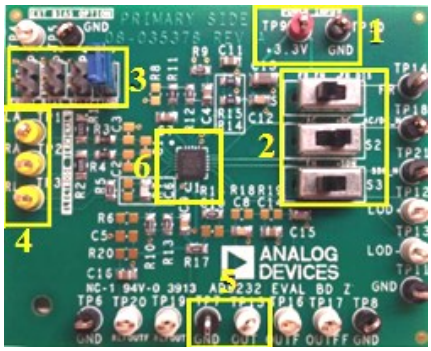


Figura 3.4 Tablero EVAL-AD8232.

3.1.5 Ensamblaje de adaptador SMD.

Una vez se obtenidas las señales y se y comprobada la utilidad del chip AD8232, se acondicionó un microscopio y estación de soldar como estación de trabajo (ver Figura 3.5) para adaptar el microchip de montaje superficial (SMD) a un adaptador DIP para utilizar en breadboard y realizar un circuito de ECG con aplicación para monitor cardíaco, y obtener una señal ECG útil para digitalizar.

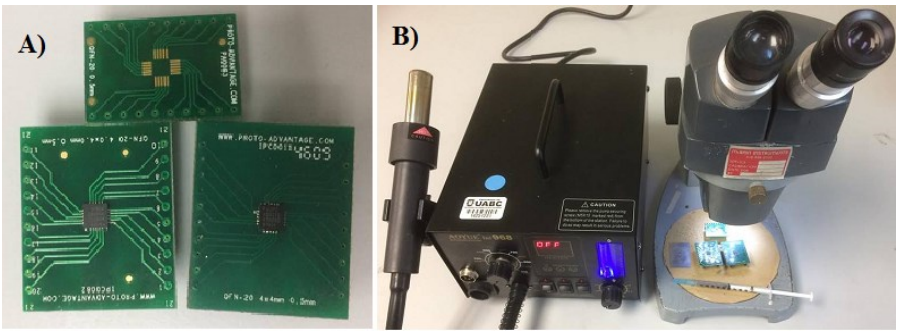


Figura 3.5 A) Adaptador SMD a DIP con el microchip AD8232 y B) Estación de trabajo.

3.1.6 Pruebas en breadboard.

Después de soldar el microchip al adaptador para breadboard, se realizó el circuito propuesto por la compañía Analog Devices (ver Figura 3.6) con los filtros necesarios para la adquisición de la señal de ECG con aplicación de monitor cardíaco de una derivación. Se adquirió una señal ECG de paciente con configuración de 3 electrodos y se realizaron las

mediciones de sus ondas, intervalos, segmentos y ruido en la señal para validar su aplicación como monitor cardíaco.

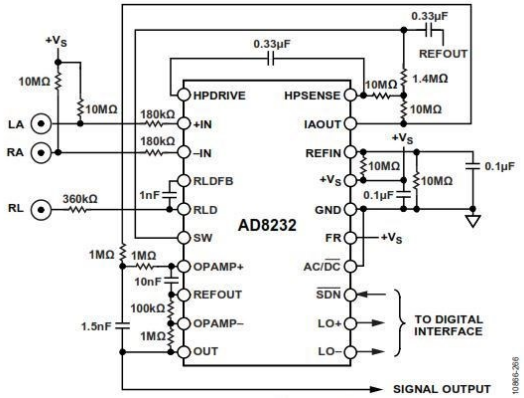


Figura 3.6 Circuito de prueba #1 de ECG con el microchip AD8232 propuesto por la compañía Analog Devices (Aplicación de monitor cardíaco).

3.1.7 Simulación.

3.1.7.1 Creación de componente SPICE.

Una vez que se evalúo y comprobó el funcionamiento del chip AD8232 con el circuito de prueba #1 se utilizó la herramienta Utilboard que se instala en conjunto con Multisim para crear el *footprint* y modelo 3D (Ver figura 3.7), ya que son necesarios para crear el componente SPICE en el software Multisim y realizar los análisis de estabilidad del sistema.

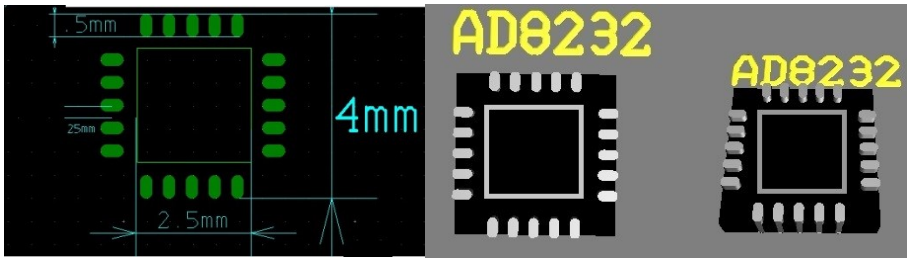


Figura 3.7 A) Footprint de componente AD8232 y B) Modelo 3D.

Finalmente se creó el componente SPICE y en conjunto de una herramienta de simulación de onda ECG de la compañía National Instruments LABVIEW [58] se realizó el circuito de la Figura 3.8 con el componente AD8232 en el software Multisim.

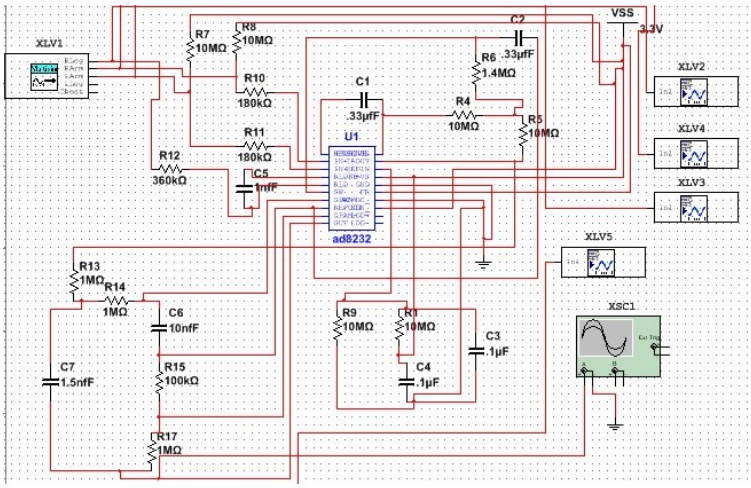


Figura 3.8 Circuito con componente SPICE y herramienta de simulación ECG.

3.1.7.2 Estabilidad de un Instrumento de Medición (Fiabilidad vs Robustez)

Como se describió en la sección de marco teórico, durante el diseño de un equipo es necesario considerar la confiabilidad y robustez, para que este pueda ser capaz de funcionar de forma fiable bajo condiciones ambientales no controlables como: temperatura, ruido, presión, y desgaste. En la experimentación se simuló algunas técnicas de análisis como: Monte Carlo, Worst Case y Variación de temperatura. Se utilizó análisis Fourier para observar la señal ECG del sistema en términos de frecuencia [45]. Para utilizar estos análisis en el software Multisim es necesario seleccionar la pestaña "*Simulate"*, la opción de "*analyses and simulation*" y seleccionar el método de análisis al que se desea someter el circuito (ver figura 3.9). Los resultados obtenidos se pueden observar en la sección 4.3.

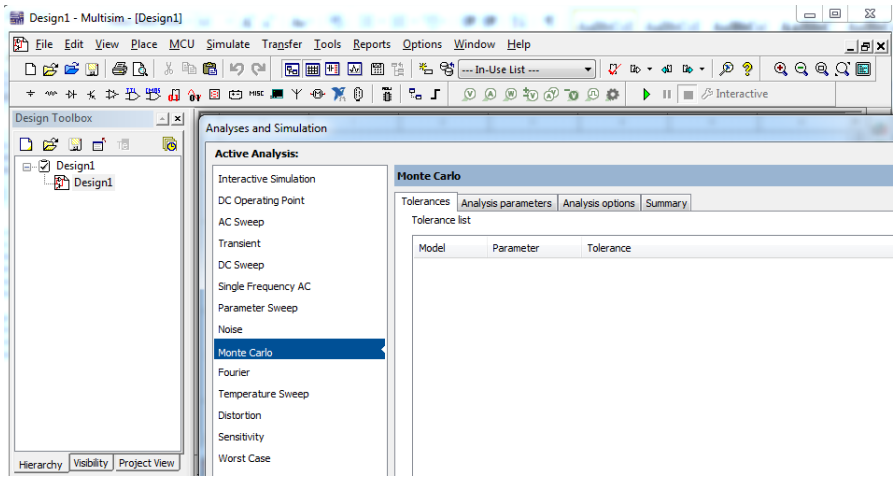


Figura 3.9 Software Multisim análisis y simulación.

3.2 Arduino Nano

3.2.1 Caracterización del Arduino Nano para desarrollo de un electrocardiógrafo.

Para la aplicación en un electrocardiógrafo es importante considerar la etapa de conversión analógico-digital, para llevar acabo un registro confiable y preciso es importante tener en cuenta dos conceptos para la realización de cálculos y validación de dicha etapa el cual es resolución y frecuencia de muestreo.

3.2.1.1 Resolución

La resolución se define como la menor diferencia entre dos valores sucesivos que puede detectar el ADC. El conversor del Arduino cuenta con una resolución de 10 bits (2^{10}), lo cual limita el rango de salida a 1024 valores. Teniendo en cuenta que el conversor es lineal, el rango de entrada varía de 0-5V, entonces la menor diferencia entre dos valores sucesivos de voltaje se obtiene al dividir 5V /1024 es 4.88 mV. Se sabe que la señal del ECG sin amplificar está en el orden de 0-3mV, por si sola sería imposible realizar la conversión, pero con el circuito de prueba #1 (etapa inicial analógica) la salida varia de 0-5V estando en el rango de conversión de la señal analógico-digital [59].

3.2.1.2 Frecuencia de muestreo

La frecuencia de muestreo indica el número de conversiones que puede realizar el convertidor A/D por unidad de tiempo. Según la hoja de datos el convertidor del chip ATmega328 puede trabajar a una frecuencia máxima de muestreo de 9.6KHz idealmente. Es posible aumentar la frecuencia de muestreo disminuyendo la resolución del convertidor, pero para la aplicación de ECG es suficiente estos rangos de resolución y frecuencia de muestreo [59].

3.2.2 Medición del tiempo del convertidor A/D, y transmisión de datos por interrupciones (ISR).

Para controlar la frecuencia de muestreo se puede utilizar 2 métodos: el método por delay, e interrupciones. Al agregar el comando "**delay**" después de realizar una lectura el microcontrolador espera un intervalo de tiempo que se establece en milisegundos, antes de leer el siguiente valor del puerto analógico, manipulando así la frecuencia de muestreo. Este método presenta una importante desventaja debido a que al utilizar el comando "**delay**" el microcontrolador se congela y no puede realizar otra función hasta terminar el tiempo de espera establecido. Otro método con mayor precisión es el de interrupciones las

cuales se ejecutarán en un determinado intervalo de tiempo para controlar el número de muestras leído en 1 segundo y así manipular la frecuencia de muestreo. Este método es más eficaz y preciso debido a que interrumpe el programa cada que pase el tiempo establecido en la interrupción para priorizar la adquisición de información.

3.2.2.1 Medición del tiempo del convertidor A/D (ADC) por método delay.

Para medir el tiempo en que el ADC puede realizar la conversión de una lectura y desplegarla al puerto digital, se utilizó como entrada al sistema una señal senoidal conocida de 500Hz con una amplitud 2Vpp (con offset de 2V), la señal digitalizada se envió al puerto D para obtener la señal digital y se recuperó mediante convertidor digital-analógico de 8 bits (DAC) para medir con el osciloscopio el tiempo entre lecturas del convertidor A/D (Ver figura 3.10).

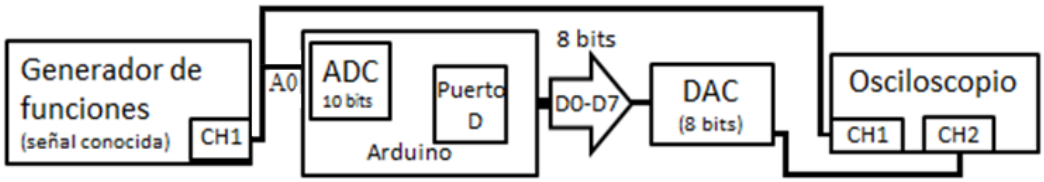


Figura 3.10 Esquema a bloques para medir el tiempo del convertidor A/D.

Para la adquisición de la señal analógica es necesario declarar un pin analógico como sensor o entrada del sistema (A0) con el comando "**AnalogRead**". Se configuró el puerto D como salida digital con el comando "DDRD=0xFF", en la figura 3.11 (o sección de anexos A1) se observa el código para obtener la señal y desplegarla al puerto D. Debido a que el DAC utilizado es de 8 bits y el convertidor A/D es de 10 bits se realiza un corrimiento a la derecha al momento de realizar la lectura, esto para escalar la señal de 10 a 8 bits. Se puede ver también un "**delay**" que se ha desactivado, ya que lo que se busca es la frecuencia máxima de trabajo del convertidor A/D el "**delay**" debe ser nulo.

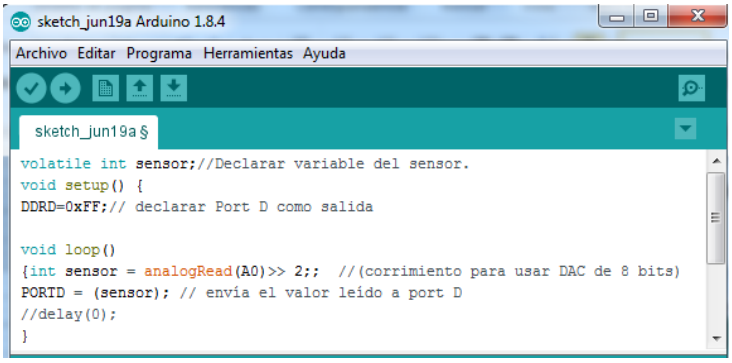


Figura 3.11 Código para leer una señal analógica y desplegarla al puerto D.

Se realizó la medición de la señal de 500Hz y el tiempo de conversión entre cada lectura fue de 112µs, (ver Fig. 3.12 y 3.13). Si calculamos el número máximo de lecturas dividiendo un segundo entre 112µs la velocidad del convertidor es 8.92KHz.

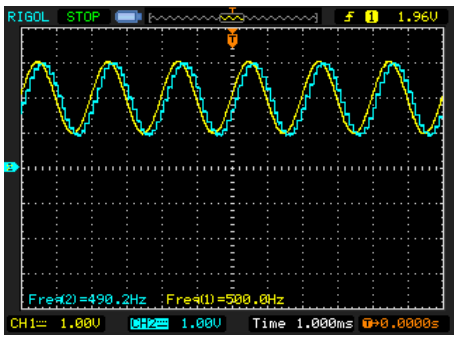


Figura 3.12 Señal entrada 500Hz y salida PORTD directo.

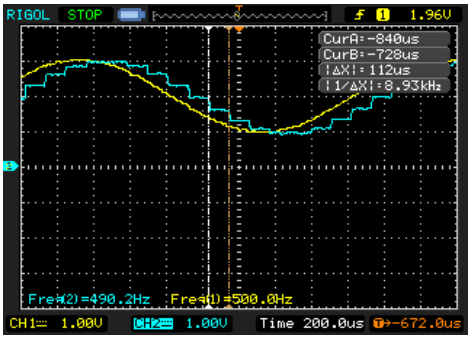


Figura 3.13 Intervalo de tiempo entre cada lectura del ADC.

3.2.2.2Muestreo de la señal mediante interrupciones.

Al igual que el método anterior se utilizó el diagrama de la figura 3.10. Se utilizó una señal senoidal conocida de 1Vpp de 50Hz (con offset de 1V) y se muestreo por el método de interrupciones para desplegarla al puerto D. Se puede ver en la figura 3.14 (y sección de anexos A.2) el código para muestrear por interrupciones una señal y desplegarla al puerto D. Se utiliza una librería llamada "TimerOne" y los comandos **timer1.intialize** (para definir el tiempo que se ejecutará la interrupción) **y timer1.attachInterrupt** (mandar a llamar la interrupción) para realizar las interrupciones cada determinado tiempo según lo deseado. Se configuró el timer con el comando de **timer1.intialize** (tiempo) a 2777; que surge de dividir 1 segundo entre 360 muestras, lo cual nos arroja un valor de 2.7 ms y se debe multiplicar por 1000, debido a que se requiere expresar en microsegundos.

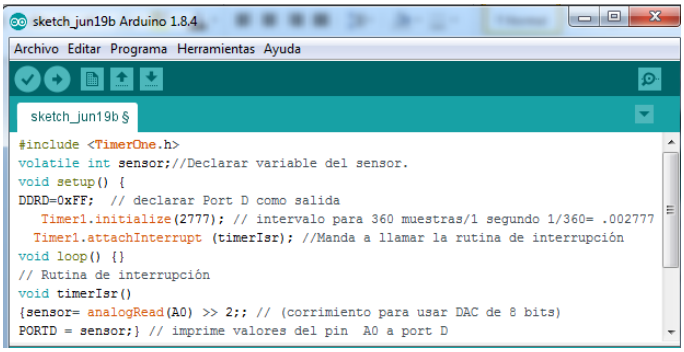


Figura 3.14 Código para leer una señal analógica y desplegarla al puerto D por interrupciones.

La señal de entrada muestreada se desplegó al puerto D (Ver figura 3.15) y se midió el intervalo entre cada lectura siendo de 2.7 ms (ver figura 3.16), mismo que se configuró en el código comprobando así el método de interrupciones

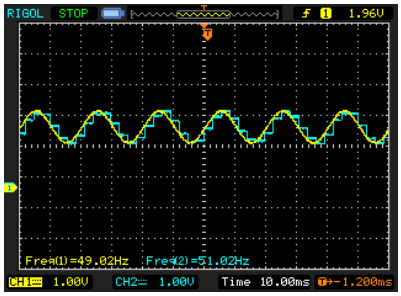


Figura 3.15 Señal muestreada a 360Hz con interrupciones.

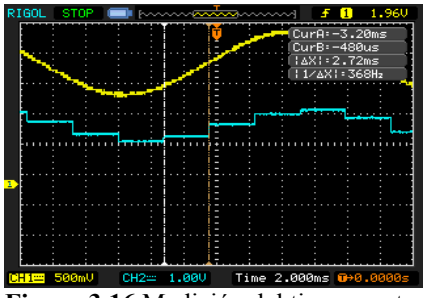


Figura 3.16 Medición del tiempo entre cada lectura del ADC muestreada por el método de interrupciones.

3.2.3 Caracterización y validación de trasmisión de datos con puerto serial Arduino.

Una vez establecido el uso y configuración del convertidor A/D, se utilizó un cable USB para conectar el puerto serial de Arduino al puerto USB del PC. Se cargó el código A.3 de anexos para transmitir vía serial la información. Se utilizó la conexión indicada en la figura 3.17.

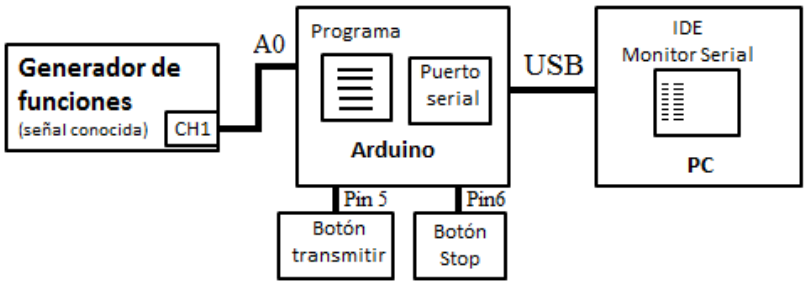


Figura 3.17 Esquema para transmitir una señal al puerto serial con control de 2 botones.

Se realizaron pruebas para saber cuál es la mayor cantidad de datos que nos permite enviar el puerto serial en 1 segundo. Para esto se configuró el comando "**delay**" nulo ya que se trata del número máximo de datos, y se utilizó el comando "**millis**" el cual permite iniciar el conteo de un reloj interno del Arduino, y nos permite conocer el valor del dato leído, y el tiempo en que se realizó la medición, en la figura 3.18 se puede ver que el monitor serial del software IDE de Arduino nos proporciona el dato leído, y el tiempo en segundos.

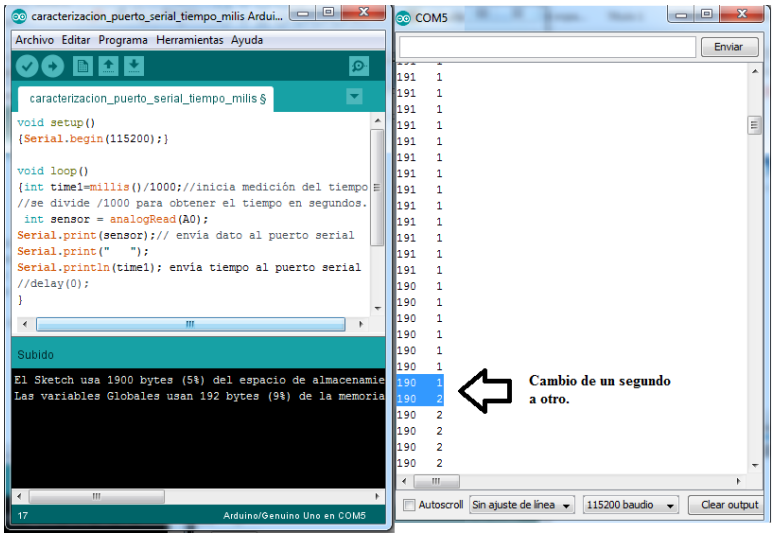


Figura 3.18 Datos transmitidos al puerto serial con comando millis para medición del tiempo.

Para validar el número de datos enviados y recibidos por el puerto serial, se transmitió una señal conocida durante más de 1 minuto y se recuperó la información con el monitor serie. Estos datos se exportaron al software Excel separando el número de datos enviados cada segundo en una columna. Se puede ver en la figura 3.19 que el número promedio de datos transmitidos es de 2084 con un error de hasta 12 muestras.

	1 seg.	2 seg.	3 seg.	4 seg.	5 seg	6 seg.	7 seg.
2075							
2076	0.81 0	0.50 1	0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2077		0.50 1	0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2078		0.50 1	0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2079		0.50 1	0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2080		0.50 1	0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2081		0.50 1	0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2082		0.50 1	0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2083		0.50 1	0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2084		0.50 1	0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2085			0.45 2	0.45 3	0.44 4	0.45 5	0.43 6
2086			0.45 2		0.44 4	0.45 5	0.43 6
2087					0.44 4	0.45 5	0.43 6
2088					0.44 4	0.45 5	
2089					0.44 4	0.45 5	
2090					0.44 4	0.45 5	
2091					0.44 4		
2092					0.44 4		
2093					0.44 4		
2094					0.44 4		
2095					0.44 4		
2096					0.44 4		

Figura 3.19 Datos transmitidos al puerto serial durante 7 segundos por método con "delay" nulo.

Para obtener el máximo número de datos recibidos utilizado el método de interrupciones, se cargó el código A4 de la sección de anexos y se configuró el programa para obtener 2000 muestras por segundo, es decir en el código se ajustó la interrupción a 500µs. Se tomaron datos durante 7 segundos en tiempo real y se exportaron al software Excel los datos en columna, para contar cada segundo cuantos datos se tenían, se puede ver en la figura 3.20 que se tiene un error de 1 muestra por segundo. Se continuó aumentando la frecuencia de muestreo, pero al aumentarla demasiado no daba tiempo a que los datos se enviarán al PC, se aumentó a prueba y error para saber el límite la máxima frecuencia sin presentar perdida de información es 2084 muestras por segundo de esta manera se caracterizó nuestro puerto serial con Arduino para una lectura y envió de información.

1998	1 seg.	2 seg.	3 seg.	4 seg.	5 seg.	6 seg.	7 seg.
1999	0.472	0.423	0.514	0.535	0.526	0.517	0.408
2000		0.423		0.545		0.517	0.408
2001		0.423		0.545		0.517	
2002							
2003			2000 Muestras por segundo.				
2004			500 µs. Intervalo de interrupcion				
2005			±1 Muestras de error				

2082	1 seg.	2 seg.	3 seg.	4 seg.	5 seg.	6 seg.	7 seg.
2083	0.452	0.463	0.452	0.465	0.452	0.457	0.458
2084		0.463		0.465		0.467	0.458
2085		0.453					0.458
2086							
2087			2084 Muestras por segundo.				
2088			480 µs. Intervalo de Interrupción.				
2089			±1 Muestras de error.				

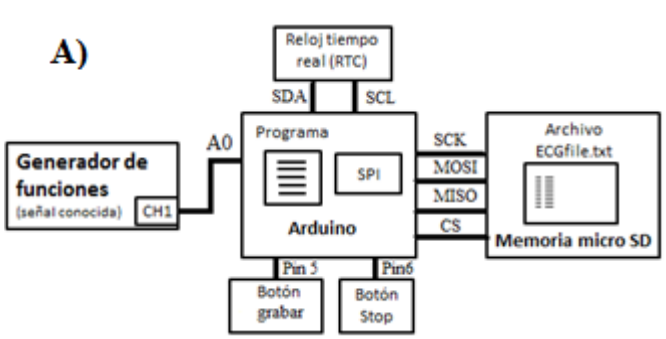
Figura 3.20 Datos transmitidos al puerto serial por método interrupciones.

3.2.4 Caracterización y validación de registro de datos en SD card.

Antes de caracterizar y validar la frecuencia de muestreo se debe realizar este procedimiento sólo en una ocasión, esto para asegurarse que el RTC está configurado en hora y fecha actual, luego de este procedimiento puede durar hasta 10 años con esta configuración o hasta que se reemplace la batería. Primero se cargó el código A5 de la sección de anexos para configurar la hora del RTC de forma manual, o se puede utilizar el código A6 para cargar la hora automáticamente del PC del cual se programó. Para comprobar que la hora cargó correctamente se utilizó el código A7 que imprime la hora y la envía al monitor serial del software IDE.

Una vez que se configuró el RTC y comprobó la hora y fecha se caracterizó el número de datos grabados en 1 segundo en la memoria micro SD. Se utilizó el diagrama de la figura 3.21 A) y se cargó el código A9 para iniciar la lectura de una señal conocida y grabarla con el data logger controlado por los botones para grabar y detener grabación, se graba en un archivo llamado ECGfile.txt Se configuró el código para grabar 500 muestras

por segundo, y debido a que es una grabación fuera de tiempo se utilizó la hora del RTC debido a que no se puede monitorear el flujo de datos en tiempo real. Una vez terminada la grabación en la memoria se conectó a un PC y se extrajo el documento "ECGfile.txt" los datos contenidos en este documento se importaron al software Excel como se ve en la figura 3.21 B) para ver el número de datos enviados por segundo.



B)

495	Voltaje	Fecha	Hora
496	1.52	09/03/2017	15:44:40
497	1.4	09/03/2017	15:44:40
498	1.4	09/03/2017	15:44:40
499	1.4	09/03/2017	15:44:40
500	1.39	09/03/2017	15:44:40
501	1.39	09/03/2017	15:44:41
502	1.39	09/03/2017	15:44:41
503	1.39	09/03/2017	15:44:41
504	1.39	09/03/2017	15:44:41
505	1.39	09/03/2017	15:44:41
506	1.39	09/03/2017	15:44:41

Figura 3.21 A) diagrama de conexión para registro SD. B) Registro de datos en 1 segundo (500 muestras).

Para obtener el número máximo de datos de la grabación en micro SD, se realizaron ajustes a prueba y error en el intervalo de la interrupción para luego leer la cantidad de datos por segundo grabados en la memoria. Inicialmente el intervalo que se utilizó fue 2000 µs y se fue reduciendo hasta obtener mayor número de muestras. El valor del intervalo de interrupción para el máximo número de datos sin presentar perdida de información fue 485 µs, si este intervalo se reduce más el microcontrolador no da tiempo a realizar la escritura en la memoria micro SD, cuando interrumpe el proceso para leer un nuevo valor, y como resultado el archivo"ECGfile.txt" se ve en blanco sin ningún dato grabado. Como se puede ver en la Figura 3.22 el número promedio máximo de muestras para grabación en memoria micro SD fue 2019 muestras.

2017	1 seg.	2 seg.	3 seg.	4 seg.	5 seg.	6 seg.	7 seg.
2018	0.2	2.22	0.02	0.03	0.02	0.457	0.458
2019		0.06		0.03	0		0.458
2020		0.02		0.03			0.458
2021							
2022			2019 Muestras por segundo.				
2023			485 µs. Intervalo de Interrupción.				
2024			±1 Muestras de error.				

Figura 3.22 Número máximo de datos grabados en SD en 1 segundo.

3.2.5 Caracterización y validación de trasmisión de datos por bluetooth.

Para caracterizar y validar la transmisión de datos por bluetooth se cargó el código A10 de la sección de anexos para configurar el módulo bluetooth HC-06, y siguiendo el esquema de conexión de la figura 3.23 se configuró el módulo como indicaba el manual [60] con características como nombre: ECGsignal, velocidad de transmisión a 115200 baudios, y contraseña. Una vez configurado el módulo se cargó el programa A12 para leer una señal conocida y transmitirla de forma inalámbrica a nuestro dispositivo con bluetooth. Para validar la transmisión vía bluetooth se utilizó un PC con bluetooth integrado y se realizó el emparejamiento para vincular ambos dispositivos.

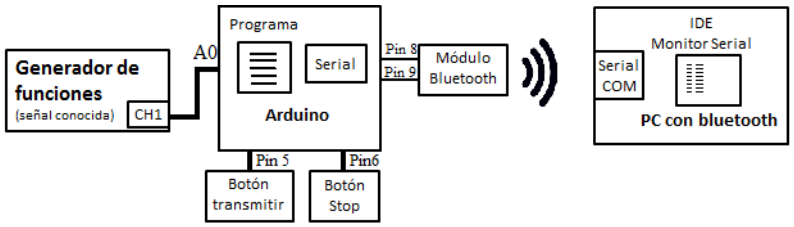


Figura 3.23 Diagrama a bloques de conexión para transmisión inalámbrica a PC.

Se utilizó el software IDE y monitor serial para recibir los datos, es importante mencionar que para recibir adecuadamente la información es necesario indicar el puerto COM que corresponde al puerto bluetooth integrado del PC, debido a que se tienen puertos COM que corresponde a los puertos USB del PC. Una vez recibidos los datos en el software IDE se importaron a Excel conociendo así el número máximo de datos enviados en 1 segundo siendo 909 como se ve en la figura 3.24. Este método transmite menos datos que los otros debido a que podemos pensar que esto se puede deber a que es un método inalámbrico, la versión del bluetooth del PC o dispositivo que recibe la señal, pero se tiene incertidumbre sobre esto. Aun así esta frecuencia de muestreo es suficiente para la aplicación de transmisión de una señal ECG por bluetooth en tiempo real.

907	1 seg.	2 seg.	3 seg.	4 seg.	5 seg.	6 seg.
908	6734	6745	6846	6747	6858	6759
909		6765		6737	6818	6739
910		6805		6747		
911						
912						
913						
914						
915			909 Muestras por segundo.			
916			1100 µs Intervalo de Interrupción.			
917			±1 Muestra de Error.			

Figura 3.24 Datos transmitidos inalámbricamente por módulo bluetooth.

3.2.6 Trasmisión de datos con puerto serial y uso de LABVIEW.

Una alternativa a la herramienta serial plotter para visualizar la señal transmitida por el puerto serie, es mediante el software LABVIEW debido a que es muy utilizado en la visualización y procesamiento de señales, se busca aplicar este estándar para crear una interfaz de visualización. Se utilizó el código A4, para transmisión al puerto serial (se quitó las diagonales "//" de la última línea para imprimir la letra "v" después de transmitir un dato). El software de LABVIEW funciona con programación por bloques (Ver ambiente de programación en figura 3.25) donde la etapa receptora de datos se configuró a una tasa de transferencia de 115200 baudios, las muestras son almacenadas en un buffer donde el programa interpreta los datos recibidos, se configuró para que al recibir la letra "v" identifica como un nuevo valor a graficar, y este queda grabado para graficarse en el último bloque. Cada uno de estos pasos se consultaron en el manual de LABVIEW [61] ya que cada elemento del bloque tiene sus diferentes aplicaciones en este caso se utilizaron para la recepción de datos del puerto serial.

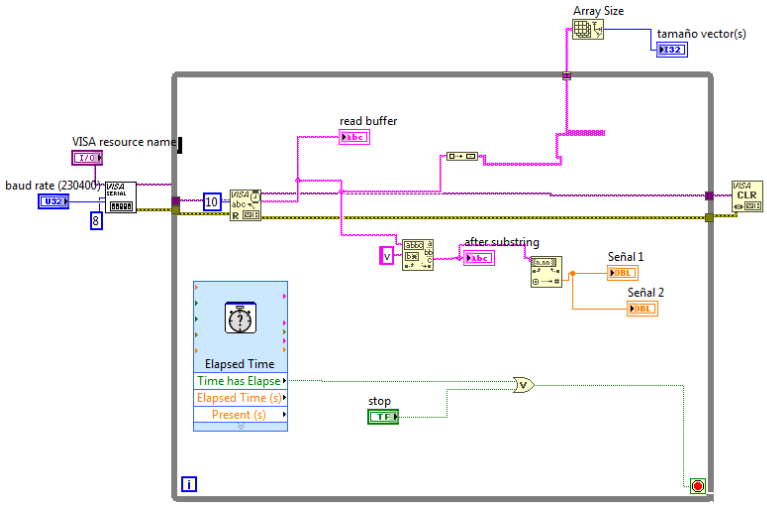


Figura 3.25 Diagrama de programación LABVIEW para recibir y graficar datos del puerto serial.

Para la interfaz de visualización y graficar la información, se puede ver en la figura 3.26 es necesario configurar la casilla "VISA *resource name*" donde se debe seleccionar el puerto COM serial del cual se estén recibiendo los datos del Arduino y así poder ver la gráfica, es importante observar que en el código se configure la tasa de baudios a 115200 ya que esta debe coincidir con la del programa en LABVIEW y se obtiene la gráfica en tiempo real.

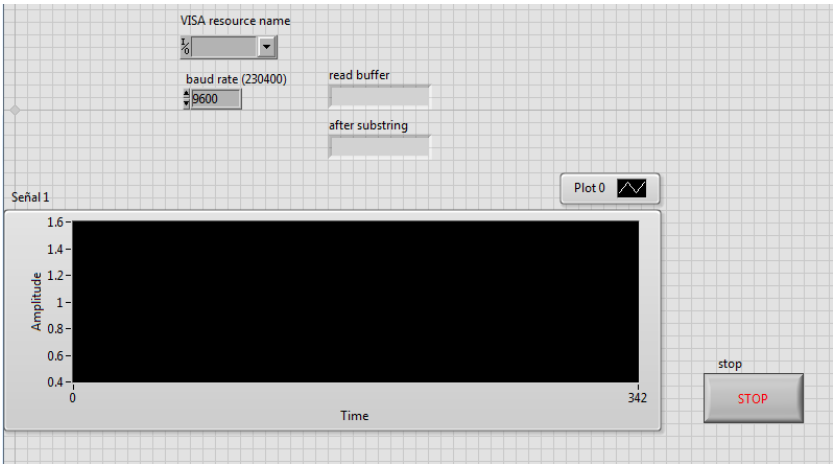


Figura 3.26 Interfaz programa LABVIEW para graficar y recibir datos del puerto serial.

Una vez realizados estos pasos para configurar el puerto COM y la tasa de baudios, se utilizó la señal ideal de ECG creada en la sección 3.6.3 para visualizar la señal en tiempo real en el software LABVIEW (Ver Figura 3.27).

Figura 3.27 Señal de ECG ideal transmitida por puerto serie visualizada en software LABVIEW.

3.3 Modo de operación transmisión en serie.

3.3.1 Transmisión de señal de ECG ideal por puerto serial y gráfica en Serial Plotter de Arduino.

Una vez caracterizada la transmisión serial, siguiendo la configuración de la figura 3.17 se cargó el código A4 para leer, transmitir, y graficar una señal de ECG ideal muestreada a una frecuencia de 360Hz (esta señal ideal es creada como lo indica la sección 3.6.3 de este documento). Se eligió esta frecuencia debido a que se busca posteriormente evaluar la capacidad del sistema para aplicación de filtros digitales los cuales están diseñados para esta frecuencia de muestreo. Se transmitió la señal al puerto serial y a diferencia de utilizar el monitor serial se graficó con la herramienta Serial Plotter de Arduino del software IDE (Figura 3.28).

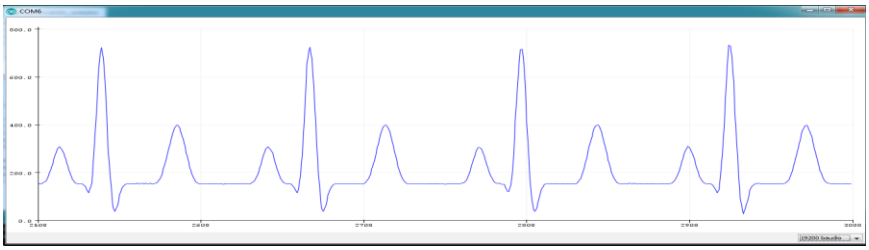


Figura 3.28 Señal de ECG ideal transmitida al puerto serial por Arduino.

3.4 Modo de operación grabación en micro SD data logger.

3.4.1 Códigos data logger.

3.4.1.1 Código de Arduino para data logger con hora y fecha con comando delay.

Se cargó el código A8 para realizar un registro ECG con hora y fecha este código sigue el diagrama de flujo en la figura 3.29 primero se deben definir los pines y librerías que utilizarán, se deben declarar las variables, inicializar la memoria SD card, abrir o crear un archivo "ECGfile.txt". El paso siguiente es realizar la lectura del ADC con **delay** nulo, después mediante el bus I2C se extrae la hora y fecha del RTC, y se imprimen los valores en la memoria SD mediante el protocolo SPI para finalmente cerrar el archivo, obteniendo un registro con hora y fecha (Ver tabla 3.3).

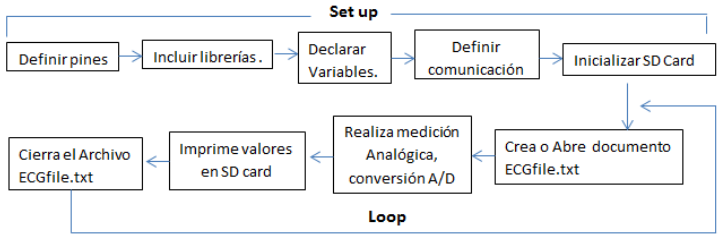


Figura 3.29 Diagrama de flujo del código de Arduino para data logger con hora y fecha con delay nulo.

Tabla 3.3 Registro de muestra, voltaje, fecha y hora para ECG.

# Muestra	Dato	Fecha	Hora
1	2.16	03/02/2017	09:31:40
2	1.93	03/02/2017	09:31:40
3	1.79	03/02/2017	09:31:40
4	1.69	03/02/2017	09:31:40
5	1.64	03/02/2017	09:31:40
6	1.57	03/02/2017	09:31:40
7	1.56	03/02/2017	09:31:40
8	1.52	03/02/2017	09:31:40
9	1.58	03/02/2017	09:31:40
10	1.57	03/02/2017	09:31:40

3.4.1.2 Código de Arduino para data logger con interrupciones.

En el código A9 se han implementado algunas recomendaciones (Ver sección 2.12.3 de este documento) para grabación en un data logger, este código sigue la estructura de la figura 3.30. Primero en la función de set up se definen los pines, incluyen librerías, declaran variables a utilizar, definir la comunicación, inicializar la memoria SD card y definir el intervalo de interrupción. En este código se ha agregado un botón para inicializar la grabación, y otro para finalizarla. Después de definir el tiempo de la interrupción se espera a que se presione el botón para iniciar el muestreo o grabación, si este no se presiona se sigue en espera en un **ciclo loop,** y al presionarse el botón se crea o abre el documento "ECGfile.txt" y se manda a llamar la interrupción en una función llamada "grabar", hasta que se presione el botón para finalizar la interrupción, si este botón no se presiona se seguirá ejecutando la interrupción, y si el botón se presiona se cierra el archivo y termina la grabación en SD.

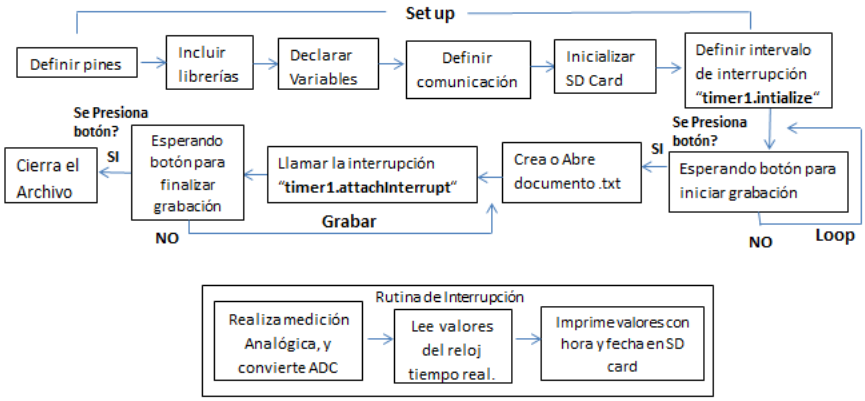


Figura 3.30 Diagrama de flujo del código de Arduino para data logger con hora y fecha utilizando interrupciones.

3.4.1.3 Almacenamiento de un registro ECG en SD card.

Para conocer el espacio de memoria que consume el registro de ECG, se realizaron pruebas con registros de duración de 1, 12, 24 horas. Se utilizaron frecuencias de muestreo de 360Hz, y 500Hz para conocer el tamaño de los archivos "ECGfile.txt" según el número de muestras. Se alternó también el tipo de contenido grabado en el registro con dato, hora, y fecha y registros con dato y hora únicamente. Se utilizó una señal de ECG ideal creada en MATLAB (como se explica en la sección 3.6.3). Se generó una tabla con los resultados obtenidos y se muestra en tal sección de resultados 4.6.2.

3.4.2 Obtención de los datos y método para graficar.

Como se ha mencionado antes los datos son grabados en la SD en un archivo de texto, para graficar estos datos y poder observar la señal de ECG es necesario conectar la memoria a la PC que visualizará la información. Un ejemplo de software compatible con el formato .txt es Excel y para graficar la información se deben seguir los siguientes pasos: se presionó la pestaña abrir archivo, localizó el archivo en la SD card, y se seleccionó el archivo ECGfile.txt con el botón abrir. Una vez importados los datos del archivo ECGfile.txt al software Excel al graficar la primera columna que corresponde a la señal leída se puede obtener una gráfica como la de la figura 3.31.

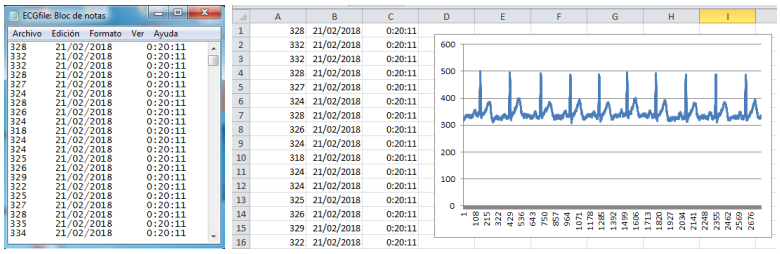


Figura 3.31 Registro de datos del data logger y señal ECG graficada en Excel.

3.4.3 Método para graficar en software THEREMINO ECG.

THEREMINO es una plataforma de código abierto, que busca la implementación y desarrollo de software dedicado a la conexión de sensores y dispositivos inteligentes con el fin de medir y estudiar señales. Esta plataforma tiene aplicaciones dedicadas al entretenimiento, herramientas para la enseñanza e investigación científica. El software que se utilizó para graficar la señal ECG está en la categoría de biometría, es necesario descargar la aplicación THEREMINO ECG e instalar en su PC [62]. Para graficar la señal de ECG es necesario extraer la columna de datos, del archivo "ECGfile" generado en la SD card sin la hora y fecha, y guardar esta columna de datos en un nuevo archivo de texto. Como puede observarse en la figura 3.32 se tiene un botón para importar archivos de texto *Load data,* se y se debe seleccionar el archivo que contiene la columna únicamente de la señal ECG. Este software se utilizó para visualizar registros de larga duración con una frecuencia de muestreo de 500Hz. Y aunque tiene funciones para modificar la velocidad del papel registro, contar latidos por minuto, tomar captura de pantalla y otras funciones de monitoreo, en esta investigación se utilizó solo con fines de visualizar el registro ECG generado por nuestro prototipo.

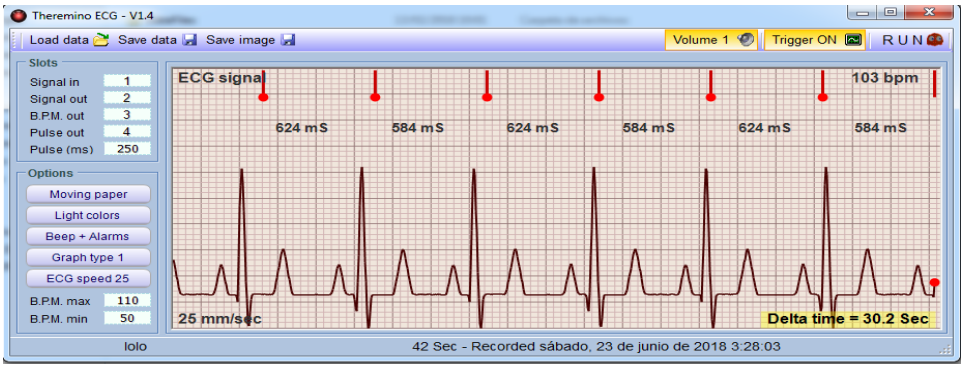


Figura 3.32 Software THEREMINO ECG.

3.4.4 Análisis de un registro ECG por tiempo prolongado.

Para esta prueba se utilizaron 8 registros de ECG grabados como se explica en la sección 3.4.1.3 con duración de 1, y 2 horas, con frecuencias de muestreo de 500Hz y 360Hz. Para esta prueba se utilizaron 2 tipos de señal: en un caso una señal virtual de ECG generada como se indica en la sección 3.6.3 y se realizaron estas mismas pruebas grabando una señal de paciente sin antecedentes de problemas cardíacos en estado de reposo. Para el análisis de las señales se importaron los datos de los archivos "ECGfile".txt generados en el data logger al software Excel y se graficaron secciones de los registros de larga duración hasta revisar todos los datos del registro. Se puede ver en la figura 3.33 A) una sección del registro con la señal ideal, y en la figura 3.33 B) una sección del registro con la señal del paciente. Esta prueba se realizó con el fin de observar el correcto funcionamiento del data logger.

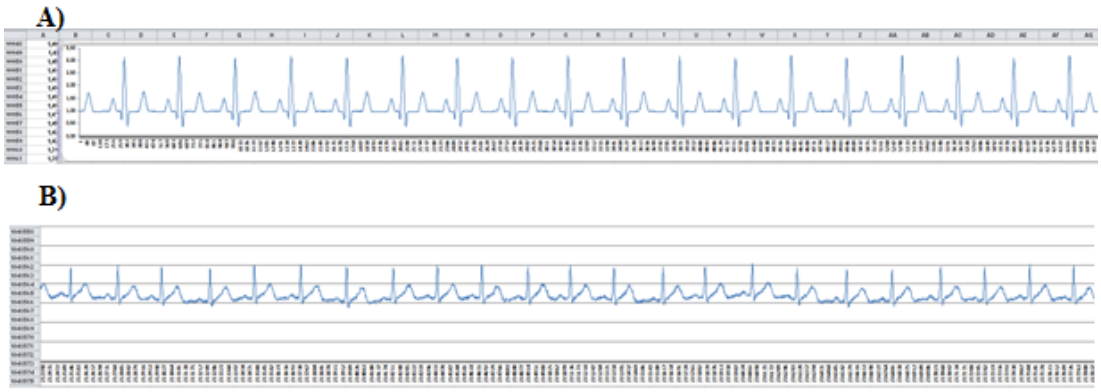


Figura 3.33 A) Análisis de registro ECG por tiempo prolongado grabando una señal virtual en software Excel. B) Análisis de registro ECG por tiempo prolongado grabando una señal de paciente en software Excel.

3.5 Modo de operación transmisión datos por bluetooth.

3.5.1 Configuración módulo bluetooth HC-06

El módulo HC-06 está configurado de fábrica para una velocidad de 9600 baudios, y nombre direccional HC-06. En las pruebas realizadas anteriormente de la sección 3.2.5 se configuró con el código A10, el nombre y contraseña del dispositivo. La conexión hardware utilizada para la programación del bluetooth se puede ver en la figura 3.34.

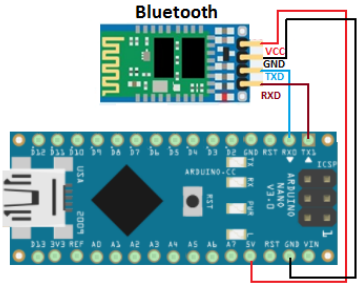


Figura 3.34 Conexiones de Arduino Nano y módulo HC-06 para configuración de dispositivo.

Para realizar la configuración del módulo HC-06 (utilizando el código A10) es necesario declarar 3 variables, el nombre del dispositivo, la contraseña, y la velocidad '8' esta velocidad según la hoja de datos la tasa máxima de datos soportada por el protocolo de comunicación bluetooth en PC y corresponde a 115200 baudios, pero es suficiente para lograr la transferencia de 906 muestras por segundo (ver sección 3.2.5). Para la configuración de este dispositivo se utilizaron comandos AT en el programa, y se configuró nombre "ECGsignal", la velocidad de baudios, y la contraseña "1111", agregando un **delay** de 2 segundos entre cada uno de estos comandos para que se guarde esta configuración (Figura 3.35), una vez configurado este módulo podremos encontrar el dispositivo en nuestro PC, vincularlo, e iniciar la transferencia de datos a través del puerto serial COM.

```
char nombreBT[10]="ECGsignal";
char velocidad= '8';
char pin[5]="1111";

void setup() {
Serial.begin(115200);

Serial.print("AT");
delay(2000);

Serial.print("AT+NAME");
Serial.print(nombreBT);
delay(2000);

Serial.print("AT+BAUD");
Serial.print(velocidad);
delay(2000);

Serial.print("AT+PIN");
Serial.print(pin);
delay(2000);
```

Figura 3.35 Código para configurar módulo HC-06, nombre: ECGsignal, velocidad: '8', y contraseña: 1111.

3.5.2 Transmisión de señal ECG ideal por bluetooth y recibidos en LABVIEW en PC.

Una vez configurado nuestro módulo bluetooth y enlazado nuestro dispositivo al PC que recibirá los datos. Se cargó el código A12 para transmitir y recibir la señal de ECG por bluetooth y se configuró el software LABVIEW seleccionando el puerto COM del puerto serial bluetooth, para visualizar la señal. En la interfaz de LABVIEW se configuró la tasa de baudios a 115200, y se obtuvo la señal de la figura 3.36.

Figura 3.36 Señal de ECG ideal transmitida por módulo bluetooth en LABVIEW.

3.5.3 Transmisión de señal ECG por bluetooth a celular con sistema Android con App "Bluetooth Graphics Arduino".

Una vez configurado nuestro dispositivo y cargado el código A11, para visualizar la señal en un teléfono inteligente con software Android se enlazó nuestro ECG portátil por conexión bluetooth, y se descargó en el teléfono aplicación "Bluetooth Graphics Arduino", esta aplicación nos permite graficar hasta 3 canales de datos recibidos por nuestro dispositivo. Para enlazar el teléfono con nuestro dispositivo se debe encender el bluetooth del teléfono inteligente, y buscar nuestro ECG portátil de nombre "ECGsignal" e ingresar la contraseña "1111" (Figura 3.37)

Figura 3.37 Enlace Bluetooth entre Arduino y celular con sistema Android.

Una vez enlazado nuestro dispositivo podemos ver el estado "conectado" (ver Figura 3.38) y se utilizó la señal de ECG ideal de la sección 3.6.3 y se transmitió en tiempo real.

Figura 3.38 Captura de pantalla de señal ECG recibida en celular con sistema Android.

3.6 Aplicación de filtros digitales bajo la plataforma Arduino.

3.6.1 Aplicación de filtros utilizando Arduino.

Un uso importante de la plataforma Arduino que se evalúo, es la aplicación de filtros digitales en tiempo real mediante ecuaciones de diferencias. Las señales de entrada para esta prueba fueron creadas en el software MATLAB y se agregaron diferentes tipos de ruidos para ver la respuesta de los filtros. Una vez generadas las señales se copiaron las columnas con los datos del software MATLAB, y se introdujeron en un generador de funciones modelo AFG3021B Tektronix, se utilizó el software ArbExpress para generar un archivo con formato .csv para reproducir la señal en el generador de funciones.

Se utilizó el diagrama de la figura 3.39 para generar la señal de entrada con ruido, se programó la ecuación de diferencias del filtro, se envió al puerto D, y se recuperó la señal de forma analógica por un convertidor digital-analógico (DAC) de 8 bits, para poder visualizar la señal filtrada en el osciloscopio.

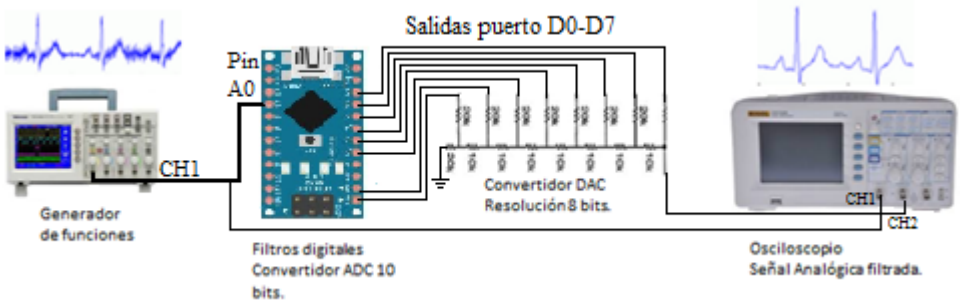


Figura 3.39 Diagrama de conexión para conversión de señal analógica y aplicación de filtros digitales con Arduino Nano.

3.6.2 Elementos para filtros digitales: buffer circular, y tiempo de operación de la ecuación.

Para la aplicación de filtros digitales con ecuaciones de diferencia (ver estructura del código en figura 3.40), se consideraron algunos aspectos como la utilización de un buffer circular, para guardar valores anteriores de la variable leída, y aplicar la ecuación de diferencia con el valor actual según lo requiera la ecuación. Se utilizó el código A13 de la sección de anexos para realizar el desplazamiento de una lectura en un buffer circular de 7 valores, debido a que la ecuación del filtro $y[n] = x[n] - x[n-6]$ utiliza 7 lecturas en el buffer; ya que es el número máximo de datos de memoria que se requirió para la aplicación de filtros en este trabajo (ver figura 3.41)

También se midió el tiempo que tarda el microcontrolador en realizar la operación de una ecuación de diferencias y la actualización de los valores en el buffer para filtrar una señal, para esto se utilizaron los códigos A14-A17 de la sección de anexos. Las operaciones que se pueden presentar en una ecuación de diferencias, son multiplicar, dividir, sumar y restar según el filtro que se desea implementar. El tiempo que el microcontrolador tarda en efectuar las operaciones de los filtros va desde 116 microsegundos en el caso del filtro más sencillo #1, y 252 microsegundos para el filtro más complejo #4 lo cual no representa un problema que pueda retrasar la información en el filtrado de la señal para un monitor ECG.

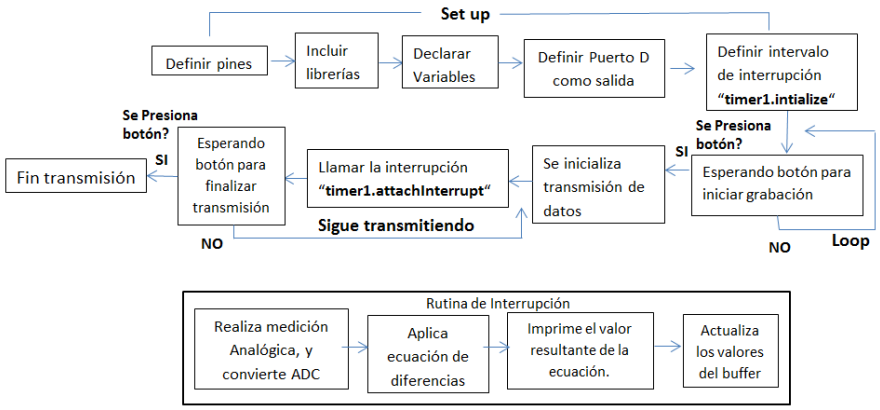


Figura 3.40 A) Diagrama del código para aplicación de filtros digitales.

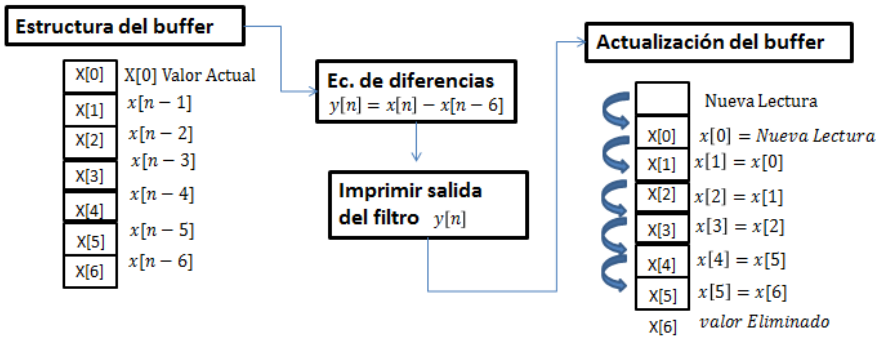


Figura 3.41 B) Estructura del buffer circular, ecuación de diferencias, y actualización de buffer.

3.6.3 Generación de las señales de prueba.

Para crear las señales de entrada para la simulación de filtros se utilizó el software MATLAB, se descargó un complemento del libro "*Analog and Digital Signal Processing*", de A. Ambardar [65] llamado "ADSP *toolbox*" [63] la cual contiene material de apoyo y funciones, para ayudar a la resolución de los ejercicios del libro. Se descargó la carpeta "ADSP *toolbox*" al directorio principal de MATLAB y se ejecutó la función ecgsim (N, m) donde se definieron 2 coeficientes, el valor N nos define el número de muestras (multiplicado por 100), y el valor m nos define el suavizado de la señal. Con esto se creó una señal ideal de 300 muestras, y con el comando "*resample*" se re-muestreó a 360Hz para reproducirla en un generador de funciones a 1 Hz y así obtener 360 muestras por segundo, esto debido a que los filtros de este trabajo eliminan la frecuencia de 60Hz, y sus armónicos a 360Hz. Una de las ventajas de trabajar con 360Hz es que las ecuaciones de diferencias en comparación con otras frecuencias de trabajo son más cortas y sencillas lo que nos ayuda a

reducir la carga del microcontrolador. Una vez obtenida la columna de datos de la señal ECG ideal se importó al software Excel o ArbExpress y generar un archivo con formato .csv para reproducir en el generador de funciones.

Luego de obtener la señal de ECG ideal a 360 muestras se utilizó el código A27 de la sección de anexos y se le agregó un ruido de 60 Hertz con una amplitud igual a 0.25V y dos de sus armónicos, el primero de 120 Hertz con una amplitud igual a 0.1V y el segundo de 180 Hertz con una amplitud de 0.08V. Posteriormente se le añadió una señal sinodal de 0.1 Hertz simulando el movimiento basal de la señal. (Ver fig. 3.42).

Al final se obtuvieron 4 señales de prueba:

1) Señal de ECG ideal concatenada muestreada a 360.
2) Señal de ECG concatenada con ruido de 60 Hertz.
3) Señal de ECG concatenada con ruido de 60 Hertz y sus armónicos.
4) Señal de ECG concatenada con ruido de 60 Hertz, sus armónicos y con una senoidal de 0.1 Hertz simulando el movimiento de la línea basal.

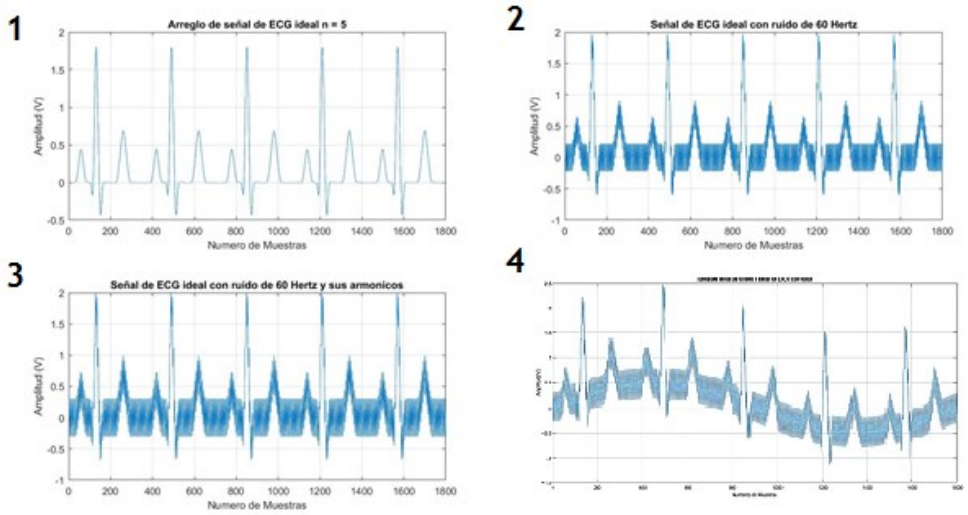
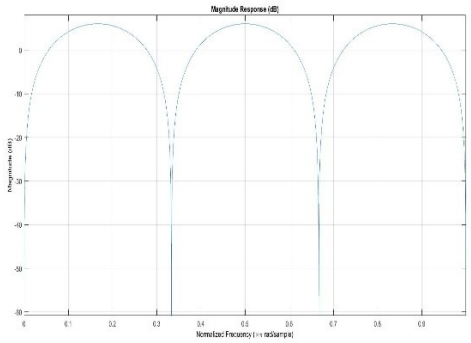
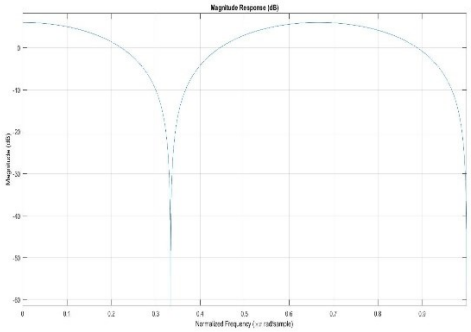
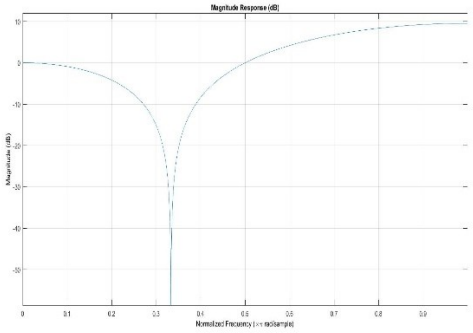
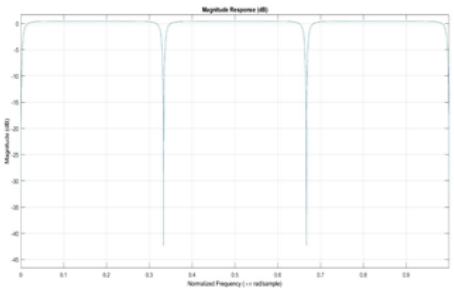


Figura 3.42 Señales ECG de prueba, creadas en MATLAB.

3.6.4 Filtros y ecuaciones de diferencias.

Los filtros que se utilizaron para evaluar la plataforma de Arduino se pueden ver en la tabla 3.4, con sus respectivas ecuaciones de diferencias, y respuestas en magnitud que se obtuvieron al ejecutar código A27 en el software MATLAB para simular la respuesta en magnitud de estos filtros. En la sección de anexos los códigos A18, A19, A20, y A21 corresponden a los filtros de la tabla 3.4.

Tabla 3.4 Filtro digitales, ecuación de diferencias y respuesta de magnitud (MATLAB).

No. de filtro	Ecuación de diferencias	Respuesta en magnitud
1	Filtro peine para eliminar ruido de 60, 120,180Hz, y offset [64]. $y[n] = x[n] - x[n-6]$	
2	Filtro peine que elimina el ruido de 60 Hertz y su armónico de 180 Hertz [65]. $y[n] = x[n] + x[n-3]$	
3	Filtro que elimina solo ruido de 60 Hertz [64]. $y[n] = x[n] - ax[n-1] + bx[n-2]\ ]$ $donde$: $a = -1$ $b = 1$	
4	Filtro Notch periódico para eliminar ruido de 60, 120,180Hz, y offset [65,66]. $y[n] = x[n] - x[n-6] + \alpha y[n-6]]$ Donde α = 0.9	

3.6.5 Muestreo señal ECG ideal a 360Hz.

Para visualizar la señal en osciloscopio como método de prueba, se utilizó el código A2 para muestrear por interrupciones, siguiendo el diagrama de la figura 3.39 y se cargó un archivo con la señal de prueba #1 (señal ECG ideal creada en MATLAB) al generador de funciones. (Ver figura 3.43).

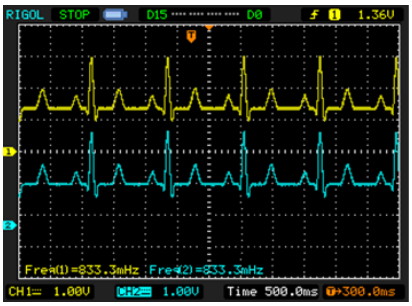


Figura 3.43 Señal entrada ECG ideal (canal 1) y señal muestreada a 360Hz (canal 2).

3.6.6 Respuesta de filtro digital.

Para observar la respuesta de un filtro digital se cargó el código A18 de la sección de anexos, y se utilizó la señal de prueba #4 (señal de ECG con 4 ruidos 60Hz, 120Hz, 180Hz, y un offset de baja frecuencia de 0.1Hz) y se puede ver en la señal de salida en la figura 3.44 A) la señal azul corresponde a la salida simulada en MATLAB en la misma figura B). Se puede apreciar una distorsión en la señal de salida debido ya que la respuesta del filtro se representa en magnitud, y fase. Y al presentarse una respuesta no lineal en la fase, esto afecta directamente la respuesta del filtro. Para observar la respuesta de todos los filtros ver la sección de resultados.

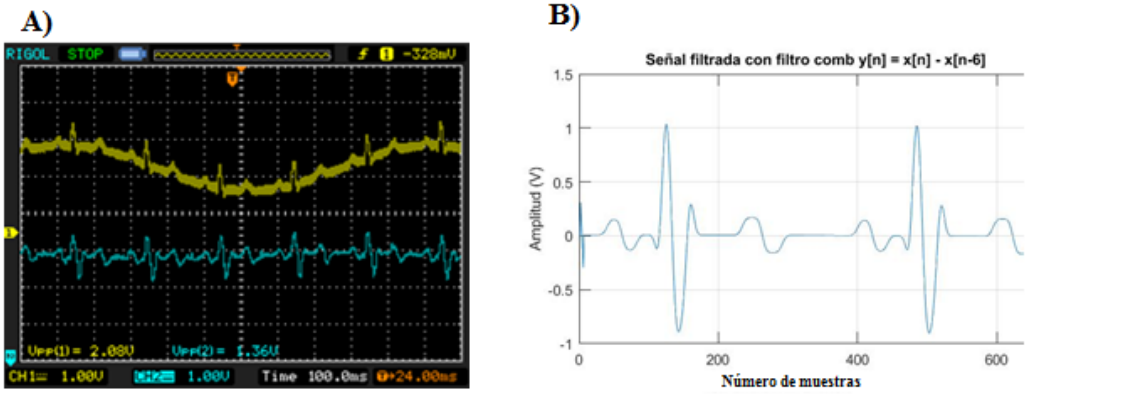


Figura 3.44 A) Señal de prueba #4 en la entrada del sistema (amarilla), y Señal de salida (Azul) respuesta del filtro #1. B) Respuesta de filtro #1 simulada en MATBLAB.

3.6.7 Filtro para eliminar ruido de 60Hz con visualización por transmisión serial.

3.6.7.1 Filtro para eliminar ruido de 60Hz con visualización en Serial Plotter.

Se utilizó el código A22 y la señal de prueba #2 (ver figura 3.45 A) con una señal ECG y ruido de 60Hz, siguiendo el diagrama de conexión de la figura 3.17 para ver la respuesta del filtro en el modo de transmisión por cable Serial. Se puede ver en la figura 3.45 B) que la componente de ruido de 60Hz se ha filtrado.

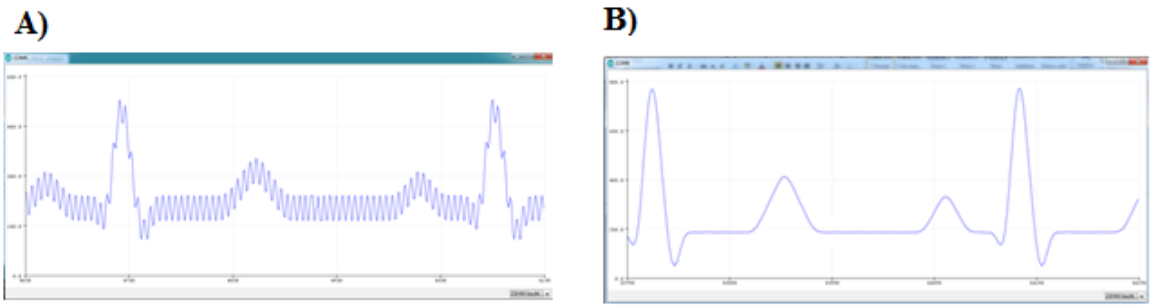


Figura 3.45 A) Señal de ECG con ruido 60Hz transmitida al puerto Serial B) Señal ECG filtrada en software Arduino Plotter.

3.6.7.2 Filtro para eliminar ruido de 60Hz con visualización en LABVIEW.

Se utilizó el código A22 y la señal de prueba #2 (ver figura 3.46 A) con una señal ECG y ruido de 60Hz, siguiendo el diagrama de conexión de la figura 3.17 para ver la respuesta del filtro en el modo de transmisión por cable Serial con el software LABVIEW. Se puede ver en la figura 3.46 B) que la componente de ruido de 60Hz se ha filtrado.

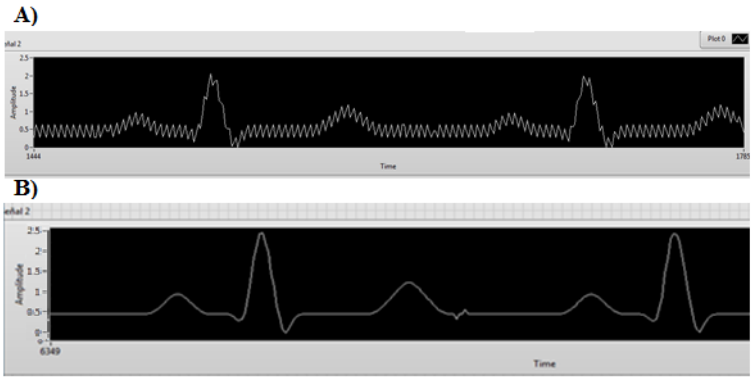


Figura 3.46 A) Señal de ECG con ruido 60Hz. B) Señal filtrada. Transmitidas por puerto serie visualizadas en software LABVIEW.

3.6.8 Filtro para eliminar ruido de 60Hz con visualización en micro SD card.

Se utilizó el código A23 y la señal de prueba #2 (ver figura 3.47 A) con una señal ECG y ruido de 60Hz, siguiendo el diagrama de conexión de la figura 3.21 para ver la respuesta del filtro en grabación micro SD card y se utilizó el software Excel para graficar

los datos. Se puede ver en la figura 3.47 B) que la componente de ruido de 60Hz se ha filtrado.

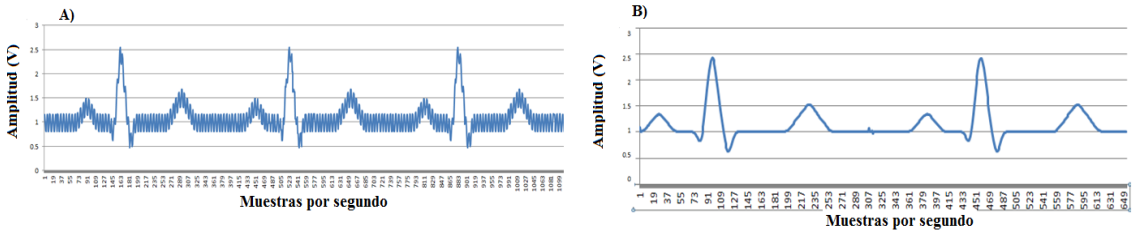


Figura 3.47 A) Señal de ECG con ruido de 60Hz. B) Señal ECG filtrada. Grabadas en memoria SD y graficada en software Excel.

3.6.9 Filtro para eliminar ruido de 60Hz con visualización vía bluetooth celular.

Se utilizó el código A24 y la señal de prueba #2 (ver figura 3.48 A) con una señal ECG y ruido de 60Hz, siguiendo el diagrama de conexión de la figura 3.23 para ver la respuesta del filtro en el modo transmisión vía bluetooth. Para esta prueba se utilizó un celular con sistema operativo Android, se descargó la aplicación de Arduino Graphics y se graficó la respuesta del filtro sin la componente de ruido de 60Hz (ver figura 3.48 B).

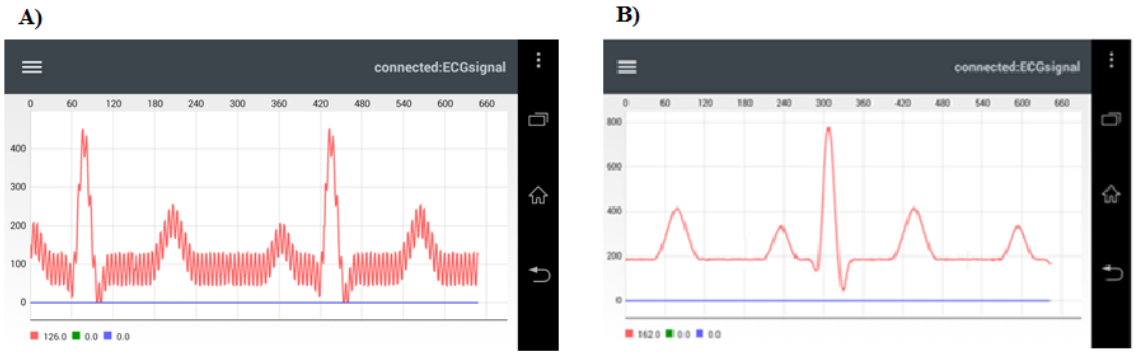


Figura 3.48 Captura de pantalla de celular con sistema Android A) Señal ECG con ruido de 60Hz. B) Señal ECG filtrada.

3.6.10 Filtro para eliminar ruido de 60Hz con visualización vía bluetooth PC.

Se utilizó el código A25 y la señal de prueba #2 (ver figura 3.49 A) con una señal ECG y ruido de 60Hz, siguiendo el diagrama de conexión de la figura 3.23 para ver la respuesta del filtro en el modo transmisión vía bluetooth a PC. Para esta prueba se utilizó un una laptop con bluetooth integrado, y el software LABVIEW para visualizar la respuesta del filtro. Se puede ver en la figura 3.49 B) la respuesta del filtro sin la componente de ruido de 60Hz.

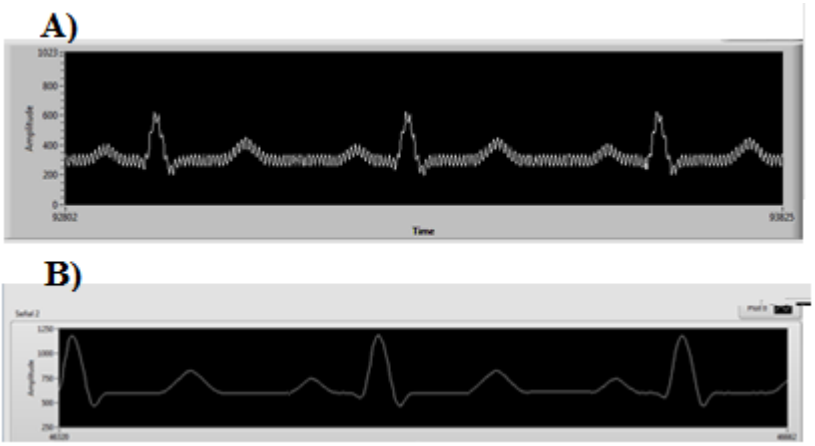


Figura 3.49 A) Señal de ECG con ruido 60Hz. B) Señal de ECG filtrada. Transmitidas por módulo bluetooth, y visualizadas en software LABVIEW.

3.7 Circuito ECG con filtros analógicos para aplicaciones de usuario.

Para el desarrollo de un ECG portátil para monitoreo de uso personal es necesario considerar que el usuario o desarrollador pueden tener necesidades según la aplicación para la cual se quiere utilizar el equipo. Se siguió el diagrama de la figura 3.50 y se evaluaron filtros hardware que limitan el ancho de banda en la etapa de adquisición de la señal, algunas de estas aplicaciones propuestas por la hoja de datos de Analog Devices son:

- Monitor cardíaco (en manos).
- Medición cerca del corazón (pecho).
- Monitoreo durante el ejercicio.
- Circuito EVAL-AD8232

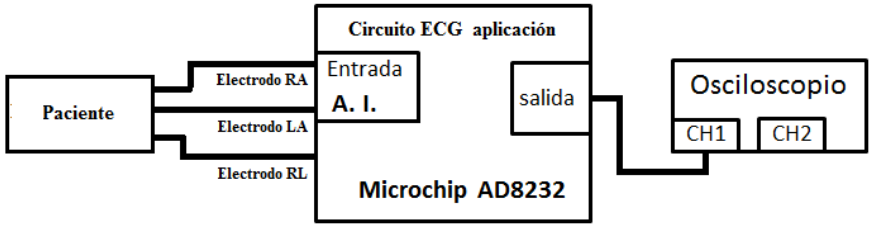


Figura 3.50 Diagrama de conexión para adquisición y visualización de una señal ECG con paciente.

Se implementaron los filtros de la tabla 3.5 y se registraron sus señales.

Tabla 3.5 Aplicaciones con filtros hardware con el chip AD8232.

Aplicación para usuario.	**Rango de frecuencia**	**Ganancia**
Monitor cardíaco (en manos).	0.5-40Hz	1100
Medición cerca del corazón (pecho).	7Hz-N/A	100
Monitoreo durante el ejercicio.	7-19Hz	1100
Circuito EVAL-AD8232	7.2-25Hz	1100

3.7.1 Monitor cardíaco.

Para obtener una forma de onda de ECG con una distorsión mínima, el AD8232 se configuró con un filtro de paso alto de dos polos de 0.5 Hz seguido de un filtro de paso bajo de dos polos y 40 Hz (ver Figura 3.51). Un tercer electrodo es impulsado para un rechazo óptimo en modo común. Además del filtrado de 40 Hz, la etapa del amplificador operacional está configurada para una ganancia de 11, lo que resulta en una ganancia total del sistema de 1100.

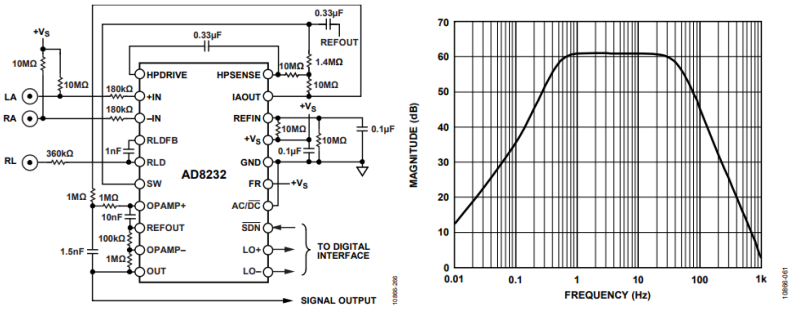


Figura 3.51 Circuito ECG con aplicación de monitor cardíaco y respuesta en frecuencia.

La ecuación para calcular la frecuencia de corte es la siguiente (ec. 6):

$$f_C = \frac{10}{2\pi\sqrt{R1\ C1\ R2\ C2}} \tag{6}$$

Ec. 6 Ecuación para calcular frecuencia de corte para el arreglo de filtro pasa altas del chip AD8232.

Aplicando los valores del circuito a la ecuación 6, R1=R2=10MΩ, y C1=C2=0.33µF la frecuencia de corte del filtro pasa altas es de 0.48Hz y para el filtro pasa bajas se utilizó la ecuación 7.

$$f_C = 1/(2\pi\sqrt{(R1\ C1\ R2\ C2)}) \tag{7}$$

Ec. 7 Ecuación para calcular frecuencia de corte para el arreglo de filtro pasa altas del chip AD8232.

Luego de sustituir los valores R1=R2=1MΩ, C1=1.5nF, y C2=10nF. La frecuencia de corte del filtro pasa bajas es 41.09Hz. Con una ganancia fija del chip de 100, y un arreglo de resistencias R16=1MΩ, y R17=100kΩ, aplicando la ecuación 8 se obtiene una ganancia total de 1100.

$$Gain = 1 + \frac{R16}{R17} \tag{8}$$

Ec. 8 Ecuación para calcular frecuencia de corte para el arreglo de filtro pasa altas del chip AD8232.

3.7.2 Medición cerca del corazón (pecho)

Un filtro de paso alto de un polo está configurado a 7 Hz, y no hay filtro de paso bajo. No se utiliza ganancia en el amplificador operacional de salida, lo que reduce el número de resistencias para una ganancia total del sistema de 100. Los terminales de entrada en esta configuración usan dos resistencias de 180 kΩ para proteger al usuario de las condiciones de fallas y dos resistencias de 10 MΩ para polarización de entrada (ver Figura 3.52).

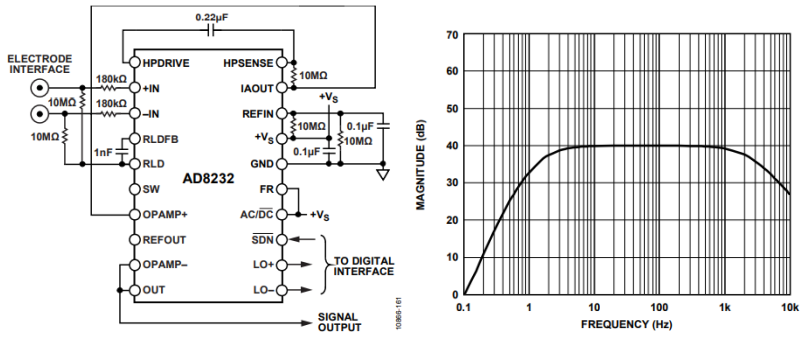


Figura 3.52 Circuito ECG con aplicación cerca del pecho y respuesta en frecuencia.

Para calcular la frecuencia de corte de este filtro pasa altas se aplicó la ec.9 siendo R11=10MΩ, y C7=0.22µF se obtuvo una frecuencia de corte de 7.2 Hz.

$$f_C = \frac{100}{2\pi \times R11 \times C7} \tag{9}$$

Ec. 9 Ecuación para calcular frecuencia de corte para el arreglo de filtro pasa bajas del chip AD8232.

3.7.3 Monitoreo durante el ejercicio.

El circuito en la Figura 3.53 usa un filtro de paso alto de dos polos configurado a 7 Hz. Un filtro de paso bajo de dos polos a 19 Hz sigue los filtros de paso alto para eliminar cualquier otro artefacto y ruido. La etapa de filtro de paso bajo también incluye una ganancia de 11 para una ganancia total de 1100. La naturaleza de su banda estrecha para eliminar el artefacto de movimiento y ruido, distorsiona la onda significativamente esta aplicación se recomienda únicamente para monitoreo del pulso o ritmo cardíaco y no para análisis de ningún tipo.

Para calcular la frecuencia de corte del filtro pasa bajas se empleó la Ec. 10 siendo R9= 100kΩ, y C4=0.22µF, obteniendo una frecuencia de corte de 7.23Hz y para la frecuencia de corte del filtro pasa altas se utilizó la Ec. 7, siendo R1=R2=1MΩ, C1=22nF, y C2=3.3nF obteniendo una frecuencia de corte de 18.67Hz.

$$f_C = \frac{1}{2\pi \times R9 \times C4} \tag{10}$$

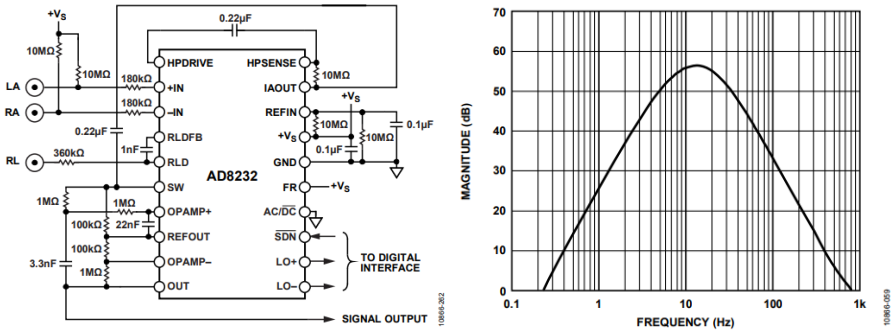


Figura 3.53 Circuito ECG con aplicación de monitoreo durante el ejercicio y respuesta en frecuencia.

3.7.4 Tablero Eval-AD8232 Analog Devices aplicación de monitor ECG.

El circuito de la placa de evaluación AD8232-EVALZ (ver figura 3.54) está diseñada para aplicaciones que involucran tres electrodos conectados a las manos (terminales LA, RA, y RL). El amplificador interno tiene una ganancia fija de 100, y el amplificador operacional está configurado para una ganancia de 11. La ganancia total es de 1100 V / V, lo que limita la señal de entrada diferencial máxima a aproximadamente 2.7 mVpp. Exceder esta amplitud no daña el AD8232; sin embargo, la señal en la salida aparece distorsionada (debido a la saturación del amplificador de instrumentación).

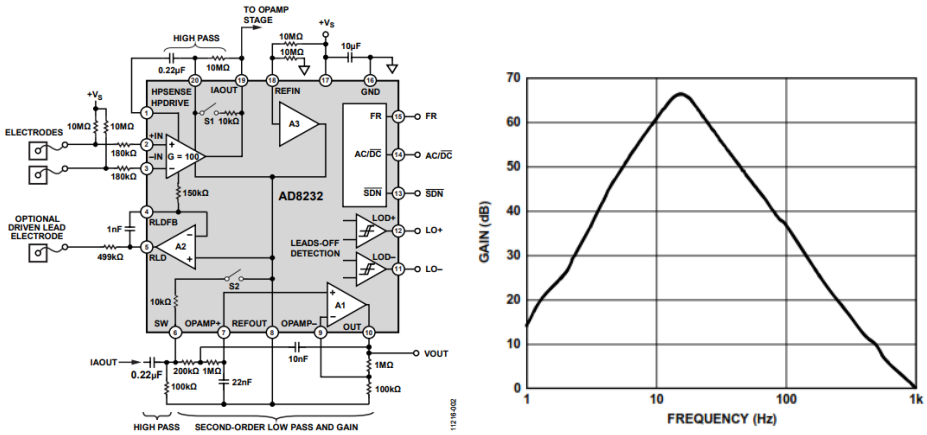


Figura 3.54 Circuito ECG del tablero de evaluación Eval-AD8232 y respuesta en frecuencia.

Para calcular las frecuencias de corte de este filtro, se utilizaron las ecuaciones 7 y 10 respectivamente, siendo los valores de R1=200kΩ, R2=1MΩ, C1=0.022µF, y C2=10nF, obteniendo como resultado una frecuencia de corte de 24Hz, y para la frecuencia del filtro pasa bajas, los valores fueron R9=100kΩ y C4=0.22µF.

3.8 Medición de parámetros de calidad.

3.8.1 Medición del CMRR

Para medir el CMRR se ha utilizado el diagrama de conexión de la figura 3.55, para medir la ganancia diferencial se conectó una de las entradas RA al generador con una señal de entrada de 1mVp y la otra entrada a tierra. El resultado de esta configuración es una señal amplificada 1100 veces, se puede ver en la sección de resultados que la amplitud en estas mediciones está en el rango de 1.1V. Para la conexión de ganancia común se puede ver la figura 3.56 y es necesario conectar en las 2 entradas del amplificador de instrumentación la misma señal, la señal esperada en esta configuración es una señal muy pequeña debido a que el amplificador de instrumentación no debe de amplificar las señales que son iguales. En la sección de resultados puede verse que al medir la ganancia de modo común se obtienen señales muy pequeñas.

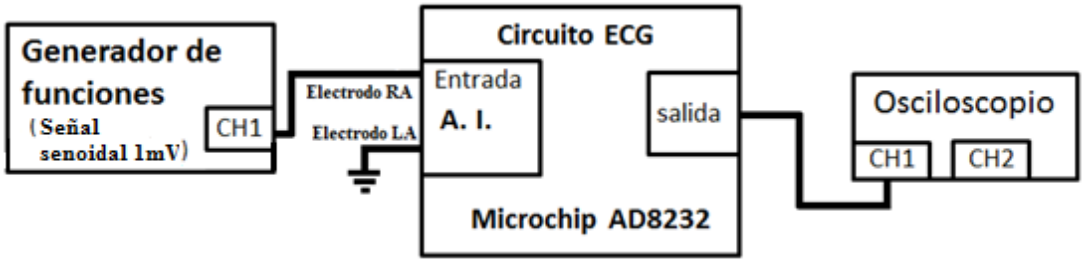


Figura 3.55 Diagrama de conexión para medir la ganancia en modo diferencial.

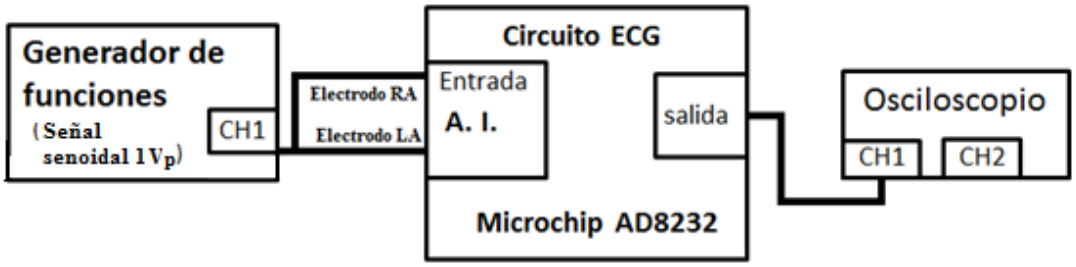


Figura 3.56 Diagrama de conexión para medir la ganancia en modo común.

3.8.2 Medición del consumo de corriente y tiempo máximo de operación.

Una de las características de interés en los sistemas de monitoreo personal portátiles es el máximo tiempo que pueden estar en funcionamiento por ciclo de carga de baterías, siendo estas desechables o recargables. Y para determinar el tiempo máximo de operación del prototipo ECG se midió el consumo de corriente de cada uno de los elementos con ayuda de un amperímetro. Se conectó el circuito como lo indica el diagrama de la figura 3.57 y se midió también el consumo total del circuito mientras ejecutaba cada modo de operación: transmisión en serie, grabación en SD card, transmisión por bluetooth a PC y a teléfono inteligente. Una vez conocida la corriente que consume el prototipo en cada modo de operación, se calculó el tiempo máximo de uso teórico con respecto a la capacidad de baterías para realizar la prueba de duración máxima en cada modo de operación. Este valor se calcula dividiendo el número de miliamperios de consumo, entre la capacidad en miliamperios de la batería obteniendo así el número de horas de operación Los resultados pueden verse en la sección 4.7 de este documento.

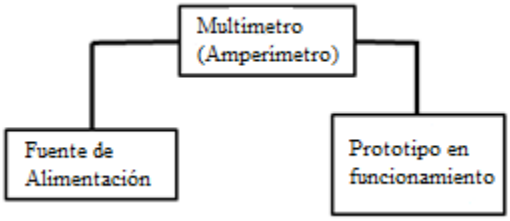


Figura 3.57 Diagrama para medir consumo de corriente del prototipo. Fuente de alimentación baterías LiPO 7.4V, medición de corriente del prototipo en cada modo de operación: transmisión serial, grabación en SD, transmisión por bluetooth.

3.8.3 Medición de la corriente de fuga.

Para el desarrollo de un dispositivo médico uno de los aspectos de seguridad más importantes según la normativa IEC 6060-1 [35] es la corriente de fuga que pueda presentarse durante el funcionamiento hacia el paciente a través de los electrodos. Debido a que es un dispositivo con alimentación interna de baterías se clasifica en un dispositivo de clase IP y tipo BF con una pieza aplicada a paciente (conductor de superficie) el límite máximo permitido de corriente de fuga es 10μA.

Para medir la Corriente de fuga se siguieron las instrucciones de un manual de seguridad hospitalaria de la marca Fluke [44] y se realizó la conexión de la figura 3.58.

Se utilizó un medidor de corrientes de fuga (*leakage current*): modelo UT251A el cual tiene una sensibilidad de 1µA (ver figura 3.59). Los resultados obtenidos de la corriente de fuga de cada modo de operación se pueden ver en la sección 4.8.

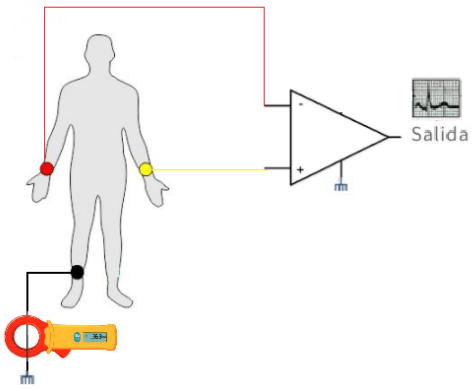


Figura 3.58 Conexión para medir corriente de fuga.

Figura 3.59 Medidor de corrientes de Fuga UT251A. Sensibilidad: 1 µA.

CAPÍTULO 4

RESULTADOS

En este capítulo se presentan los resultados de cada una de las etapas desarrolladas en el capítulo 3 de esta investigación desde la evaluación del microchip AD8232, validación de la señal, los resultados de la simulación con software multisim, digitalización de la señal ECG, aplicación de filtros digitales y hardware, pruebas de almacenamiento con data logger, medición de CMRR, consumo energía, y máxima duración de operación del prototipo por ciclo de carga.

4.1 Evaluación del tablero AD8232

Las señales obtenidas durante la evaluación del tablero Eval-AD8232 (ver sección 3.1.4) utilizaron la configuración de 2 y 3 electrodos de las tablas 3.1 y 3.2. Se puede observar en la figura 4.1 que ambas señales son prácticamente iguales, con la diferencia únicamente de la colocación y número de electrodos esto está pensado para el diseño de sistemas de ECG con diferentes aplicaciones. Se midió la amplitud de cada onda utilizando cursores con el osciloscopio, la onda P fue de 250mV, el complejo QRS alcanzó una amplitud de 1.60V, y la onda T fue de 360mV. Estos valores están dentro del rango del estándar en la forma de onda de ECG normal, [42] y la señal de entrada fue generada con el simulador SimCube.

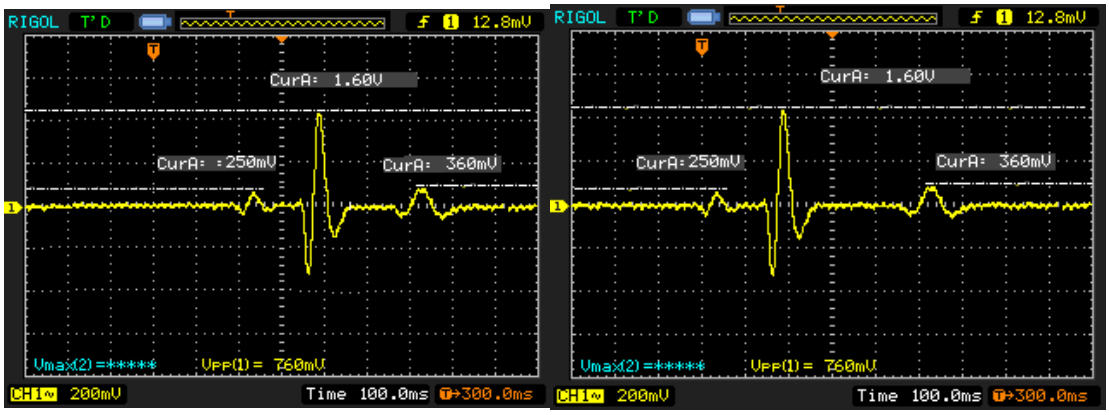


Figura 4.1 Señal ECG con 2 y 3 electrodos utilizando simulador, cursores de amplitud para medir el voltaje de cada onda de la señal. Configuración de osciloscopio modo: DC, 1X, 200mV/Div, 100ms/Div.

Después de comprobar el funcionamiento del tablero de evaluación AD8232 con el simulador SimCube como se indica en la sección 3.1.3, se realizó la de adquisición de la señal utilizando la configuración de 3 electrodos (ver tabla 3.1), con la adquisición de la señal de un paciente electrodos sin antecedentes de problemas cardíacos. La señal obtenida es congruente con la morfología estándar y es muy similar a las señales obtenidas con el simulador. (Ver Fig. 4.2)

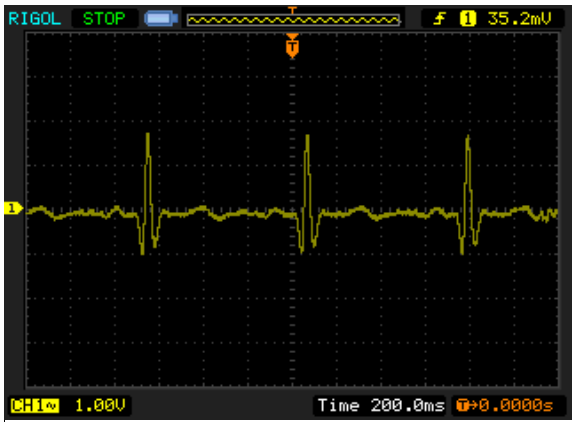


Figura 4.2 Onda de ECG con 3 electrodos con electrodos Ag/AgCl y paciente.

Luego de ensamblar el microchip AD8232 a un adaptador DIP y armar el circuito de prueba #1 en breadboard como se indica en la sección 3.1.6 de experimentación, se conectó a un paciente y se obtuvo la señal de la figura 4.3 se puede ver que la forma de onda es distinta a la del tablero de evaluación, esto se debe a que el ancho de banda de ambos circuitos es distinto (ver tabla 3.5). Se realizó también la validación de la señal. Se midió la amplitud y duración de la forma de onda y segmentos de la señal ECG, según el estándar médico de una señal ECG normal la onda P debe tener una amplitud de100-250mV, y duración entre 90-110ms, el intervalo P-R dura entre 110-200ms, el complejo QRS alcanza una amplitud aproximada de 1.6V y una duración de 70-110ms, el intervalo Q-T dura 350-440ms, el segmento ST dura 50-150ms, y la onda T tiene una amplitud entre 100-500mV [42].

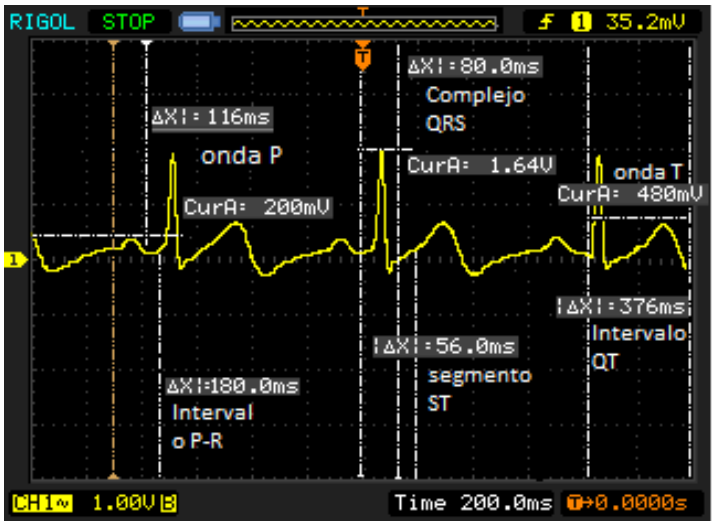


Figura 4.3 Onda de ECG de paciente con 3 electrodos en circuito de prueba #1 en breadboard y mediciones de amplitud y duración de sus ondas y segmentos.

Como parte de la sección 3.1.6 pruebas en breadboard se realizó la medición de ruido para validar la señal ECG capturada con el software Waveforms de la compañía Digilent. Como se mencionó en la sección 2.6 del marco teórico para el diseño de un electrocardiógrafo se debe hacer consideraciones para eliminar diferentes tipos de ruido, y dado que el ancho de banda del circuito y el microchip AD8232 están diseñados para eliminar ruido por artefactos de movimiento, ruido por contacto de electrodo, ruido offset, ruido causado por contracción muscular y órganos, existe también ruido generado por componentes internos del circuito como osciladores, ruido electromagnético por interferencia de artefactos electrónicos, que pueden ser reducidos con aislamiento en el cableado y un chasis con aislamiento el cual el prototipo aún no cuenta con estos últimos.

El ruido medido se realizó sobre la línea basal de la señal ECG, realizando un ajuste en la escala a 30mV/Div para visualizar variaciones (componentes de alta frecuencia) que a escala normal 1V/Div no se perciben de forma clara. El ruido medido después de la fase de amplificación con ganancia de 1100 es de aproximadamente 20mV lo que significa que el ruido de entrada al sistema es de aproximadamente 20µV estando dentro de la normativa IEC 60601-2-27 ≤ 30µV [41] (Ver figura 4.4).

Figura 4.4 Medición de ruido en señal de ECG adquirida con circuito de prueba #1 con paciente. Se realizó un ajuste en la escala hasta 30mV/Div, para observar las variaciones sobre la línea basal de la señal ECG.

4.2 Resultados de simulación del macro modelo SPICE con Multisim.

Como se mencionó en la sección 3.1.7 de la experimentación, para crear un componente SPICE (o componente software) para el software Multisim, se realizó el footprint, y el modelo 3D. Una vez creado el componente SPICE se simuló el circuito de prueba #1 hardware. Se utilizó el circuito de la figura 3.8 para observar la respuesta del sistema en el software y posteriormente realizar pruebas de análisis para estabilidad del sistema. Se puede ver en la Figura 4.5 la morfología estándar de la onda ECG.

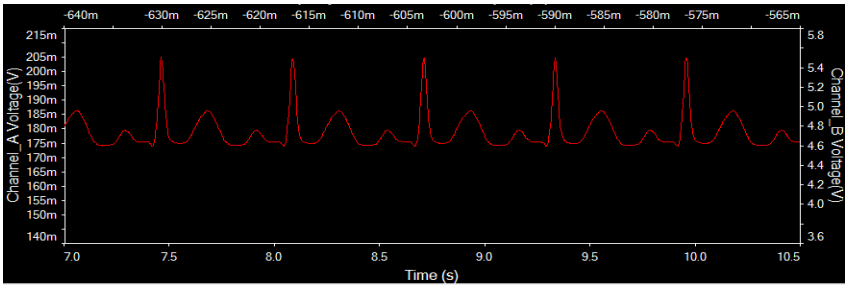


Figura 4.5 Señal simulada en el componente SPICE AD8232 en circuito de prueba #1.

4.3 Análisis con Multisim para estabilidad del sistema.

En las simulaciones de los métodos para la estabilidad del sistema, se puede ver los resultados del análisis Montecarlo (ver Figura 4.6 y 4.7) nos arrojaron los valores de resistencia (Ω), y capacitancia (F) los cuales se ajustaron al 1% de error que permite la normativa ECG IEC 60601-2-27 en los componentes electrónicos hardware del circuito.

Este análisis realiza corridas, en las que varía de forma aleatoria el valor de los componentes del circuito dentro del rango de 1%, observando la figura 4.5 se puede ver la línea basal de la señal ECG simulada está en 178mV, y el resultado de offset después de realizar 10 corridas se mantuvo entre 178.15 -178.70 mV es decir sin ningún cambio significativo para la señal ECG.

Otro análisis utilizado frecuentemente para el estudio de la electrocardiografía es el análisis de Fourier (ver figura 4.8) con este podemos descomponer una señal en armónicos con frecuencias que contengan el ancho de banda de dicha señal y así realizar un mejor diseño de filtros para la adquisición de la señal, como se puede ver en los resultados de la figura 4.8 la magnitud más significativa de la señal ECG está de 1-80Hz, teniendo una disminución de 81-150Hz, y de 151-250Hz es casi despreciable lo que se confirma con los rangos de frecuencia estándar de una señal ECG.

El análisis Worst Case es parecido al Montecarlo, pero a diferencia de este cuando el análisis Montecarlo realiza corridas durante la simulación sus elementos pueden o no llegar al límite del porcentaje de error establecido por el usuario, la forma en que sus elementos varían es aleatoria y en el análisis Worst Case (ver figura 4.9) consiste en forzar la simulación con el máximo del porcentaje de error establecido por el usuario, en este caso 1%, y observar el cambio en la salida. En la figura 4.9 se observa el valor máximo de variación en los componentes hardware del circuito y sirvió en este caso para corroborar la información del análisis Montecarlo la salida del offset sobre la línea basal fue de 178.35mV.

El último análisis es el de barrido de temperatura, (ver Figura 4.10) este nos ayuda a conocer la variación del sistema con respecto al cambio de temperatura, en la tabla de resultados de la simulación se modificó la temperatura desde 0^{o}C-70^{o}C y se graficó la variación del offset la cual se comportó como lo indica la hoja de datos del microchip y en un electrocardiógrafo es muy importante la temperatura debido a que afecta de forma directa al voltaje offset en la entrada del sistema.

# de corrida	tiempo	valor salida	sigma σ	Valor de resistencia		# de corrida	tiempo	valor salida	sigma σ	Valor de resistencia
Nominal Run	0,128	0.178354	0,0608296	rr1 resistance=1e+007		Run #2		0.178779	1,173	rr1 resistance=1.00032e+007
				rr13 resistance=1e+006						rr13 resistance=993661
				rr14 resistance=1e+006						rr14 resistance=995747
				rr4 resistance=1e+007						rr4 resistance=9.95191e+006
				rr5 resistance=1e+007						rr5 resistance=9.98662e+006
				rr6 resistance=1.4e+006						rr6 resistance=1.40014e+006
				rr7 resistance=1e+007						rr7 resistance=9.99571e+006
				rr8 resistance=1e+007						rr8 resistance=1.00172e+007
				rr9 resistance=1e+007						rr9 resistance=1.00586e+007
				rr10 resistance=180000						rr10 resistance=180197
				rr11 resistance=180000						rr11 resistance=179741
				rr12 resistance=360000						rr12 resistance=359973
				rr15 resistance=100000						rr15 resistance=100157
				rr17 resistance=1e+006						rr17 resistance=1.00176e+006
Run #1		0.177784	1,71345	rr1 resistance=1.0039e+007		Run #3		0.178305	(0 0,201198	rr1 resistance=9.997e+006
				rr13 resistance=1.00055e+006						rr13 resistance=998577
				rr14 resistance=1.00242e+006						rr14 resistance=1.0009e+006
				rr4 resistance=1.00207e+007						rr4 resistance=9.99355e+006
				rr5 resistance=1.00281e+007						rr5 resistance=1.00371e+007
				rr6 resistance=1.40792e+006						rr6 resistance=1.39924e+006
				rr7 resistance=9.99266e+006						rr7 resistance=9.98965e+006
				rr8 resistance=9.99054e+006						rr8 resistance=9.9924e+006
				rr9 resistance=9.97243e+006						rr9 resistance=1.00025e+007
				rr10 resistance=179491						rr10 resistance=179674
				rr11 resistance=179638						rr11 resistance=180191
				rr12 resistance=358430						rr12 resistance=359994
				rr15 resistance=99664						rr15 resistance=100217
				rr17 resistance=999565						rr17 resistance=997243

Figura 4.6 Resultados de análisis Montecarlo al variar el valor de las resistencias del circuito prueba#1, 1% y cómo afecta el valor de salida del circuito ante estas variaciones aleatorias.

Vsalida	sigma σ	valor de Capacitor	# de corrida	Vsalida	sigma σ	valor de Capacitor
0.178354	1,7335	cc1=3.3e-007	Run #2	0.178354	0,611211	cc1=3.28894e-007
		cc2=3.3e-007				cc2=3.29534e-007
		cc3=1e-007				cc3=1.00438e-007
		cc4=1e-007				cc4=1.00331e-007
		cc5=1e-009				cc5=9.97479e-010
		cc6=1e-008				cc6=9.99139e-009
		cc7=1.5e-009				cc7=1.50082e-009
0.178354	0,240894	cc1=3.27885e-007	Run #3	0.178354	1,20636	cc1=3.30061e-007
		cc2=3.29697e-007				cc2=3.28787e-007
		cc3=9.9831e-008				cc3=9.99457e-008
		cc4 =9.94774e-008				cc4=1.00269e-007
		cc5=9.99883e-010				cc5=1.00345e-009
		cc6=1.00051e-008				cc6=9.9976e-009
		cc7=1.50674e-009				cc7=1.49523e-009

Figura 4.7 Resultados de análisis Montecarlo al variar el valor de los capacitores del circuito prueba#1, 1% y cómo afecta el valor de salida del circuito ante estas variaciones aleatorias.

# armonico	Frecuencia	Magnitud	Fase												
0	0	4.1851E-05	0	26	65	0.00228649	-0.050095	52	130	0.0004116	-0.0654167	78	195	0.00023284	0.25607833
1	2.5	0.00757386	0.06561	27	67.5	0.00240893	-0.5611111	53	132.5	0.0005401	-0.2210889	79	197.5	0.00011051	0.43610389
2	5	0.00964988	0.117755	28	70	0.00139729	0.72744444	54	135	0.00051125	-0.4088444	80	200	0.00018685	0.26500222
3	7.5	0.00663503	-0.1792	29	72.5	0.0006135	0.11683944	55	137.5	0.00039461	-0.4803222	81	202.5	9.2384E-05	0.02494456
4	10	0.00137026	-0.5497222	30	75	0.00150753	-0.3006111	56	140	0.0004536	-0.5878333	82	205	7.3173E-05	0.35901833
5	12.5	0.00018146	0.200645	31	77.5	0.00114309	-0.0485017	57	142.5	0.00036128	-0.8787778	83	207.5	9.7119E-05	0.14597056
6	15	0.00210322	0.85638333	32	80	0.00063993	-0.2573389	58	145	0.00011169	0.65920556	84	210	3.2423E-05	0.16542722
7	17.5	0.00389238	0.51970944	33	82.5	0.00068134	-0.5317722	59	147.5	4.345E-05	-0.4352667	85	212.5	7.6328E-05	0.29751
8	20	0.00254235	0.26115722	34	85	0.00011005	-0.5117056	60	150	0.00013053	0.47304944	86	215	6.3558E-05	0.07746722
9	22.5	0.00168679	0.42719833	35	87.5	0.00108638	-0.6376111	61	152.5	0.00030407	0.05981389	87	217.5	3.4663E-05	0.22526944
10	25	0.00239421	0.22005111	36	90	0.00125569	-0.0238778	62	155	0.00026831	-0.1770167	88	220	5.503E-05	0.13106222
11	27.5	0.00085028	-0.29325	37	92.5	0.00066275	-0.2989833	63	157.5	0.00028345	-0.1404389	89	222.5	2.4854E-05	-0.0863778
12	30	0.00138059	0.74448333	38	95	0.00057687	-0.6175556	64	160	0.00042131	-0.3379167	90	225	2.4168E-05	0.38650333
13	32.5	0.00090502	0.38563333	39	97.5	0.00057947	-0.8181111	65	162.5	0.00028119	-0.6477778	91	227.5	3.3454E-05	0.20171389
14	35	0.0011293	0.87306667	40	100	0.00031758	-0.7683889	66	165	2.9142E-05	0.00819417	92	230	2.4429E-05	0.25626222
15	37.5	0.00188966	0.55751111	41	102.5	0.0001619	0.92606667	67	167.5	0.00019352	-0.2747722	93	232.5	3.9853E-05	0.22121111
16	40	0.00041201	0.32955944	42	105	0.00030707	0.36139056	68	170	0.00015098	-0.3698167	94	235	3.7053E-05	0.06340333
17	42.5	0.00152417	0.78672222	43	107.5	0.00076509	-0.1928944	69	172.5	0.00023879	-0.3966944	95	237.5	2.79E-05	0.04385617
18	45	0.00130441	0.46676278	44	110	0.000783	0.20376389	70	175	0.00024116	-0.6963889	96	240	2.7714E-05	0.01328589
19	47.5	0.00097587	0.97843889	45	112.5	0.00059698	-0.1011056	71	177.5	5.1922E-05	-0.9266667	97	242.5	2.1334E-05	0.00424564
20	50	0.00256204	0.68081667	46	115	0.00068192	-0.3608778	72	180	0.00018371	-0.6328333	98	245	2.3117E-05	0.07217722
21	52.5	0.00150831	0.44304778	47	117.5	0.00060371	-0.4523111	73	182.5	0.00028702	0.86107222	99	247.5	2.5332E-05	0.04873817
22	55	0.00255342	0.70070556	48	120	0.00037594	-0.5369278	74	185	0.00031695	0.24017278	100	250	-1.516E-05	0.04621183
23	57.5	0.00533547	0.37049833	49	122.5	0.00026922	-0.7395556	75	187.5	0.00040347	-0.4028389				
24	60	0.00554253	-0.050095	50	125	0.00026962	-0.915	76	190	0.00050813	-0.9475				
25	62.5	0.00356578	-0.5611111	51	127.5	0.00034072	0.87252778	77	192.5	0.00046376	0.6017				

Figura 4.8 Resultados de análisis Fourier al descomponer la señal obtenida por el circuito de prueba #1, en 100 armónicos con frecuencia de 0-250Hz.

Worst Case Run	
DC operación para todos los componentes:: 0.00211212, (1.18423% of nominal)	
Los cambios de tolerancia 1% para lograr el peor de los casos	
rr13 decreció a 990000	rr13 incrementó a 110000
rr14 decreció a 990000	rr14 incrementó a 110000
rr1 decreció a 9.9e+006	rr1 incrementó a 1.1e+006
rr4 incrementó a 1.01e+007	rr4 decrecio a .99e+007
rr5 decreció a 9.9e+006	rr5 aumentó a 1.1e+006
rr6 incrementó a 1.414e+006	rr6 decreció a 1.386e+006
rr7 incrementó a 1.01e+007	rr7 decreció a .99e+007
rr8 incrementó a 1.01e+007	rr8 decreció a .99e+007
rr9 incrementó a 1.01e+007	rr9 decreció a .99e+007
rr10 incrementó a 181800	rr10 decreció a 178200
rr11 incrementó a 181800	rr11 decreció a 178200
rr12 decreció a 356400	rr12 aumentó a 363600
rr15 incrementó a 101000	rr15 decreció a 990000
rr17 decreció a 990000	rr17 aumentó a 110000

Figura 4.9 Resultado en el peor de los casos con variación de 1% en los componentes del circuito prueba #1.

Temperatura	voltaje	Temperatura	Voltaje	Temperatura	Voltaje
0	-0.1125365	26	0	52	0.1004311
1	-0.1083775	27	0.0041774	53	0.1046234
2	-0.1042178	28	0.0083555	54	0.1088162
3	-0.1000573	29	0.0125341	55	0.1130097
4	-0.0958962	30	0.0167135	56	0.1172037
5	-0.0917344	31	0.0208934	57	0.1213983
6	-0.0875718	32	0.0250741	58	0.1255935
7	-0.0834086	33	0.0292553	59	0.1297893
8	-0.0792446	34	0.0334372	60	0.1339857
9	-0.07508	35	0.0376197	61	0.1381826
10	-0.0709146	36	0.0418028	62	0.1423802
11	-0.0667486	37	0.0459865	63	0.1465782
12	-0.0625819	38	0.0501709	64	0.1507769
13	-0.0584144	39	0.0543559	65	0.1549761
14	-0.0542463	40	0.0585415	66	0.159176
15	-0.0500776	41	0.0627277	67	0.1633763
16	-0.0459081	42	0.0669146	68	0.1675773
17	-0.041738	43	0.071102	69	0.1717788
18	-0.0375672	44	0.0752901	70	0.1759808
19	-0.0333957	45	0.0794788		
20	-0.0292235	46	0.083668		
21	-0.0250507	47	0.0878579		
22	-0.0208772	48	0.0920484		
23	-0.0167031	49	0.0962395		
24	-0.0125283	50	0.1004311		
25	-0.0083528	51	0.1046234		

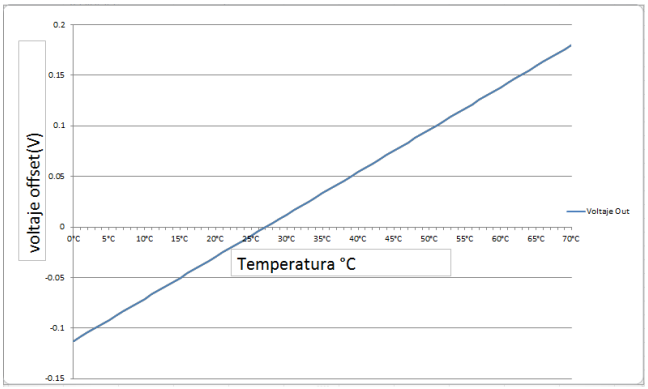


Figura 4.10 Tabla y gráfica del análisis de barrido de temperatura de 0-70 °C. (Temperatura de operación).

4.4 Señal ECG con paciente en cada modo de operación.

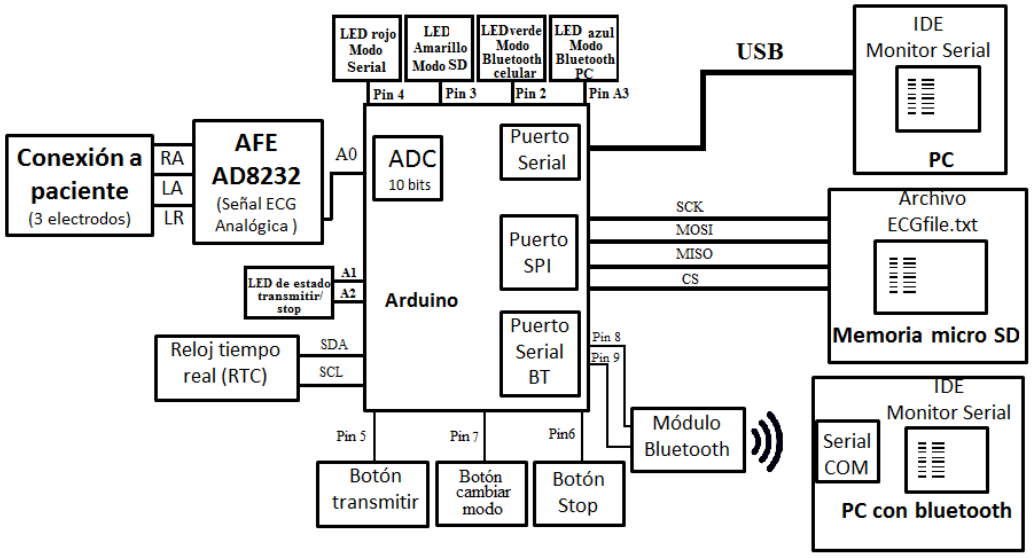


Figura 4.11 Estructura y diagrama de conexión del electrocardiógrafo para monitoreo de uso personal y tiempo prolongado.

4.4.1 Transmisión serial de señal ECG de paciente a puerto COM de PC.

Para realizar las pruebas con pacientes se realizó el diagrama de la figura 4.11, se utilizó el software de LABVIEW y se configuró el puerto COM del PC como se indica en la sección 3.2.6 para la recepción de datos. Como resultado se obtuvo señal ECG proveniente de un paciente transmitida en tiempo real (Ver Figura 4.12).

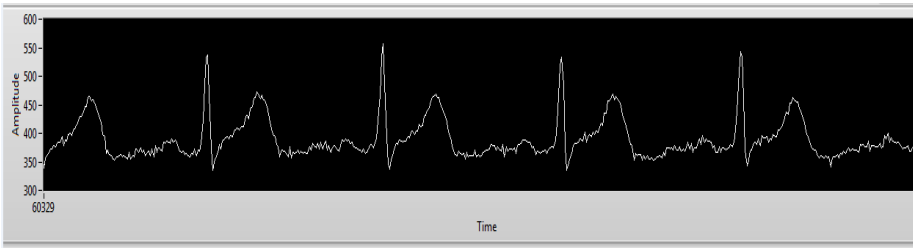


Figura 4.12 Modo #1 Cable serie capturada en LABVIEW.

4.4.2 Grabación en SD card con paciente.

Para la etapa de grabación en data logger, se utilizó el diagrama de la figura 4.11 y se registraron los datos en un archivo llamado "ECGfile.txt", se siguieron los pasos de la sección 3.4.3 para graficar la señal ECG con el software THEREMINO pero se puede graficar en cualquier software compatible con el formato .txt. Como resultado se obtuvo un registro por tiempo prolongado de ECG almacenado en la memoria micro SD. En la figura 4.13 se puede observar en el registro un fenómeno atípico, el tercer pulso tiene 4 cuadros antes de presentarse el siguiente pulso, mientras que en los demás se presentan 5 cuadros de diferencia, con esto podemos darnos cuenta que el prototipo implementado es capaz de registrar datos atípicos ya sea que puedan reflejar algún estado de salud irregular o patología.

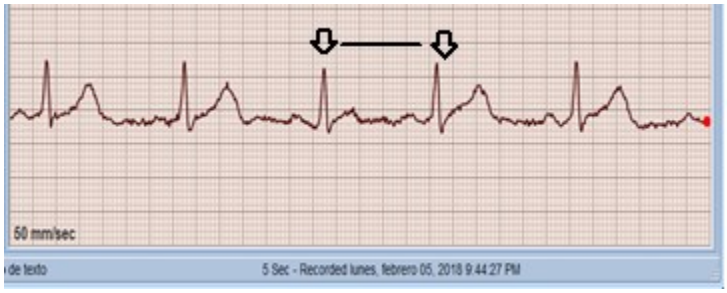


Figura 4.13 Modo #2 Grabación SD gráfica en THEREMINO.

4.4.3 Transmisión bluetooth con señal de paciente.

Para la transmisión inalámbrica de la señal ECG, se utilizó el diagrama de la figura 4.11, y se transmitieron los datos mediante 2 métodos: conexión a un teléfono inteligente con sistema Android, y comunicación inalámbrica a PC con bluetooth integrado. Para graficar la señal transmitida a un teléfono inteligente se siguió el procedimiento de las secciones 3.5.1 y 3.5.3 de la sección de experimentación, se utilizó la aplicación "Arduino Graphics" y se obtuvo la señal ECG en tiempo real (ver figura 4.14).

Figura 4.14 Modo #3 Bluetooth Screenshot en celular Android.

4.4.4 Transmisión bluetooth PC.

Para la transmisión inalámbrica hacia un PC. Se utilizó la misma interfaz de LABVIEW (ver Figura 4.15) mediante el puerto COM del bluetooth integrado en la laptop, esto mismo se puede realizar con cualquier computadora de escritorio y una unidad bluetooth.

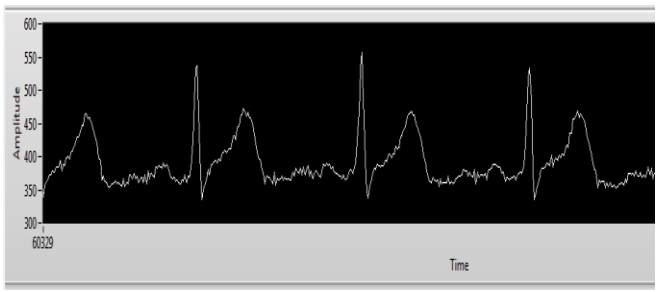


Figura 4.15 Modo #4 Bluetooth en PC capturada en LABVIEW.

4.5 Filtros digitales aplicados al sistema de filtrado digital en tiempo real.

4.5.1Filtro peine para eliminar el ruido de 60 Hz, sus armónicos y offset (filtro #1).

La ecuación de diferencias #1 se implementó en el programa de Arduino, con un buffer circular para recorrer las lecturas y aplicar la operación (ver secciones 3.6.1-3.6.5). El filtro peine fue implementando para eliminar la frecuencia de 60 Hertz, sus armónicos y offset. El filtro en el sistema en tiempo real se probó con la señal simulada #4 con 4 tipos de ruidos 60, 120, 180, y offset de la sección 3.6.3. Como se observa en la figura 4.16 los ruidos fueron exitosamente eliminados, y como en la simulación este filtro generó una distorsión en la forma de la onda debido a que la respuesta de fase de este filtro no es lineal.

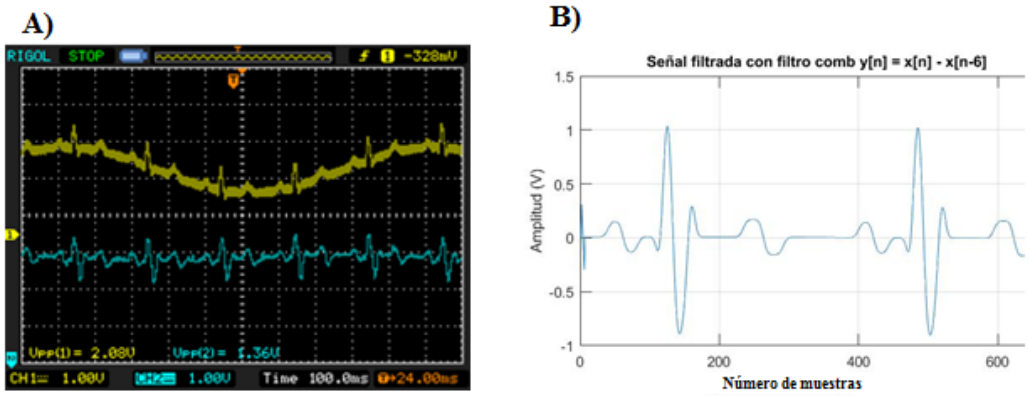


Figura 4.16 A) Señal de prueba #4 en la entrada del sistema (amarilla), y Señal de salida (Azul) respuesta del filtro #1. B) Respuesta de filtro #1 simulada en MATBLAB.

Este filtro elimina el offset y como se sabe el convertidor analógico no procesa valores negativos de voltaje, por lo que fue necesario sumarle una señal de voltaje en bits, sumando un valor de 127 bits antes de ser enviada al PORT D esto para simular un offset al momento de convertir el valor analógico a un valor digital.

4.5.2 Filtro peine para eliminar el ruido de 60Hz (ecuación #2).

El siguiente filtro utiliza la ecuación #2 que sirve para eliminar el ruido de 60Hz y el armónico de 180Hz. Se utilizó primero la señal de prueba #2 de la sección 3.6.3 (ver Figura 3.42) debido a que contiene únicamente el ruido de 60Hz.

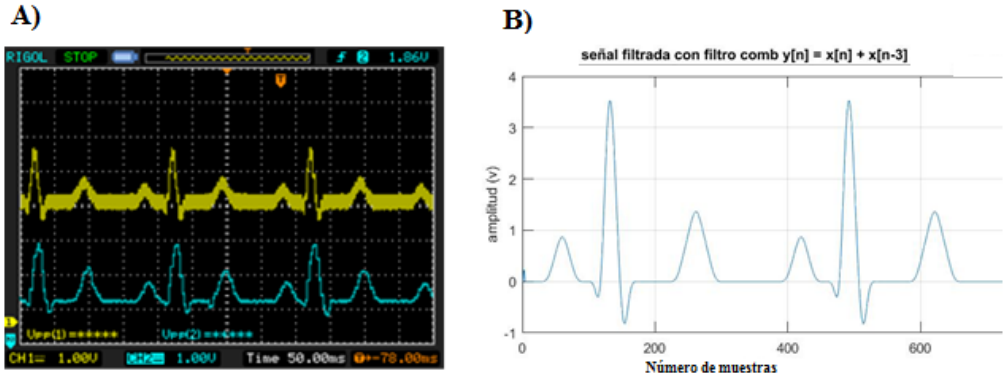


Figura 4.17 A) Señal de prueba #2 en la entrada del sistema (amarilla), y Señal de salida (Azul) respuesta del filtro #2. B) Respuesta de filtro #2 simulada en MATBLAB.

Este filtro también fue probado con la señal de prueba #4. En la figura 4.18 podemos observar una señal sin las componentes de ruido de 60Hz y 180Hz, pero este filtro no elimina la componente de 120Hz, ni el offset de 0.1Hz.

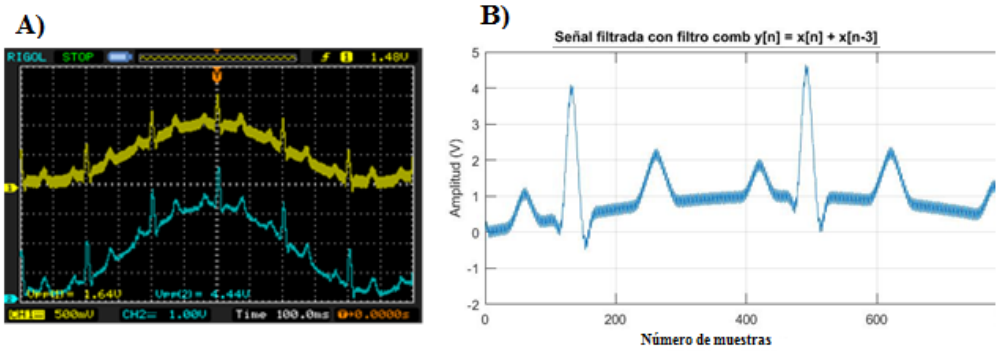


Figura 4.18 A) Señal de prueba #4 en la entrada del sistema (amarilla), y Señal de salida (Azul) respuesta del filtro #2. B) Respuesta de filtro #2 simulada en MATBLAB.

4.5.3 Filtro que elimina solo la frecuencia 60Hz (ecuación #3).

Para este filtro se ingresó la señal de prueba #3 con ruido de 60, 120, 180Hz. El diseño de este filtro solo elimina la componente de 60Hz, por lo que se puede apreciar en la figura 4.19 la respuesta del filtro nos otorga una señal de ECG en la que se ha eliminado el ruido de 60Hz pero con una señal remanente que contiene las componentes de ruido 120Hz y 180Hz. Este filtro tampoco filtra la señal offset se comprobó al introducir la señal de prueba #4 (ver Figura 4.20).

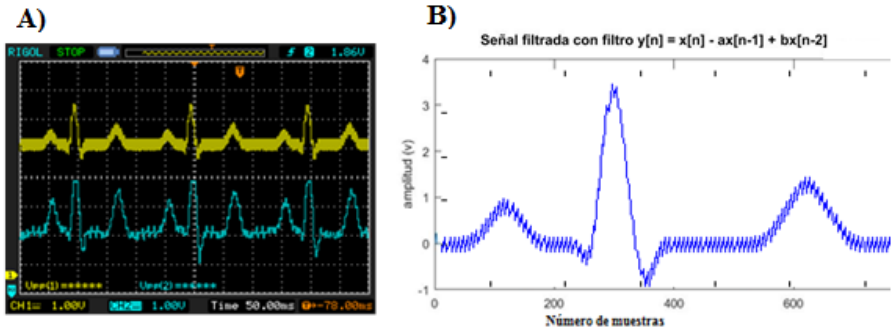


Figura 4.19 A) Señal de prueba #3 en la entrada del sistema (amarilla), y Señal de salida (Azul) respuesta del filtro #3. B) Respuesta de filtro #3 simulada en MATBLAB.

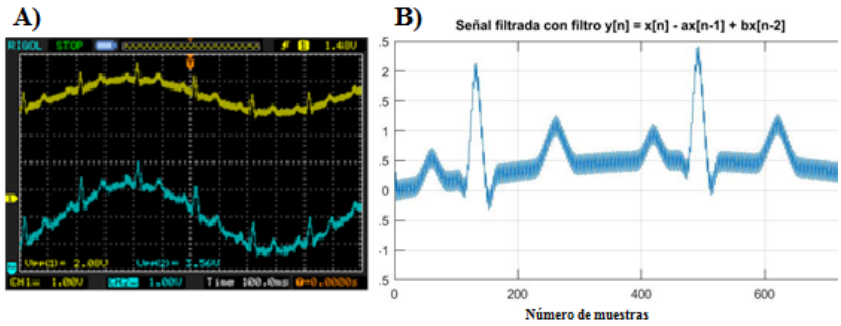


Figura 4.20 A) Señal de prueba #4 en la entrada del sistema (amarilla), y Señal de salida (Azul) respuesta del filtro #3. B) Respuesta de filtro #3 simulada en MATBLAB.

4.5.4 Filtro Notch periódico (ecuación #4).

El filtro #4 fue probado con la señal de prueba #4 para un α=0.5 y un α=0.9. La figura 4.21 muestra la señal de salida del filtro (azul) para un α=0.5. Se puede observar que al igual que en la simulación la señal es muy parecida a la señal de salida del filtro #1 conforme se acerca el valor de α a cero, pues en este caso la ecuación del filtro#4 sería igual a la del filtro #1, se puede observar que conforme alfa se acerca a 1 la distorsión de la señal disminuye y deja de eliminar el ruido offset (ver figura 4.22).

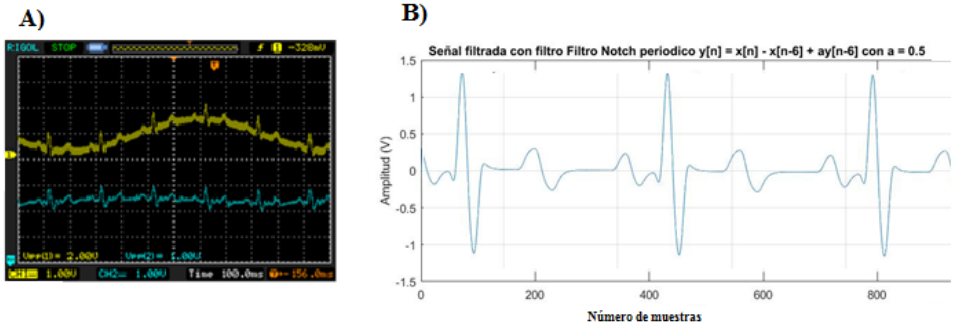


Figura 4.21 A) Señal de prueba #4 en la entrada del sistema (amarilla), y Señal de salida (Azul) respuesta del filtro #4. B) Respuesta de filtro #4 simulada en MATBLAB. Para un α = 0.5.

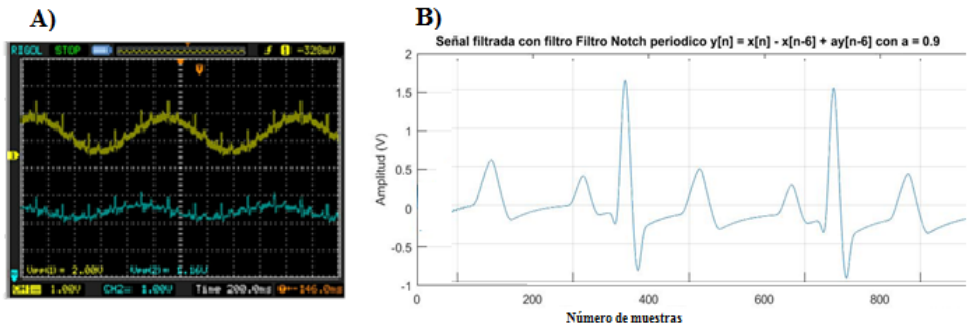


Figura 4.22 A) Señal de prueba #4 en la entrada del sistema (amarilla), y Señal de salida (Azul) respuesta del filtro #4. B) Respuesta de filtro #4 simulada en MATBLAB. Para un $\alpha = 0.9$.

4.6 Pruebas realizadas con data logger.

4.6.1 Señal virtual.

Se creó una señal virtual de ECG utilizando software MATLAB como se indica en la sección 3.6.3 pero en vez de utilizar el generador Tektronik se reprodujo con el generador de Digilent y software Waveforms, (ver Figura 4.23) para realizar la grabación de la señal ECG y observar la estabilidad del prototipo durante tiempos prolongados (24 horas).

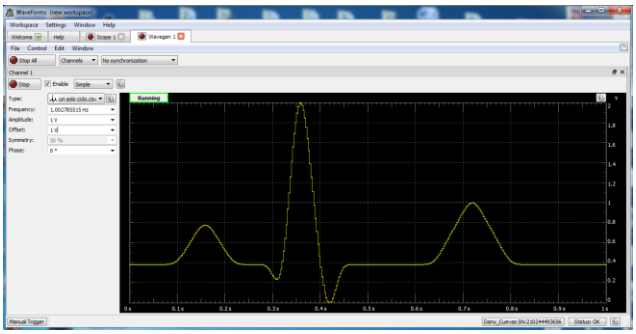


Figura 4.23 Señal ideal ECG generada a 360Hz.

4.6.2 Pruebas de grabación en micro SD por tiempo prolongado.

Para conocer el espacio de memoria que consume el registro de ECG se realizaron pruebas con 1, 12, 24 horas, con frecuencias de muestreo de 360Hz, y 500Hz. Se modificó también el tipo de contenido del registro con dato y hora, y dato, hora y fecha, los resultados del espacio de los archivos .txt generados se presentan en la tabla 4.1.

Tabla 4.1 Espacio de memoria consumido en data logger, MB utilizados en memoria SD.

Duración de la prueba	Tipo de contenido. Dato y hora	Tipo de contenido. Dato hora y fecha	Frecuencia de muestreo
1 hora	18.4 MB	30.5MB	360Hz
12 horas	206 MB	366 MB	360Hz
24 horas	407 MB	732 MB	360Hz
1 hora	23.7 MB	40.3 MB	500Hz
12 horas	284 MB	513 MB	500Hz
24 horas	524 MB	991 MB	500Hz

Archivo .txt con dato y hora con 360Hz.

- Se realizaron pruebas de 1, 12, y 24 horas de grabación, el tamaño de los archivos fueron 18.4MB, 206MB, y 407MB respectivamente.

Archivo .txt con dato, hora y fecha con 360Hz.

- Se realizaron pruebas de 1, 12, y 24 horas de grabación, el tamaño de los archivos fueron 30.5MB, 366MB, y 732MB respectivamente.

Archivo .txt con dato, y hora con 500Hz.

- Se realizaron pruebas de 1, 12, y 24 horas de grabación, el tamaño de los archivos fueron 23.7MB, 284MB, y 524MB respectivamente.

Archivo .txt con dato, hora y fecha con 500Hz.

- Se realizaron pruebas de 1, 12, y 24 horas de grabación, el tamaño de los archivos fueron 40.3MB, 513MB, y 991MB respectivamente.

4.6.3 Pruebas con paciente

Una vez que se realizaron las pruebas de larga duración del prototipo ECG con una señal simulada, se realizó la configuración de 3 electrodos (ver configuración en sección 3.1.4). Se registraron archivos de ECG con un paciente sin antecedentes de problemas cardíacos en condiciones de reposo durante 1 y 2 horas, y se graficaron en los software Excel y THEREMINO (Ver figura 4.24).

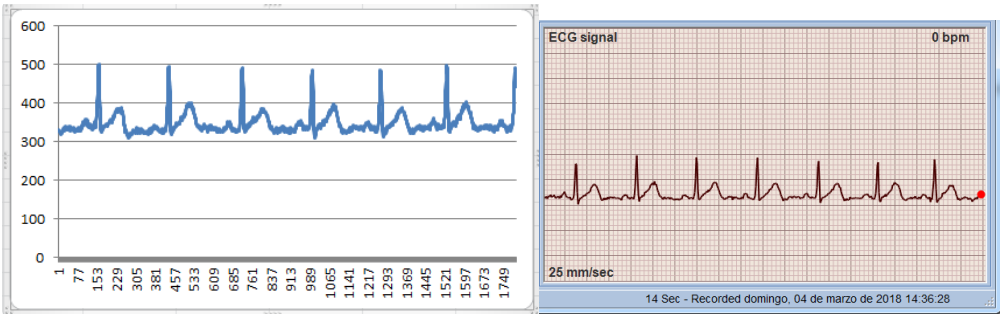


Figura 4.24 Gráfica en formato Excel, y THEREMINO.

Pruebas de grabación en SD con muestreo de 360 y 500 Hz en paciente.

Archivo .txt con dato y hora.

Se realizó prueba de 1 hora de grabación.
Se realizó prueba de 2 horas de grabación.

Archivo .txt con dato, hora y fecha.

Se realizó prueba de 1 hora de grabación.
Se realizó prueba de 2 horas de grabación.

4.6.4 Análisis de un registro ECG por tiempo prolongado.

El análisis de las señales de ECG se realizó de forma visual observando la morfología de la señal ECG de larga duración de los archivos de 1 y 2 horas. Se utilizó una señal virtual y una de paciente, se importaron y graficaron los datos de los registro en el software Excel para observar la señal ECG detalladamente. Al revisar el archivo "ECGfile.txt" de una señal virtual grabada en la memoria SD, se observa que un pico del complejo QRS no se ha registrado adecuadamente dado que se trata de una señal virtual no debería presentar cambio (ver figura 4.25). Se revisaron los documentos y este fenómeno no presenta un patrón si no que es inusual, se presentó en una ocasión en cada archivo de 1 hora, y solo 2 veces en un archivo de 2 horas.

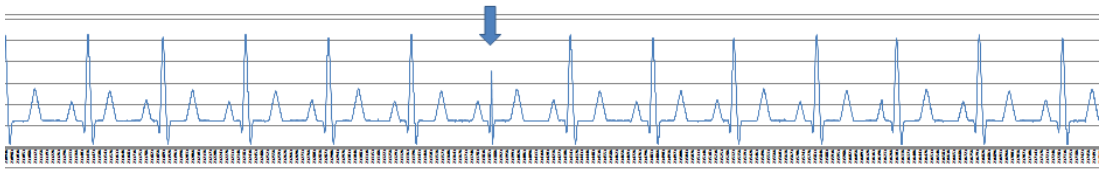


Figura 4.25 Identificación del evento anormal en el registro de la señal ECG virtual de larga duración.

Se intentó caracterizar esta anomalía en la señal para poder determinar el origen. Como se puede ver en la figura 4.26 se ha extraído el fragmento de la señal que posee el evento atípico, y se ha revisado las muestras entre cada intervalo R-R de la señal ECG. Puede verse en la figura que la cantidad de muestras en los latidos regulares es de 360 muestras. Y en los intervalos aledaños al pico QRS irregular hay 342 muestras al pico anterior, y 337 al pico posterior. Esto sugiere que se han perdido 41 muestras entre dos latidos. Las posibles causas se siguen investigando, pero pueden ser: problema con el ADC del microcontrolador (una posible inestabilidad), algún retardo al realizar las interrupciones, o un problema con el generador de funciones de la compañía Digilent.

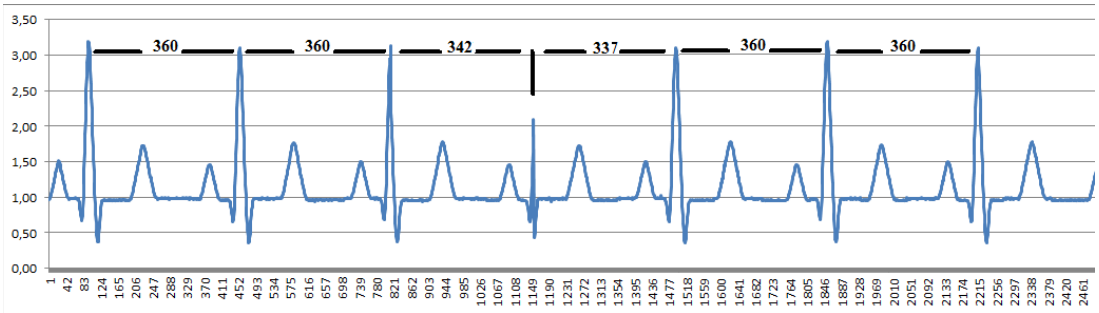


Figura 4.26 Extracción del evento anormal en el registro de la señal ECG virtual y medición de tiempo del RTC.

La caracterización del evento irregular en cada uno de los registros ECG se puede ver en la tabla 4.2. Se puede ver que el número de muestras perdidas por ciclo no excede las 50 muestras por latido cuando ocurre este evento atípico. Y comparado con el tamaño del archivo mayor a 1,200,000 o 3,600,000 muestras puede no ser significativo; pero se desconoce los efectos que este evento pueda generar en un diagnóstico o monitoreo de un ECG de larga duración. Por lo cual seguimos evaluando para conocer su relevancia y su causa, y corregirse en el trabajo a futuro de esta investigación, también se espera revisar los archivos grabados de 24 horas para asegurar registros que no presente este tipo de eventos.

Tabla 4.2 Caracterización del evento atípico del complejo QRS en registros ECG de larga duración. Se consideran la duración del registro, frecuencia de muestreo, número de muestra que presentó la irregularidad, número total de datos del registro, tiempo en que se registró el evento y número de muestras perdidas

Nombre de registro	# de muestra que presentó un evento atípico	# de datos del registro	Tiempo del evento atípico	# de muestras pérdidas durante el evento atípico.
Registro 1 hora 360Hz	53,157	1,288,907	2:28:00	41
Registro 1 hora 500Hz	19,301	1,784,059	0:39:00	49
Registro 2 horas 360Hz	2,344,956	2,579,257	1:49:06	31
Registro 2 horas 500Hz	3,247,076	3,568,588	1:49:12	34
Registro 2 horas 500Hz	3,556,625	3,568,588	1:59:37	27

Una vez caracterizados los eventos atípicos de los registros de la señal ideal, se realizó también la revisión de los registros de ECG con señales adquiridas de paciente, y en estos casos no se presentaron eventos irregulares, aunque si algunos artefactos típicos de movimiento (ver ejemplo de esta señal en la Figura 4.27).

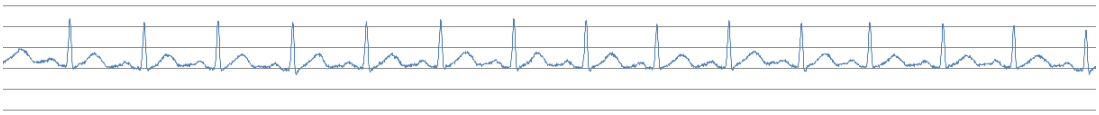


Figura 4.27 Señal ECG con paciente, sin eventos atípicos.

4.7 Consumo de corriente y medición de corriente de fuga.

4.7.1 Análisis de un registro ECG por tiempo prolongado.

En esta sección se presentan los resultados de la medición de consumo de corriente, los cuales se midieron como se indica en la sección 3.8.2 de experimentación. Se armó la conexión de la figura 3.57 y se midió el consumo de corriente de cada componente individual y en cada modo de operación (ver tabla 4.3 y 4.4).

Tabla 4.3 Consumo de corriente de componentes.

Componente	Medición (mA)
Arduino Nano	15.8mA
Módulo SD	35-45mA
Módulo Bluetooth	20-25mA
LED	20mA

Tabla 4.4 Consumo de corriente del prototipo en cada modo de operación.

Modo de operación	Medición mA
Reposo (sin transmisión).	55-70mA
Transmisión en serie.	125-135mA
Grabación en SD.	105-135mA
Transmisión Bluetooth Cel.	120-125mA
Transmisión Bluetooth PC.	120-125mA

Los valores obtenidos de consumo de corriente en cada modo de operación se utilizaron para calcular y comprobar el tiempo de operación máximo del prototipo por ciclo de carga de la batería. Para esto se utilizaron 2 baterías de 7.4V de polímero de litio (LIPO) de 1300mA, y 4000mA las cuales se utilizaron como alimentación del prototipo. Se puede ver en la tabla 4.5 el tiempo máximo que se dejó funcionando el prototipo antes de agotar las baterías.

Tabla 4.5 Máxima duración con baterías, en cada modo de operación.

Modo de operación	Batería LiPo 1300mA	Batería Li-ion 4000mA
Reposo (sin transmisión).	18hr.	55hr.
Transmisión en serie.	9hr, 30min.	29hr, 30 min.
Grabación en SD.	10hr, 30 min.	31hr, 30min.
Transmisión Bluetooth Cel.	10hr, 30min.	31hr, 30min.
Transmisión Bluetooth PC.	10hr, 30min.	31hr, 30min.

4.7.2 Medición de la corriente de fuga.

Para medir la Corriente de fuga se realizó la conexión de la Figura 3.58 siguiendo las indicaciones de un manual de seguridad hospitalaria de la marca Fluke [44] como se menciona en la sección 3.8.3 de la experimentación. Se puede ver en la Tabla 4.6 la corriente de fuga del circuito, la cual se midió en cada modo de operación, y en modo de reposo.

Tabla 4.6 Corriente de fuga en cada modo de operación.

Modo de operación	Corriente de fuga
Reposo (sin transmisión).	1-2µA
Transmisión en serie.	1-2µA
Grabación en SD.	1-2µA
Transmisión Bluetooth Cel.	1-2µA
Transmisión Bluetooth PC.	1-2µA

4.8 Circuitos ECG con filtrado hardware para diferentes usos del dispositivo.

En la sección de experimentación 3.7 se muestra el ancho de banda, circuitos, y respuesta en frecuencia de los filtros hardware para aplicaciones de diseño el microchip AD8232 (ver tabla 3.5). Se puede ver en las figuras 4.28-4.32 la señal ECG de salida de cada aplicación: monitor cardíaco, monitor cerca del pecho, monitor *fitness* y la réplica hecha en breadboard del tablero EVAL-AD8232. En todos los casos las señales registradas con nuestro prototipo muestran comportamientos típicos de estas aplicaciones.

4.8.1 Monitor cardíaco.

La configuración de monitor cardíaco está diseñada para monitorear la forma de la onda del ECG su funcionamiento es igual al de un monitor ECG. Tiene como requisito que el paciente permanezca inmóvil durante la medición, o se verá una distorsión en la forma de la onda por ruido por artefactos de movimiento. Se grabó durante 1 minuto con este circuito, la señal resultante puede verse en la Figura 4.28. La morfología corresponde a la del ECG convencional, aunque esta aplicación es sensible a los artefactos por movimiento.

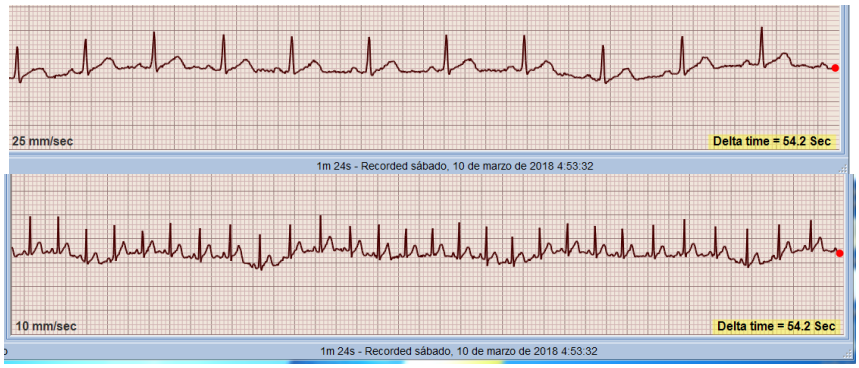


Figura 4.28 Señal ECG con circuito para aplicación de monitor cardíaco (THEREMINO). En el trazo superior, la primera onda T está muy suavizada, y tarda menos el ciclo.

4.8.2 Medición cerca del corazón (pecho)

Para los dispositivos portátiles de ejercicio, el AD8232 generalmente se coloca en una cápsula cerca del corazón. Los dos electrodos (RA, LA) de detección se colocan debajo de los músculos pectorales; solo cuenta con una etapa de amplificación debido a que la señal del corazón es fuerte y hay menos interferencia de artefactos musculares. Se grabó durante 1 minuto la señal de ECG se puede ver en la figura 4.30 que la señal ECG no se aprecia claramente, esto se debe a que la amplificación de este circuito es 100 a diferencia

de las otras aplicaciones que es 1100. En la figura 4.29 se ha utilizado el osciloscopio con una escala de 100mV.

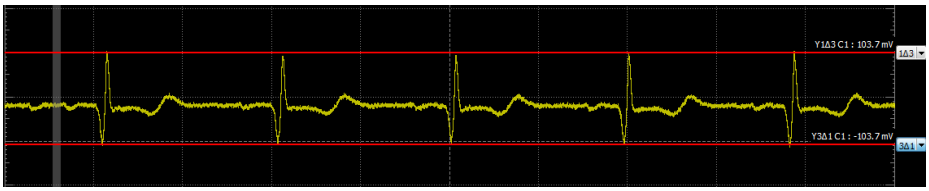


Figura 4.29 Señal ECG con circuito para aplicación cerca del pecho (osciloscopio).

Figura 4.30 Señal ECG con circuito para aplicación cerca del pecho (THEREMINO).

4.8.3 Monitoreo durante el ejercicio.

En esta aplicación, la medición se realiza en las manos o muñecas y está pensado para su implementación en aparatos *fitness* como caminadoras o bicicletas. Como en la sección de experimentación 3.7.3 se menciona tiene un filtro de banda estrecha lo cual distorsiona significativamente la forma de onda del ECG. Por lo tanto, solo es adecuado para determinar la frecuencia cardíaca y no para analizar las características de la señal del ECG. Se grabó 1 minuto con este circuito, la señal resultante de este circuito puede verse en la figura 4.31 que las ondas P y T se modifican con los artefactos de movimiento, pero el pico se mantiene constante esto para monitorear el número de pulsaciones por minuto.

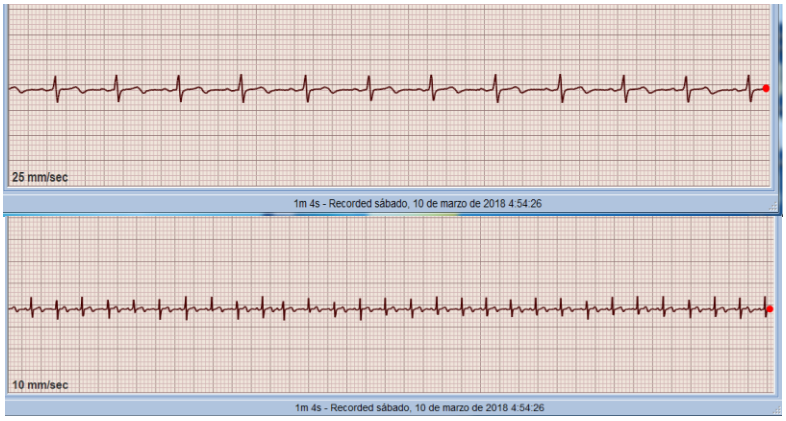


Figura 4.31 Señal ECG con circuito para aplicación monitoreo durante el ejercicio (THEREMINO). En el trazo inferior, en los latidos 5, 8 y 12 desaparece la parte negativa de la onda esto se debe a los movimientos realizados durante la prueba.

4.8.4 Replica del Tablero Eval-AD8232 Analog Devices con aplicación de monitor ECG.

Para esta prueba se replicó el circuito del tablero de evaluación de la compañía Analog Devices con aplicación de monitor ECG, a diferencia de la aplicación de monitor cardíaco de la sección 4.9.1 el cual tiene un filtro pasa altas de 7.2Hz, y un filtro pasa bajas de 25Hz, su respuesta del filtro según la hoja de datos es más estrecha que la aplicación de monitor cardíaco lo cual modifica la onda resultante. Se grabó 1 minuto de señal ECG y se graficó se puede apreciar la morfología de ECG convencional, se pueden ver algunos eventos en el trazo inferior, generadas probablemente por artefactos de movimiento. (Ver figura 4.32).

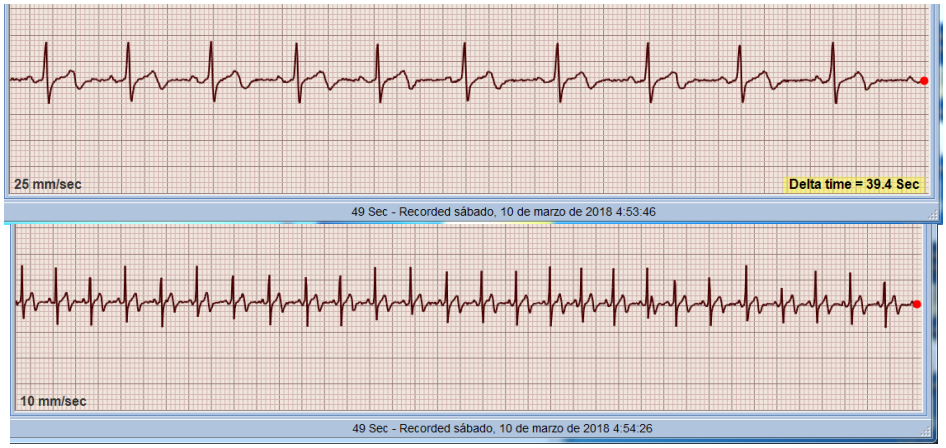


Figura 4.32 Señal ECG con circuito para aplicación monitoreo durante el ejercicio (THEREMINO). Se puede observar un evento raro en el trazo inferior.

4.9 Medición de CMRR

Como se indica en la sección 3.8.1 de experimentación, se implementaron los diagramas de las figuras 3.55 y 3.56 para la medición del CMRR. Este valor hace referencia al rechazo al modo común es decir la capacidad del amplificador de instrumentación para no amplificar las señales similares en su entrada. Para determinar el CMRR se midieron la ganancia en modo común y ganancia diferencial de cada uno de los circuitos y se calculó el CMRR (ver figuras 4.33, 4.34, 4.35. 4.36 y tabla 4.7).

Tabla 4.7 CMRR en cada diseño del circuito para aplicaciones

Aplicación	CMRR
Monitor cardíaco reposo (en manos).	88.7dB
Medición cerca del corazón (pecho).	30.05 dB
Monitoreo durante el ejercicio.	76.83 dB
Circuito Eval-AD8232.	86.88dB

Monitor cardíaco reposo (en manos).

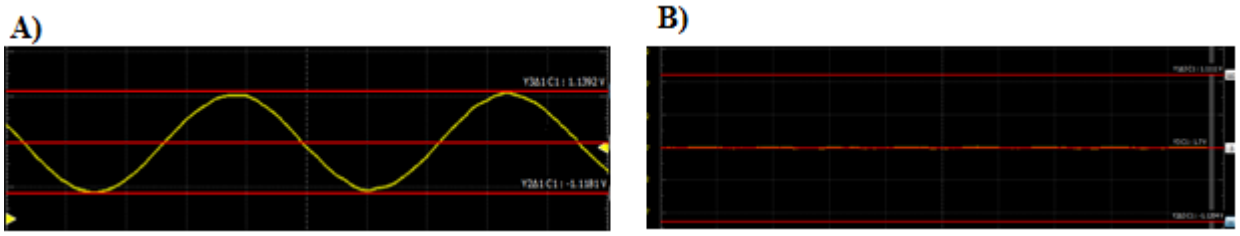


Figura 4.33 A) Ganancia diferencial =1100, Vin=1mV, Vout=1.1286V. B) Ganancia modo común=.0402, Vin=1V, Vout=40mV. Cálculo de CMRR=20Log(1100/.0402)=20Log(27,363.18)=88.7dB

Medición cerca del corazón (pecho).

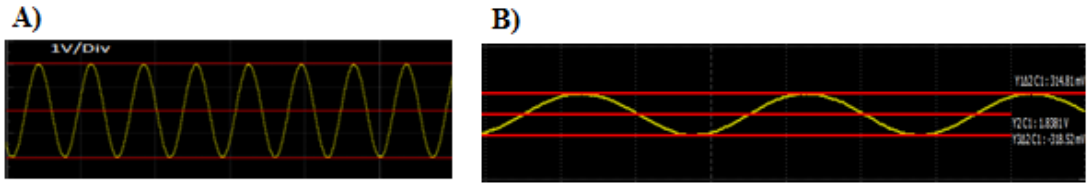


Figura 4.34 A) Ganancia diferencial =100, Vin=10mV, Vout=1.02V. **B)** Ganancia Común=3.14, Vin=100mV, Vout=314mV. Cálculo de CMRR=20Log100/.3.14=20Log(31.84)=30.05dB

Monitoreo durante el ejercicio.

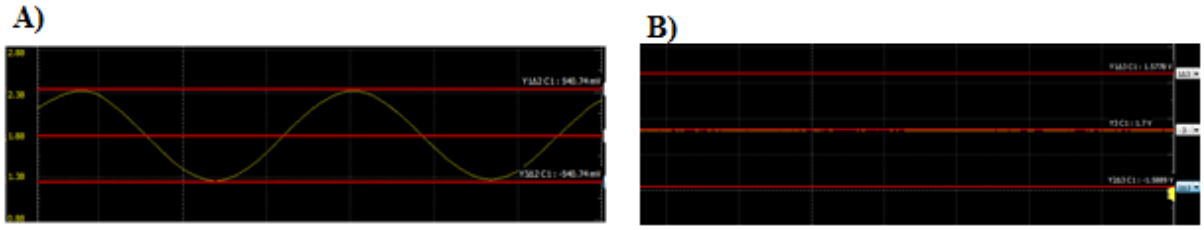


Figura 4.35 A) Ganancia diferencial =540, Vin=1mV, Vout=540mV. B) Ganancia Común=.0777, Vin=1V, Vout=78mV. Cálculo de CMRR=20Log(540/.0777)=20Log(6942.83)=76.83dB

Replica de Tablero Eval-AD8232 Analog Devices con aplicación de monitor ECG.

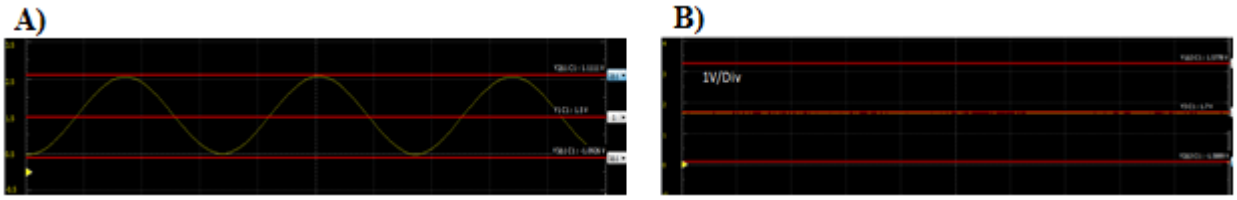


Figura 4.36 A) Ganancia diferencial =1100, Vin=1mV, Vout=1.11V. B) Ganancia Modo común =0.05, Vin=1V, Vout=50mV. Cálculo de CMRR=20Log(1100/.05)=20Log(22,000)=86.84dB

4.10 Presupuesto

Como se ha indicado a lo largo de este proyecto uno de los objetivos de utilizar una plataforma de código abierto, y componentes de fácil acceso es que el prototipo de ECG sea de bajo costo, se puede ver en la tabla 4.8 que el costo total del proyecto es de 20USD, esto sin considerar costos equipo de laboratorio y el desarrollo. Lo cual considerando el costo de un equipo de ECG para monitoreo del mercado es un costo muy accesible.

Tabla 4.8 Presupuesto para el desarrollo de un electrocardiógrafo portátil de uso personal por tiempo prolongado.

Componentes Electrónicos	**Costo (USD)**	**Equipo**	**Costo (USD)**
CI AD8232	$2.00	Estación de aire caliente	En existencia
Data logger shield	$2.00	Osciloscopio	En existencia
Arduino Nano	$3.00	Laptop	En existencia
LEDS, botones, capacitores y resistencias	$4.00	Breadboard	En existencia
Módulo bluetooth HC-06	$3.00	Generador de funciones	En existencia
Cables ECG snap	$6.00	Tablero de evaluación de Analog Devices	En existencia
TOTAL	**20 (USD)**		

4.11 Características y calidad

Esta sección resume los parámetros medidos durante este proyecto. Se puede ver en las tablas 4.9-4.12 las características generales del ECG, su funcionamiento, calidad, y seguridad establecidos en las normativas de diseño de un electrocardiógrafo de uso personal para monitoreo portátil (Ver tabla 2.1).

Tabla 4.9 Características ECG.

Aplicación	**Monitor de ritmo cardíaco de uso personal y tiempo prolongado.**
Canal	Mono canal
Derivación	Derivación I,II,III
Ancho de Banda	0.5-40Hz
Ganancia	1100
Frecuencia de muestreo	360-2100Hz (Arduino)
Voltaje de alimentación	Batería 7.4V (1300mA/4000mA)
Consumo de corriente	125mA-135mA
Conectividad	Serial, Bluetooth, SD card
Alimentación operable electrodos	1-3 mV
Corriente de fuga al paciente	1-2 μA
CMRR	88.7dB

Tabla 4.10 Características microchip AD8232

Aplicación	**Etapa inicial analógica**
Common-mode rejection ratio (CMRR)	80dB
Voltaje de alimentación	2.5-3.5V
Impedancia de entrada	10 GΩ ‖7.5 pF
Power Supply Rejection Ratio (PSRR)	90dB
Temperatura Operación.	−40°C a +85°C
Dimensiones	4×4 mm

Tabla 4.11 Requerimientos normativa ANSI/AAMI/IEC 60601-2-47:2012

Parámetro	**Prototipo ECG**	**Normativa**
Aplicación	**Monitoreo**	**Monitoreo**
Ancho de Banda	0.5-40Hz	0.5-40Hz
Ganancia	1100	1000-2000
Frecuencia de Muestreo	360-2100Hz	500-1000Hz
Rango dinámico de operación (electrodos)	0-3 mV	0-3mV
Corriente de fuga al paciente	1-2 µA	≤10 µA
CMRR	88.7dB	≥80dB
Impedancia de entrada	10MΩ	5MΩ
Ruido en la señal	20µV	≤30µV
Offset DC	±300mV	±300mV
Tiempo grabación	32 Hrs	≥24Hrs
Precisión componentes	1%	1%

Tabla 4.12 Validación de señal ECG de prototipo con estándar médico.

	Estándar [42]	Señal ECG paciente
Amplitud onda P	100-250mV	200mV
Duración onda P	90-110ms	116ms
Duración intervalo P-R	110-200ms	180ms
Amplitud complejo QRS	1.6V	1.64V
Duración complejo QRS	70-110ms	80ms
Duración intervalo Q-T	350-440ms	376ms
Duración segmento ST	50-150ms	56ms
Amplitud onda T	100-500mV	480mV

DISCUSIÓN DE RESULTADOS

Evaluación del chip AD8232.

A lo largo de esta investigación se han realizado numerosas pruebas, pero antes de la etapa de experimentación del proyecto fue necesaria la lectura e investigación sobre las nuevas tecnologías de los AFE's; se consultaron artículos, revistas científicas, y tesis para evaluar la utilidad del microchip AD8232. En base a esto se determinó que el chip AD8232 es una buena opción para el desarrollo de un sistema personal de ECG portátil que ha sido utilizado en distintas aplicaciones, como puede verse en la sección de antecedentes, y que puede integrarse en un diseño de uso prolongado. Una vez seleccionado el microchip AD8232 se adquirió el tablero de evaluación EVAL-AD8232, ofrecido por la compañía Analog Devices, y se configuró en los modos de operación de 2, y 3 electrodos. Se diseñó un circuito en breadboard con la aplicación de monitor cardíaco. Estas pruebas proporcionaron una señal ECG con morfología y duración apropiada lo cual confirmó la utilidad del chip para esta aplicación, se realizaron las mediciones con un osciloscopio RIGOL (ver Tabla 4.12).

Otro aspecto importante en la etapa de la evaluación del microchip AD8232 fue la simulación con el software Multisim y los análisis para la estabilidad del sistema, como se puede ver en la sección de resultados 4.3 el análisis Montecarlo y Worst Case ayudaron a determinar el valor de los componentes del circuito con un margen de 1% indicados en la normatividad de diseño de un ECG, y observar la variación en el offset de la señal de salida, la cual no tuvo cambios significativos. El análisis Fourier aplicado a la señal ECG determinó que dicha señal tiene más fuerza (o magnitud) dentro del rango de 0.5-70Hz, lo cual ya es bien conocido por la teoría. Al igual que la simulación del barrido de temperatura nos proporcionó un comportamiento similar al de la hoja de datos del chip AD8232 para conocer el rango de temperatura óptimo del prototipo.

Digitalización de la señal.

Para digitalizar la señal analógica se utilizó la plataforma de código abierto Arduino Nano con el microcontrolador ATmega 328. Se caracterizó su convertidor A/D y frecuencia de muestreo midiendo el intervalo de tiempo entre cada muestra del convertidor ADC. Se utilizaron sus diferentes protocolos de comunicación como serial, SPI, e I2C para

desarrollar un programa con los modos de operación: modo 1 serial, modo 2 escritura en SD card, modo 3 transmisión en bluetooth. Se caracterizó cada modo de operación y se validó el número de datos y la máxima capacidad de transferencia. Se evaluó y comparó el uso de interrupciones (ISR) contra la transferencia y uso de retardos (comando **delay**), presentando claras ventajas el método de interrupciones dando una mayor estabilidad y disminuyó el margen de error en el muestreo. Una vez caracterizado el muestreo y transmisión en cada modo de operación con interrupciones se obtuvieron las señales de ECG de paciente en formato digital. Cada modo de operación se realizó de forma individual con sus códigos correspondientes (ver en la sección de anexos). Posteriormente se fueron haciendo mejoras como agregar un menú de operación con los 4 modos en un mismo programa, utilizar LEDS para indicar el modo de operación, estado de transmisión, y agregar botones para seleccionar modo de operación e iniciar/finalizar transmisión o grabación.

Evaluación del sistema para filtros digitales.

Un aspecto importante propuesto en este trabajo fue el evaluar la capacidad de la plataforma Arduino para la aplicación de filtros digitales mediante ecuaciones de diferencias, y al mismo tiempo realizar la conversión análogo-digital y la transmisión de la información. En esta etapa se simuló una señal ideal con MATLAB y se utilizó un generador de funciones para reproducir dicha señal. Durante esta etapa de experimentación se aplicó una señal conocida y se realizó la conversión con el ADC de análogo a digital, y se usó un DAC para la conversión de digital a análogo de baja resolución y ver la señal filtrada. Un papel muy importante fue el acomodar el número de bits del convertidor y del DAC para evitar distorsión (véase corrimientos de bit en el código A1 y A2 en la sección de Anexos).

Una vez creada la señal ideal, se crearon señales de prueba que incluían diferentes tipo de ruidos; por ejemplo, se combinó con una señal de 60Hz y sus armónicos. Las diferentes señales ruidosas se utilizaron para aplicar 4 ecuaciones de diferencias programadas en el sistema (ver sección 3.6) y ver la respuesta de los filtros.

El resultado práctico de los filtros fue igual al de la simulación en MATLAB: el filtro #1 eliminó la frecuencia de 60Hz, sus armónicos y offset aunque presentó una distorsión en su respuesta debido a la fase no lineal, el filtro #2 eliminó la frecuencia de 60,

y 180Hz únicamente, el filtro #3 eliminó únicamente la componente de 60Hz, y el filtro #4 elimino todas las componentes con un α=0.1 y presento la misma distorsión en su respuesta por la fase no lineal y se aumentó el valor de α a α=0.9 y este dejó de eliminar el offset. Con esto se confirmó la utilización de filtros digitales sencillos en la plataforma de código abierto de Arduino. Una consideración importante es el tiempo que tarda en efectuar la ecuación de diferencias llegando a ser de hasta 252μs en el filtro más complejo (filtro #4).

Modos de operación, Menú de interfaz.

En esta etapa del proyecto ya se ha logrado transmitir en cada modo de operación serial, SD card, y bluetooth de forma individual una señal ECG ideal (virtual) y de un paciente. También se evaluó la capacidad del sistema para aplicar filtros digitales en cada uno de estos. Se desarrolló la interfaz de visualización para comunicación serial y bluetooth PC en el software LABVIEW, para la grabación en SD se utilizó el software Excel y THEREMINO. Para la transmisión de la señal en modo bluetooth celular se optó por utilizar la aplicación "Arduino Graphics" en un teléfono inteligente con sistema Android. En el código del prototipo (Ver anexos código A26) se desarrolló un menú de operación para seleccionar el modo, e iniciar o finalizar la transmisión o grabación de datos. Este menú posee programación estructurada con banderas, ciclos, y funciones que se mandan a ejecutar según dichas banderas con jerarquía de menú, y submenú. Se redujo el hardware a los elementos necesarios: Arduino Nano, módulo HC-06, y módulo data logger terminando el prototipo final (ver Figura 4.11).

Pruebas con data logger.

Una vez establecido el modelo final del prototipo hardware y software, se realizaron pruebas con el data logger ya que este es el modo de operación que tiene mayor utilidad al proporcionar un registro continuo de larga duración, de la actividad del corazón para posterior análisis por un especialista. Este modo de operación es más complejo ya que involucra más elementos hardware como son: el RTC, la SD y el módulo data logger, son necesarios 2 protocolos de comunicación, y un mayor flujo de información debido al registro de la hora y fecha. Las pruebas realizadas fueron: registro por tiempo prolongado, consumo de la memoria en MB del documento generado con el registro de ECG. En la tabla 4.1 se muestra el espacio de memoria consumido en las pruebas de 1,12 y 24 horas

con una señal simulada a una frecuencia de muestreo de 360Hz, y 500Hz. Para un registro de 24 horas el espacio de memoria consumido fue de hasta 991MB y el prototipo soporta una memoria de 16Gb. Lo cual sugiere cerca de 15 registros de 24 Horas (registros de 2 semanas) con una frecuencia de muestreo de 500Hz. Este experimento se realizó para someter el prototipo a una prueba real de captura de datos, verificar la estabilidad en el sistema y conocer si es que el sistema se traba, o tiene errores durante el uso prolongado del prototipo. El sistema no se trabó hasta una frecuencia de muestreo de 2000 (ver sección 3.2.4) y trabajó durante más de 24 horas, pero se pudieron apreciar en los registros de larga duración de 1 y 2 horas con una señal ideal, eventos atípicos en los que se perdían algunas muestras. Las características de este evento se detallaron en la sección 4.6.4 y se piensa que puede ser algún error de programa al ejecutar las interrupciones, o error del generador de funciones al reproducir la señal. Es necesario indagar todavía más sobre este fenómeno y realizar el análisis completo de los registros de 24 horas.

Luego de obtener los registros de larga duración con una señal virtual se realizaron pruebas de 1, y 2 horas en condiciones de reposo con paciente. Se obtuvieron los registros de la sección 4.6.3 y se realizó un análisis para verificar la presencia de eventos anormales en la grabación de la señal los cuales fueron inexistentes, aunque podían apreciarse algunos artefactos de movimiento en la señal.

Consumo de corriente.

Una vez comprobado el funcionamiento y estabilidad del prototipo por tiempo prolongado, y con pruebas de usuario, se realizaron ensayos de máxima duración por ciclo de carga de batería para esto fue necesario 2 aspectos importantes, el consumo de corriente del prototipo en cada modo de operación, (ver tabla 4.4) y la capacidad de mA de las batería. Se utilizaron baterías de polímero de litio (LiPo) de 1300 mA, y 4000mA según sus especificaciones, y al llevarlo al practica como se puede ver en la tabla 4.5 en grabación SD tiene una duración de 11.5 horas, y 31.5 horas respectivamente para cada batería. Al observar el consumo en SD es de 105-125mA y dividiendo la capacidad de las baterías se puede comprobar la capacidad de tiempo de las baterías. Se comprobó también el tiempo de uso en los otros 3 modos de operación y en modo de reposo (sin flujo de datos, *stand by*) obteniendo un tiempo mayor a 24 horas en todos los casos.

Corriente de Fuga

Para el desarrollo de un dispositivo médico uno de los aspectos de seguridad más importantes según la normativa IEC 6060-1 [35] es la corriente de fuga que puede presentarse durante el funcionamiento del equipo hacia el paciente a través de los electrodos. Debido a que es un dispositivo con alimentación interna de baterías se clasifica en un dispositivo de clase IP y tipo BF con una pieza aplicada a paciente (conductor de superficie). El límite máximo permitido de corriente de fuga es 10μA por lo que al realizar la medición como sugiere la figura 3.58 la medición que se presentó fue de 1-2μA estando dentro de la normativa de seguridad eléctrica. El dispositivo con el que se midió es un medidor de corrientes de fuga (*leakage current*): modelo UT251A el cual tiene una sensibilidad de 1μA. La corriente de fuga se midió en cada modo de operación siendo esta la misma en todos (ver tabla 4.6). El circuito diseñado en la etapa de adquisición y acondicionamiento de la señal posee resistencias de 10MΩ para protección al paciente.

Circuito ECG con filtrado para aplicaciones (hardware).

Una de las alternativas que te brinda la compañía Analog Devices es implementar filtros hardware para distintas aplicaciones que pueden aplicar los desarrolladores de equipo entre estas están: monitor cardíaco con electrodos aplicados en las manos, medición con electrodos en el pecho, aplicación de monitoreo durante el ejercicio, y una réplica del tablero de evaluación de la compañía Analog Devices (ver Figura 3.4). En la sección de resultados 4.8 se pueden ver los circuitos de cada aplicación y su respuesta. Entre las características de estos circuitos la aplicación de monitor cardíaco tiene un filtro de 0.5-40Hz, con una ganancia de 1100, con un buen CMRR de 88.7 dB estas mediciones cumplen con los requisitos mínimos en la normativa para diseño y construcción de un electrocardiógrafo con aplicación ambulatoria ANSI/AAMI/IEC 60601-2-47:2012 [18]. Con la aplicación de medición del pecho la ganancia es de 100, (el software THEREMINO no detecta de forma apropiada la señal debido a su baja amplificación y fue necesario ajustar la escala en Osciloscopio para observar la morfología de la onda ECG). Esta aplicación tiene un filtro de 7Hz y no posee un filtro bajas, debido a que la señal es más fuerte en el pecho con esta amplificación es suficiente para poder visualizar y formar la señal de ECG si se agrega otra etapa de amplificación se puede saturar el amplificador de

instrumentación. Para la aplicación de monitoreo durante el ejercicio se aplica un filtrado de 7-24Hz con una banda estrecha en su respuesta de frecuencia, lo que causa una distorsión en la forma de la onda con un CMRR de 76.83 a pesar de que la ganancia está configurada para 1100 la banda estrecha, limita un poco la amplificación de la señal lo cual se puede observar en la figura 4.31. Esta aplicación se recomienda para percepción del ritmo cardíaco debido a que en las pruebas realizadas con paciente se pudo observar que aunque había una distorsión en las ondas P y T, el complejo QRS se mantenía constante sin cambios identificando claramente el ritmo cardíaco del individuo.

La aplicación del circuito Eval-AD8232 tuvo una ganancia de 1100 con CMRR de 86.88dB y una respuesta similar a la del tablero fabricado por la compañía Analog Devices. La ganancia de cada circuito se comprobó al realizar la medición de CMRR para cada aplicación (ver sección 4.9) y puede verificarse en la tabla 4.7. Estos resultados están dentro del rango que la normativa pide como requisito que sea un CMRR ≥80dB, y en el caso de ECG Holter ≥60dB ya que existen algunos dispositivos en el mercado que tienen CMRR de 60 dB.

CONCLUSIONES

El producto final de este proyecto fue el prototipo de un electrocardiógrafo portátil para monitoreo personal y uso prolongado con aplicación de monitor de ritmo cardíaco de uso personal y tiempo prolongado. Se validaron cada una de las etapas del desarrollo: desde la evaluación del microchip AD8232, el acondicionamiento de la señal (etapas de filtrado), el desarrollo de la interfaz de visualización (LABVIEW) e integración de cuatro modos de operación en un menú. Se siguieron los requerimientos establecidos por las normas de diseño y seguridad en el desarrollo de un dispositivo médico.

El monitor ECG es de tipo mono canal de una derivación, con un ancho de banda de 0.5-40Hz y ganancia de 1100. Puede alcanzar una frecuencia de muestreo de hasta 2000Hz sin pérdida de datos, proporcionando un archivo con el registro de la señal ECG por varias horas (hasta 24 horas a 360 muestras por segundo). Tiene un CMRR de 88.78dB, una corriente de fuga de paciente de 1-2 μA, y elementos hardware de precisión con tolerancia del 1% cumpliendo con los requerimientos establecidos en la norma del estándar de diseño para sistemas electrocardiográficos de tipo ambulatorio ANSI/AAMI/IEC 60601-2-47:2012. Trabaja con un rango de operación de electrodos 1-3 mV, se utilizó una batería LiPo de 7.4V, el prototipo tiene un consumo de corriente de 125-135mA, y cuenta con múltiples protocolos de comunicación serial, SD card, y bluetooth (celular y PC).

Se evaluó también la capacidad de la plataforma Arduino para la aplicación de filtros digitales, y se determinó que cuenta con memoria suficiente para aplicar filtros digitales sencillos en electrocardiografía. El código actual ha ocupado un 60% del espacio de memoria interna del microcontrolador ATmega328 con un algoritmo modificable para las necesidades del usuario, y lo que deja un campo de mejora, o adición de funciones el cual es una de las ventajas de trabajar con un microcontrolador basado en una plataforma de código abierto.

Es importante mencionar que aunque se puede aplicar etapas de filtros digitales, se utilizó también filtros analógicos para aprovechar la facilidad que brindan los microchips AFEs para acondicionamiento de señales (determinando el ancho de banda de entrada dentro del estándar de un ECG). El diseño del circuito hardware y el ancho de banda del filtro nos permiten aplicar a diferentes aplicaciones de ámbito clínico, ambulatorio, o con fines de

monitoreo. Y con un diseño más específico aplicaciones como percepción de ritmo cardíaco durante el ejercicio, monitor de ritmo cardíaco, o adquisición de la señal cerca del pecho.

Los resultados indican que el microchip AD8232 si logró proporcionar una señal útil, para la aplicación de monitoreo por tiempo prolongado, lo que lo convierte en un buen elemento para cumplir con la función de etapa inicial analógica como parte del circuito. Otro elemento que proporcionó resultados satisfactorios es el microcontrolador ATmega328 o plataforma de Arduino de libre acceso y bajo costo, sus diversos protocolos de comunicación permitieron mantener un bajo costo de fabricación y portabilidad al reducir el número de componentes. Este es un aspecto importante a pesar de que se encuentra en etapa de prototipo el costo total fue de 20 USD, se quiere lograr que el sistema de ECG para monitoreo personal y tiempo prolongado sea accesible a un mayor sector de la población para ayudar a enfrentar el incremento de enfermedades cardiovasculares, que aunque no busca reemplazar al equipo del sector hospitalario puede ser un apoyo importante en el diagnóstico y prevención de enfermedades.

TRABAJO FUTURO

Este proyecto proporcionó un prototipo funcional de un ECG portátil para monitoreo de uso personal y bajo costo. Este proyecto aún tiene aspectos de mejora e innovación a implementar los cuales se han considerado para el doctorado, entre las cuales son:

- Detección e indicador para batería baja, ausencia SD, y memoria sin espacio disponible.
- Asegurar la capacidad de registro por 24 horas sin errores.
- Agregar alarmas según la normativa ANSI/AAMI EC13:2002
- Implementación de electrodos inalámbricos.
- Desarrollo y ensamblaje de circuito impreso con elementos esenciales.
- Desarrollo de armazón/ montura del dispositivo ECG.
- Pruebas comparativas (correlación cruzada) del dispositivo con un electrocardiógrafo de uso clínico.
- Utilizar software libre para interpretación del ECG para validar la información de los registros.
- Prueba supervisada con especialista cardiólogo.
- Desarrollar servicios informáticos (nube, base de datos en internet), aplicación para dispositivos móviles
- Pruebas con paciente por 24 horas en condiciones de actividad diaria.

REFERENCIAS

[1] Organización Mundial de la Salud, "Enfermedades Cardiovasculares", 2015[En línea]. Disponible en: http://www.who.int/es/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds) [Accedido: 18-jun-2018]

[2] Center for Connected Health Policy, "Mobile Health," San Diego, California, 2010-2018 [En línea]. Disponible en: http://www.cchpca.org/mobile-health [Accedido: 18-jun-2018]

[3] J. Cabo, "Sistema de adquisición portátil con telemetría Bluetooth para señales Biomédicas", Universidad Politécnica de Catalunya, pág. 12, 2009 [En línea]. Disponible en: https://upcommons.upc.edu/bitstream/handle/2099.1/6927/Memoria%20Holter.pdf?sequence=1 [Accedido: 18-jun-2018]

[4] A. Lama, "Historia de la medicina. Einthoven. El hombre y su invento". Revista Médica Chile. Volumen (132), pág. 261-264, 2004 [En línea]. Disponible en: http://es.slideshare.net/Drcifuentes/el-hombre-y-su-invento [Accedido: 18-jun-2018]

[5] L. J. Fesquet, "Historia de la medicina Willem Einthoven", Valencia, España, (2006) [En línea]. Disponible en: http://www.historiadelamedicina.org/einthoven.html [Accedido: 18-jun-2018]

[6] M. ZINN, "The Electrocardiogram (ECG/EKG)", Estados Unidos, 2003 [En línea]. Disponible en: http://www.oocities.org/vifibio/02ELELECTROCARDIOGRAFO.PDF [Accedido: 18-jun-2018]

[7] B. Gastelum, "Historia de la Medicina: Willem Einthoven y la Aplicación Clínica del Electrocardiograma". Volumen (2), pág. 104-107, 2008 [En línea]. Disponible en: http://hgculiacan.com/revistahgc/archivos/Rev6%20Historia%20de%20la%20Medicina.pdf [Accedido: 18-jun-2018]

[8] Secretaria de salud, CENETEC, "Guía tecnológica de Electrocardiógrafo", México, 2006 [En línea]. Disponible en: http://www.cenetec.salud.gob.mx/descargas/biomedica/guias_tecnologicas/17gt_electrocardiografos.pdf [Accedido: 18-jun-2018]

[9] Ilustración de electrocardiógrafo CONTEC ECG 600G, Figura 1.9, 2018, [En línea]. Disponible en: http://www.equimeed.com/tienda/home/16-electrocardiografo-contec-ecg-600g-6-canales-digital-touch.html [Accedido: 18-jun-2018]

[10] Centro Medico Escuela, "Historia de Electrocardiógrafo", Buenos Aires, Argentina, 2018, "[En línea]. Disponible en:" http://www.electrocardiograma.org/historia-del-electrocardiografo.html [Accedido: 18-jun-2018]

[11] L. Kostas, M. Ignaszwski, I. Macdonald, Figura 1.10 "Primer Electrocardiógrafo Holter", Medical Journal, Volumen 56 no.2, pag 86-89, 2014.

[12] Ilustración de electrocardiógrafo Holter, Figura 1.11 "Spacelabs healtcare Manual Holter lifecard-CF", 2018 [En línea]. Disponible en: http://www.spacelabshealthcare.com/wp-content/uploads/2013/08/Lifecard-CF_English_000-1047-Rev-B1.pdf [Accedido: 18-jun-2018]

[13] A. M. Mendiguren, "Un electrocardiógrafo inteligente de bajo coste" (Tesis de pregrado) Universidad del País Vasco, España, 2014.

[14] L. Xu, "Wireless Hybrid Bio-Sensing with Mobile based Monitoring System", (Tesis de Maestría en ciencias) Stockholm, Sweden, 2013.

[15] T. Lu, P. Liu, X. Gao, Q. Lu, "A Portable ECG Monitor with Low Power Consumption and Small Size Based on AD8232 Chip". Applied Mechanics and Materials. Volumen: 513-517 pág: 2884-2887, 2014.

[16] F. Malenak, "Mobile System for Monitoring Sports", Departament of Biomedical Eginerring, Brno University of Technology, Czech Republic, 2014.

[17] Z. Shen, O. He, Y. Li, "A Health Shirt with ECG Real-time Display on Android Platform. International Journal of Information Technology", Volumen (20) no.2, 2014.

[18] L. Nguyen, R. Perry L. Seneres, N. Seyedmahmoud, "Analog Integrated Circuit Applications", Bachelor Thesis, Faculty of the Worcester Polytechnic Institute, Massachusetts, USA, 2013.

[19] R. Chaudhri, "Extending Sensing Capabilities and Modalities of Mobile Devices", Ph. D. thesis, University of Washington, Washington, USA, 2014.

[20] M. Wildan, H. Zakaria, R. Mengko, "Design of ECG Homecare: 12-Lead ECG Adquisition using Single Channel ECG Device Developed on Ad8232 Analog Front End" en The 5th International Conference on Electrical Engineering and Informatics 2015, Bali, Indonesia, 2015, pp. 371-376.

[21] *Particular requirements for the basic safety and essential performance of ambulatory electrocardiographic systems International Standard*, ANSI/AAMI/IEC 60601-2-47:2012 [En línea]. Disponible en: https://my.aami.org/aamiresources/previewfiles/601247_1701_preview.pdf [Accedido: 18-jun-2018]

[22] *Disposable electrodes,* ANSI/AAMI EC12:2000/(R)2010, [En línea]. Disponible en: https://webstore.ansi.org/Previews/PREVIEW_ANSI+AAMI+EC12-2000+(R+2010).pdf [Accedido: 18-jun-2018]

[23] S. Mulroney, y A. Myers, *Netter's Essential Physiology,* Washington, DC, Elsevier, 2011.

[24] H. Guyton, *Tratado de fisiología médica*, España, Elsevier, 2011.

[25] L. Sherwood, *Fisiología Humana*, México, CENGAGE LEARNING, 2011.

[26] A. Bayés de Luna, *Electrocardiografía Clínica*, Barcelona, España, Publicaciones Permanyer, 2012.

[27] A. Chen, "Medicine Plus", Maryland Estados Unidos, Biblioteca Nacional de Medicina de USA, 2016 [En línea]. Disponible en: https://medlineplus.gov/spanish/ency/article/003868.htm [Accedido: 18-jun-2018]

[28] J. Frye, J. Gibson, M. Sun, C. Wang, "Android Electro Cardio Monitor", University of Central Florida, Orlando, Florida, 2016 [En línea]. Disponible en: http://slideplayer.com/slide/6147059/ [Accedido: 18-jun-2018]

[29] J. P. Mompin, *Introducción a la Bioingeniería capítulo 4 Transductores bioeléctricas*, Barcelona, España, Marcombo, 1998.

[30] Ilustración de electrodos Grupo Medicastore, Figura 2.16, 2018[En línea]. Disponible en: https://grupomedicastore.mx/77-venta-de-pinzas-perillas-broches-y-electrodos-para-ecg-en-mexico [Accedido: 18-jun-2018]

[31] C. Alva, W. Reaño, J. Castillo, "Diseño y construcción de un electrocardiógrafo de bajo costo" (Tesis de pregrado), Universidad Ricardo Palma, Lima-PERÚ, 2011 [En línea]. Disponible en: http://www.urp.edu.pe/pdf/ingenieria/electronica/CIR-11_Electrocardiografo_de_Bajo_Costo.pdf [Accedido: 18-jun-2018]

[32] Texas Instrument, Filter PRO (versión 1.03.3) [Software], 2003 [En línea]. Disponible en: http://descargar.cnet.com/FilterPro/3000-6677_4-10215351.html [Accedido: 18-jun-2018]

[33] J. I. Huircán, "Conversores Análogo-Digital y Digital-Análogo: Conceptos Básicos", [sin fecha] [En línea]. Disponible en:" http://quidel.inele.ufro.cl/~jhuircan/PDF_CTOSII/ad03.pdf [Accedido: 18-jun-2018]

[34] Ilustración de Conversión analógico a digital, Figura 2.24, 2008 [En línea]. Disponible en: http://frecuenciafundamental.blogspot.mx/2008/06/glosario-convertidores.html [Accedido: 18-jun-2018]

[35] Electronic Design, "Competing ECG AFEs Reveal Chipmakers' New Business Paradigms", 2012 [En línea]. Disponible en: http://electronicdesign.com/analog/afes-target-ecgeeg-measurements [Accedido: 18-jun-2018]

[36] Electronic Design, "Electronic Design Announces 2012 Best Electronic Design Award Winners", 2012 [En línea]. Disponible en: http://electronicdesign.com/content/electronic-design-announces-2012-best-electronic-design-award-winners [Accedido: 18-jun-2018]

[37] Analog Devices, "Single-Lead Heart Rate Monitor Front End" [Archivo PDF], Massachusetts, Estados Unidos, 2012-2017 [En línea]. Disponible en: http://www.analog.com/media/en/technical-documentation/data-sheets/AD8232.pdf [Accedido: 18-jun-2018]

[38] *Medical electrical equipment general requirements for basic safety and essential performance*, IEC 60601-1. [En línea]. Disponible en:"http://www.ele.uri.edu/courses/bme484/iec60601-1ed3.0_parts.pdf [Accedido: 18-jun-2018]

[39] *Particular requirements for the basic safety and essential performance of electrocardiographic monitoring equipment*, ANSI/AAMI/IEC 60601-2-27:2011. [En línea]. Disponible en: https://my.aami.org/aamiresources/previewfiles/6060102027reaffPreview.pdf [Accedido: 18-jun-2018]

[40] *Particular requirements for the basic safety and essential performance of electrocardiographs*. International Standard ANSI/AAMI/IEC 60601-2-25:2011. [En línea]. Disponible en: https://my.aami.org/aamiresources/previewfiles/601225_1701_reaff_preview.pdf [Accedido: 18-jun-2018]

[41] Medical Device Test Equipment Education Qualification. "Filtros de ECG", 2017 [En línea]. Disponible en: http://www.medteq.net/article/2017/4/1/iec-60601-2-27-update-2005-to-2011-edition [Accedido: 18-jun-2018]

[42] R. Bistel, A. Fajardo, "Diseño de un Sistema de Adquisición y Procesamiento de la Señal de ECG basado en Instrumentación Virtual". Revista de Ingeniería Electrónica Automática y Comunicaciones (RIELAC). Vol.XXXVI. p.17-30, 2015 [En línea]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_abstract&pid=S1815-59282015000100002 [Accedido: 18-jun-2018]

[43] M. Wildan., G. Hasballah, R. Mengko, "Design of ECG Homecare: 12-Lead ECG Acquisition using Single Channel ECG Device Developed on AD8232 Analog Front End", Revista IEEE, Volume (978-1), pág: 371-376, 2015.

[44] Fluke Biomedical, "Introducción a las pruebas de seguridad eléctrica parte I", [Archivo PDF], Ohio, USA, 2014 [En línea]. Disponible en: http://support.fluke.com/biomedical/download/asset/9460539_eng_a_w.pdf [Accedido: 18-jun-2018]

[45] J. Moore, "Fiability vs. Robustness", Magazine Printed Circuit Design & FAB/Circuits Assembly, Pág: 22-23, 2016.

[46] MEDTEC, Medical Device Test Equipment Education Qualification. "Guía de actualización IEC 60601-2-25 (2005 a 2011)", 2017[En línea]. Disponible en: http://www.medteq.net/article/iec-60601-2-25-update-guide-2005-to-2011-edition [Accedido: 18-jun-2018]

[47] MEDTEC. Medical Device Test Equipment Education Qualification. "Guía de actualización IEC 60601-2-27 (2005 a 2011)", 2017[En línea]. Disponible en: http://www.medteq.net/article/2017/4/1/iec-60601-2-27-update-2005-to-2011-edition [Accedido: 18-jun-2018]

[48] V. Ripoll, F. Peris, M. Nicoles, C. Pérez, *Simulación de circuitos lineales, Ejercicios con Pspice*. Málaga, España, Editorial Club Universitario (ECU), 2003

[49] C. Gómez, A. Castillo, A. Meoño, "ARDUINO COMO UNA HERRAMIENTA PARA MEJORAR EL PROCESO DE ENSEÑANZA – APRENDIZAJE DE LAS CIENCIAS, TECNOLOGÍAS E INGENIERÍAS EN LA UNIVERSIDAD POLITÉCNICA DE TAPACHULA" In Intitución Universitaria Salazar y Herrera (IUSH)", Medellín, Colombia, 2015 [En línea]. Disponible en: http://revistas.proeditio.com/iush/quid/article/view/119 [Accedido: 18-jun-2018]

[50] M. Barrón, M. Vázquez, R. Arteaga, R. Hernández, "Sistema de adquisición de datos de bajo costo con la plataforma arduino". *Revista Mexicana C.A.Volumen* (8 núm. 1). pp 1-12, 2017.

[51] ARDUINO 1.8.5 (versión IDE 1.8.5) [Software], (2016) "[En línea]. Disponible en: https://www.arduino.cc/en/main/software [Accedido: 18-jun-2018]

[52] National Instruments,"Comunicación Serial Conceptos generales. National Instruments", 2006 [En línea]. Disponible en: http://digital.ni.com/public.nsf/allkb/039001258CEF8FB686256E0F005888D1 [Accedido: 18-jun-2018]

[53] A. O. Chase, M. H. Sampaio, J. F. Almeida, S. Brito, "Data acquisition system: an approach to the Amazonian environment. IEEE" Ame lat .Volumen (10-2) pp.1616-1621, 2012.

[54] J. Peralta, L. Ibaez, M. Carabajal, M. Rotela, P. Gomez, "Administrador de File System FAT16 y FAT32".San Justo, Argentina, 2011[En línea]. Disponible en: http://www.so-unlam.com.ar/informes/informeFAT2011.pdf [Accedido: 18-jun-2018]

[55] PROMETEC, "Tutorial Arduino SPI", PROMETEC, [sin fecha] [En línea]. Disponible en: http://www.electroensaimada.com/spi.html [Accedido: 18-jun-2018]

[56] E. Coquet, "El bus I2C", Buenos Aires, Argentina, Comunidad de electrónicos, [sin fecha] [En línea]. Disponible en: https://www.comunidadelectronicos.com/articulos/i2c.htm [Accedido: 18-jun-2018]

[57] L. Velázquez, "Bluetooth más que una conexión inalámbrica". *UNAM Revista en línea*. *Volumen* (Año 7 Núm. 74. Mes Noviembre), 2008.

[58] NATIONAL INSTRUMENTS, "Biomedical Application: Multisim Simulation with an ECG Amplifier", [Software], 2011 "[En línea]. Disponible en: http://www.ni.com/example/30925/en/ [Accedido: 18-jun-2018]

[59] C. Juan, "Adquisición de datos con Arduino I: Tiempo de muestreo y resolución" [Mensaje en un Blog] Booleanbite, 2015[En línea]. Disponible en: http://booleanbite.com/web/adquisicion-de-datos-con-arduino-i-tiempo-de-muestreo-y-resolucion/ [Accedido: 18-jun-2018]

[60] Guangzhou HC information technology, "HC-06 product data sheet Guangzhou", China, [sin fecha] [En línea]. Disponible en: https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf [Accedido: 18-jun-2018]

[61] National Instrument, "LABVIEW User Manual", Austin, Texas. National Instrument Corporation, 2000 [En línea]. Disponible en: http://www.ni.com/pdf/manuals/320999c.pdf [Accedido: 18-jun-2018]

[62] THEREMINO, (THEREMINOECG), [Software], 2017 [En línea]. Disponible en:http://www.theremino.com/es/downloads/biometry [Accedido: 18-jun-2018]

[63] Michigan Tech, Department of Electrical and Computer Engineering, Ashok Ambardar ADSP toolbox, 2013[en línea] disponible en : http://www.ece.mtu.edu/faculty/akambard/book/update.html [Accedido: 25-jul-2018]

[64] R. Strum, D. Kirk, *First Principles of Discrete Systems and Digital Signal Processing,* Michigan, USA, Prentice Hall, 1998.

[65] A. Ambardar, *Analog and Digital Signal processing*. Michigan, USA, An International Thomson Publishing Company, 1998.

[66] M. Hayes, *Schaum's Outline of Digital Signal Processing*. Georgia, USA, McGraw-Hill Book Companies, 1998.

ANEXOS

Códigos

A1. Método #1 Muestreo con comando “delay” nulo para medición de velocidad máxima del convertidor A/D.

```
volatile int sensor;//Declarar variable del sensor.
void setup() {

// declarar Port D como salida
DDRD=0xFF;

void loop()
{int sensor = analogRead(A0)>> 2;;  // (corrimiento para usar DAC de 8 bits)
PORTD = (sensor);
//delay(0);
 }          // mandar el valor leído a port D
```

A2. Método #2 muestreo por interrupciones para medir el tiempo de respuesta de E/S digitales.

```
#include <TimerOne.h>
volatile int sensor;//Declarar variable del sensor.
void setup() {
// declarar Port D como salida
DDRD=0xFF;
   Timer1.initialize(2777); // 1/360= .0027777777
  Timer1.attachInterrupt (timerIsr); // attach service routine}
void loop() {}
// Rutina de interrupción
void timerIsr()
{sensor= analogRead(A0) >> 2;; // (corrimiento para usar DAC de 8 bits)
PORTD = sensor;} // imprime valores del pin  A0 a port D
```

A3. Transmisión serial y medición de tiempo con comando “millis”.

```
void setup()
{Serial.begin(115200);}

void loop()
{int time1=millis()/1000;
 int sensor = analogRead(A0);
Serial.print(sensor);
Serial.print("   ");
Serial.println(time1);
//delay(0);
}
```

A4. Transmisión serial con botón transmitir/stop.

```
#include <RTClib.h>
RTC_DS1307 RTC;
#include <SoftwareSerial.h>
#include <TimerOne.h>  // declarar  uso de librería
SoftwareSerial BT(8,9);//RX,TX
#include <SPI.h>
#include <SD.h>
File myFile;
//#include <DS3231.h>
#include <Wire.h>
//DS3231 Clock;
byte year, month, date, DoW, hour, minute, second;
char linea[15];
char voltaje[4];

// --- Mapeamento de Hardware ---
#define butDown 7
#define butP    5
#define butM    6
#define LED1    A1
#define LED2    A2
#define CS      10
#define LED3    4
```

```
#define LED4    3
#define LED5    2
#define LED6    A3

// Funciones  Menu
void changeMenu();
void dispMenu();
void Modo1();
void Modo2();
void Modo3();
void menu4();

// Variables Globales
char menu = 0x01;  //Variable para seleccionar el menú
char set1 = 0x00; //Control de los LEDS
boolean t_butDown, t_butP, t_butM; //Banderas para almacenar el estado de los botones


void setup()
{ Wire.begin();
 RTC.begin();
  pinMode(7, INPUT_PULLUP);
  pinMode(5, INPUT_PULLUP);
  pinMode(6, INPUT_PULLUP);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(CS, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
  pinMode(LED5, OUTPUT);
  pinMode(LED6, OUTPUT);
  t_butDown = 0x00;//Bandera down 0
  t_butP    = 0x00;//Bandera grabar 0
  t_butM    = 0x00; //Bandera stop 0
  } //end setup
void loop()
{
   changeMenu();
   dispMenu();
}
//FN del menu
void changeMenu() //Cambia menu Actual
{
if(!digitalRead(butDown)) t_butDown = 0x01; //Down presionado? activar Bandera
if(digitalRead(butDown) && t_butDown)  //Up Suelto y bandera activa?
{ t_butDown = 0x00;  //Limpia bandera
 menu++;

if(menu > 0x01) menu = 0x01; //Si menu es mayor a 4 regresar menu a 1.
} //end butUp
} //end changeMenu

void dispMenu() //Mostrar menu actual
{
switch(menu) //(cambiar menu) (Control de la variable de menú)
{
case 0x01:  //Caso 1
Modo1(); //llama Modo 1 Serial port COM
digitalWrite(LED3, HIGH);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
break;
} //end Cambiar menu
} //end Mostrar Menu

void Modo1() //Modo 1 Serial port
{
if(!digitalRead(butP))    t_butP    = 0x01; // grabar presionado? Activar bandera
if(!digitalRead(butM))    t_butM    = 0x01; //stop presionado? Activar bandera
if(digitalRead(butP) && t_butP) //Botón suelto y bandera activa?
{
t_butP = 0x00; //limpiar bandera
Puertoserie(); //Llama Función Puerto Serie
digitalWrite(LED1, HIGH);//Prende LED 1 Grabar
digitalWrite(LED2, LOW);//Apaga LED 2 Finalizar

   } //end butP

   if(digitalRead(butM) && t_butM) //Botón stop suelto y bandera activa?
{
t_butM = 0x00; //Limpia bandera
```

```
Serial.end(); // Finaliza Comunicación Serial
digitalWrite(LED1, LOW); //Apaga LED 1 Grabar
digitalWrite(LED2, HIGH); //Prende LED 2 Finalizar
delay(1500); // Espera 1.5 segundos para  iniciar nuevo modo.
digitalWrite(LED2, LOW); //Apaga LED 2 Para iniciar nuevo modo
} //end butM
} //end Modo1

void Puertoserie() {
  Serial.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr);
}
void timerIsr()
{volatile int sensor= analogRead(A0);
Serial.println(sensor);
//Serial.print("v");
}
```

Códigos para data logger.

Reloj de Tiempo Real (RTC).

A5. Establecer hora en RTC (Manual).

```
#include <DS3231.h>
#include <Wire.h>
DS3231 Clock;
void setup() {
  Wire.begin();// Se inicia la interfaz I2c
  Serial.begin(9600); // Se inicia la Comunicación Serial
  Clock.setClockMode(false); //Se establece el modo horario en 12 horas (false = 24h)
  Clock.setYear((byte)(17)); //Se establece el año
  Clock.setMonth((byte)03);  //Mes
  //Clock.setDoW((byte)dia);  //Dia de la semana (no lo estoy considerando)
  Clock.setDate((byte)4); //Día
  Clock.setHour((byte)9); //Hora
  Clock.setMinute((byte)02); //Minutos
  Clock.setSecond((byte)0);} //Segundos
 void loop() {}
```

A6. Establecer hora en RTC Automático (Desde la computadora).

```
#include <Wire.h>
#include "RTClib.h"
RTC_DS1307 RTC;
void setup () {
Wire.begin(); // Inicia el puerto I2C
RTC.begin(); // Inicia la comunicación con el RTC
RTC.adjust(DateTime(__DATE__, __TIME__)); // Establece la fecha y hora
Serial.begin(9600); // Establece la velocidad de datos del puerto serie
}
void loop () {}
```

A7. Leer hora y fecha del RTC.

```
#include <DS3231.h>
#include <Wire.h>
DS3231 Clock;
byte year, month, date, DoW, hour, minute, second;
void setup() {
 Wire.begin();// Start the I2C interface
 Serial.begin(9600);} // Start the serial interface
 void loop() {
 Clock.getTime(year, month, date, DoW, hour, minute, second);
         Serial.print(date, DEC);
        Serial.print("/");
  Serial.print(month, DEC);
        Serial.print("/");
  Serial.println(year, DEC);
    Serial.print(hour, DEC);
        Serial.print(":");
  Serial.print(minute, DEC);
        Serial.print(":");
  Serial.println(second, DEC);
   delay(5000);}
```

A8. Data logger con método delay

```
#include <SPI.h>
#include <SD.h>
const int chipSelect=10;
File myFile;
#include <DS3231.h>
#include <Wire.h>
DS3231 Clock;
byte year, month, date, DoW, hour, minute, second;

void setup()
{Serial.begin(9600);
while(!Serial)
{}
Serial.print ("Inicializando SD card...");
pinMode(chipSelect,OUTPUT);
if(!SD.begin(chipSelect)) {
   Serial.print ("Fallo Lectura de SD card...");
   return;}
  Serial.println (" SD card inicializada ok");}

 void loop() {
 myFile = SD.open("ECGfile.txt", FILE_WRITE);//abrimos  el archivo
 if (myFile) {
 int sensor = analogRead(A0);
 myFile.print("");
 myFile.print(millis()/1000);
 myFile.print(",");
 myFile.print(sensor);
 myFile.print(",");
 myFile.print(date, DEC);
 myFile.print("/");
 myFile.print(month, DEC);
 myFile.print("/");
 myFile.print(year, DEC);
 myFile.print(",");
 myFile.print(hour, DEC);
 myFile.print(":");
 myFile.print(minute, DEC);
 myFile.print(":");
 myFile.println(second, DEC);
 myFile.close(); //cerramos el archivo
  } else {
    Serial.println("Error al abrir el archivo");
  }}
```

A9. Data logger con método de interrupciones y botón transmitir/stop.

```
#include <SPI.h>
#include <SD.h>
File myFile;
#include <DS3231.h>
#include <Wire.h>
#include <TimerOne.h>
DS3231 Clock;
byte year, month, date, DoW, hour, minute, second;
char linea[64]; //
char voltaje[10]; // Se usa para obtener el resultado de la conversión float a texto

#define LED1    A1
#define LED2    A2
#define chipSelect      10
#define LED3    4
#define LED4    3
#define LED5    2
#define LED6    A3

void setup()
{
    TIMSK0 = 0;
  Wire.begin();
  pinMode(7, INPUT_PULLUP);
  pinMode(6, INPUT_PULLUP);

  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(chipSelect, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
```

```
  pinMode(LED5, OUTPUT);
  pinMode(LED6, OUTPUT);

digitalWrite(LED1, LOW);//Prende LED 1 Grabar
digitalWrite(LED2, LOW);//Apaga LED 2 Finalizar

digitalWrite(LED3, HIGH);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar

  if (!SD.begin(chipSelect))
  {
digitalWrite(LED1, LOW);//Prende LED 1 Grabar
digitalWrite(LED2, LOW);//Apaga LED 2 Finalizar

digitalWrite(LED3, LOW);//Prende LED 1 Grabar
digitalWrite(LED4, HIGH);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
    return;
  }

digitalWrite(LED1, LOW);//Prende LED 1 Grabar
digitalWrite(LED2, LOW);//Apaga LED 2 Finalizar

digitalWrite(LED3, HIGH);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar

  Timer1.initialize(2777); // 1/500= .002 =2000 micros para 500 muestras
}
void loop()
{
  if (!digitalRead(7)) grabar();
   //Clock.getTime(year, month, date, DoW, hour, minute, second);
}

void grabar() {
  while (!digitalRead(7)); // Espera a que el botón se suelte
  myFile = SD.open("datalog1.txt", FILE_WRITE);//abrimos  el archivo
  if (!myFile) {
  return;}
digitalWrite(LED1, HIGH);//Prende LED 1 Grabar
digitalWrite(LED2, LOW);//Apaga LED 2 Finalizar

digitalWrite(LED3, HIGH);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar

  Clock.getTime(year, month, date, DoW, hour, minute, second);
  myFile.write(linea, sprintf(linea, "Iniciado el %02d/%02d/20%02d, a las %02d:%02d:%02d\r\n", date,
month, year, hour, minute, second)); //
   Timer1.attachInterrupt (timerIsr); // attach service
  while (digitalRead(7)){
     unsigned long tAnterior = 0;
    unsigned long tActual = millis();
    if (tActual >= tAnterior) { // Pide información nueva al RTC cada segundo
      tAnterior = tActual + 1000;
      Clock.getTime(year, month, date, DoW, hour, minute, second);}
  }
  Timer1.detachInterrupt(); // detach service
  while (!digitalRead(7)); // Espera a que el botón se suelte

  Clock.getTime(year, month, date, DoW, hour, minute, second);
  myFile.write(linea, sprintf(linea, "Finalizado el %02d/%02d/20%02d, a las %02d:%02d:%02d\r\n", date,
month, year, hour, minute, second)); // Pie
  myFile.close(); //cerramos el archivo
digitalWrite(LED1, LOW);//Prende LED 1 Grabar
digitalWrite(LED2, HIGH);//Apaga LED 2 Finalizar

digitalWrite(LED3, LOW);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, HIGH);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
}
void timerIsr()
{
dtostrf(analogRead(A0) * 1, 1, 0, voltaje);
```

```
  myFile.write(linea, sprintf(linea, "%s\t%02d/%02d/20%02d\t%02d:%02d:%02d\r\n", voltaje, date, month,
year, hour, minute, second));  }
 }
```

A10. Configurar bluetooth (nombre y password)

```
const int LED=13;
const int BTPWR=11;
char nombreBT[10]="ECGsignal";
char velocidad= '4'; // velocidad 1.- 1200, 2.- 2400, 3.-4800, 4.-9600, 5.- 19200, 6.- 38400, 7.- 57600,
8.-115200
char pin[5]="0000";

void setup() {
pinMode(LED,OUTPUT);
pinMode(BTPWR,HIGH);
digitalWrite(LED,LOW);
digitalWrite(BTPWR,HIGH);
Serial.begin(9600);
Serial.print("AT");
delay(3000);
Serial.print("AT+NAME");
Serial.print(nombreBT);
delay(3000);
Serial.print("AT+BAUD");
Serial.print(velocidad);
delay(3000);
Serial.print("AT+PIN");
Serial.print(pin);
delay(3000);
digitalWrite(LED,HIGH);
}
void loop() {}
}
```

A11. Transmitir bluetooth en celular con botón transmitir/stop

```
#include <RTClib.h>
RTC_DS1307 RTC;
#include <SoftwareSerial.h>
#include <TimerOne.h>  // declarar  uso de libreria
SoftwareSerial BT(8,9);//RX,TX
#include <SPI.h>
#include <SD.h>
File myFile;
#include <Wire.h>
byte year, month, date, DoW, hour, minute, second;
char linea[15];
char voltaje[4];

// --- Mapeamento de Hardware ---
#define butDown 7
#define butP    5
#define butM    6
#define LED1    A1
#define LED2    A2
#define CS      10
#define LED3    4
#define LED4    3
#define LED5    2
#define LED6    A3

// Funciones  Menu
void changeMenu();
void dispMenu();
void Modo3();

// Variables Globales
char menu = 0x01;  //Variable para seleccionar el menú
char set1 = 0x00; //Control de los LEDS
boolean t_butDown, t_butP, t_butM; //Banderas para almacenar el estado de los botones

void setup()
{ Wire.begin();
 RTC.begin();
  pinMode(7, INPUT_PULLUP);
  pinMode(5, INPUT_PULLUP);
  pinMode(6, INPUT_PULLUP);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(CS, OUTPUT);
```

```
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
  pinMode(LED5, OUTPUT);
  pinMode(LED6, OUTPUT);
  t_butDown = 0x00;//Bandera down 0
  t_butP    = 0x00;//Bandera grabar 0
  t_butM    = 0x00; //Bandera stop 0
  } //end setup

void loop(){
   changeMenu();
   dispMenu();
}
//FN del menu
void changeMenu() //Cambia menu Actual
{
if(!digitalRead(butDown)) t_butDown = 0x01; //Down presionado? activar Bandera
if(digitalRead(butDown) && t_butDown)  //Up Suelto y bandera activa?
{ t_butDown = 0x00;  //Limpia bandera
 menu++;
if(menu > 0x01) menu = 0x01; //Si menu es mayor a 4 regresar menu a 1.
} //end butUp
} //end changeMenu

void dispMenu() //Mostrar menu actual
{switch(menu) //(cambiar menu) (Control de la variable de menú)
{
case 0x01:  //Caso 3
Modo3();  //llama Modo 3 Bluetooth Celular
digitalWrite(LED3, LOW);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, HIGH);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
break;
} //end Cambiar menu
} //end Mostrar Menu

void Modo3()  //Modo 3 cel Bluetooth
{if(!digitalRead(butP))    t_butP    = 0x01;
if(!digitalRead(butM))    t_butM    = 0x01;
if(digitalRead(butP) && t_butP)
{t_butP = 0x00;
BluetoothCel();
digitalWrite(LED1, HIGH);
digitalWrite(LED2, LOW);
} //end butP

if(digitalRead(butM) && t_butM)
{t_butM = 0x00;
BT0CEL();
digitalWrite(LED1, LOW);
digitalWrite(LED2, HIGH);
delay(1500);
digitalWrite(LED2, LOW);
} //end butM
} //end Modo3

void BluetoothCel() {
BT.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr1);
  }
void timerIsr1()
{volatile int sensor= analogRead(A0);
int sensor;
BT.print("E");
BT.println(sensor);
}
void BT0CEL(){
BT.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr5);
}
void timerIsr5()
{ volatile int sensor=0;
BT.print("E");
BT.println(sensor);
}
```

A12. Transmitir bluetooth a PC con botón transmitir/stop

```
#include <RTClib.h>
RTC_DS1307 RTC;
#include <SoftwareSerial.h>
#include <TimerOne.h>  // declarar  uso de libreria
SoftwareSerial BT(8,9);//RX,TX
#include <SPI.h>
#include <SD.h>
File myFile;
#include <Wire.h>
byte year, month, date, DoW, hour, minute, second;
char linea[15];
char voltaje[4];

// --- Mapeamento de Hardware ---
#define butDown 7
#define butP    5
#define butM    6
#define LED1    A1
#define LED2    A2
#define CS      10
#define LED3    4
#define LED4    3
#define LED5    2
#define LED6    A3

// Funciones  Menu
void changeMenu();
void dispMenu();
void menu4();
// Variables Globales
char menu = 0x01;  //Variable para seleccionar el menú
char set1 = 0x00; //Control de los LEDS
boolean t_butDown, t_butP, t_butM; //Banderas para almacenar el estado de los botones

void setup()
{ Wire.begin();
 RTC.begin();
  pinMode(7, INPUT_PULLUP);
  pinMode(5, INPUT_PULLUP);
  pinMode(6, INPUT_PULLUP);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(CS, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
  pinMode(LED5, OUTPUT);
  pinMode(LED6, OUTPUT);
  t_butDown = 0x00;//Bandera down 0
  t_butP    = 0x00;//Bandera grabar 0
  t_butM    = 0x00; //Bandera stop 0
  } //end setup

void loop()
{changeMenu();
 dispMenu();
}
//FN del menu
void changeMenu() //Cambia menu Actual
{if(!digitalRead(butDown)) t_butDown = 0x01; //Down presionado? activar Bandera
if(digitalRead(butDown) && t_butDown)  //Up Suelto y bandera activa?
{ t_butDown = 0x00;  //Limpia bandera
 menu++;
 if(menu > 0x01) menu = 0x01; //Si menu es mayor a 4 regresar menu a 1.
} //end butUp
} //end changeMenu
void dispMenu() //Mostrar menu actual
{switch(menu) //(cambiar menu) (Control de la variable de menú)
{case 0x01: //Caso 4
Modo4(); //Llama modo 4 Bluetooth PC
digitalWrite(LED3, LOW);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, HIGH);//Apaga LED 2 Finalizar
break;
} //end Cambiar menu
} //end Mostrar Menu

void Modo4() //Transmisión Bluetooth PC
{if(!digitalRead(butP))    t_butP    = 0x01;
```

```
 if(!digitalRead(butM))    t_butM    = 0x01;
 if(digitalRead(butP) && t_butP)
 {t_butP = 0x00;
 BluetoothPC();
 digitalWrite(LED1, HIGH);
 digitalWrite(LED2, LOW);
 } //end butP

if(digitalRead(butM) && t_butM)
{t_butM = 0x00;
 BT0PC();
 digitalWrite(LED1, LOW);
 digitalWrite(LED2, HIGH);
 delay(1500);
 digitalWrite(LED2, LOW);
} //end butM
} //end Modo4

void BluetoothPC(){
BT.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr3);}
void timerIsr3()
{ volatile int sensor= analogRead(A0);
BT.print("v");
BT.print(sensor);}

void BT0PC(){
BT.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr4);}
//Rutina de interrupcion
void timerIsr4()
{ volatile int sensor=0;
BT.print("v");
BT.print(sensor);}
```

A13. Implementación buffer circular

```
#include <TimerOne.h>
volatile int sensor;
volatile float x[7]={0,0,0,0,0,0,0};
volatile float y;
void setup() {
   Timer1.initialize(1000000); //llamar la interrupcion cada 1 segundo
  Timer1.attachInterrupt (timerIsr); // manda a llamar la Rutina de interrupcion Timer ISR
  Serial.begin(9600);} // inicializa el Puerto Serial
void loop() {}

//Rutina de interrupcion
void timerIsr()
{int sensor = analogRead(A0);
x[0]=((sensor)/4);
y = x[0]-x[6]; // Se realiza la operación del filtro
 Serial.println(y);      // Imprimimos los valores del buffer
 Serial.print("x[6]=");Serial.print(x[6]);
 Serial.print("x[5]=");Serial.print(x[5]);
 Serial.print(" ");
 Serial.print("x[4]=");Serial.print(x[4]);
 Serial.print(" ");
 Serial.print("x[3]=");Serial.print(x[3]);
 Serial.print(" ");
 Serial.print("x[2]=");Serial.print(x[2]);
 Serial.print(" ");
 Serial.print("x[1]=");Serial.print(x[1]);
 Serial.print(" ");
 Serial.print("x[0]=");Serial.println(x[0]);

x[6] = (x[5]);
x[5] = (x[4]);
x[4] = (x[3]);               // Actualizamos los valores del buffer
x[3] = (x[2]);
x[2] = (x[1]);
x[1] = (x[0]);

Serial.println(y);
Serial.print("x[6]=");Serial.print(x[6]);              // Imprimimos en monitor serial
 Serial.print("x[5]=");Serial.print(x[5]);
 Serial.print(" ");
 Serial.print("x[4]=");Serial.print(x[4]);
 Serial.print(" ");
```

```
 Serial.print("x[3]=");Serial.print(x[3]);
 Serial.print(" ");
 Serial.print("x[2]=");Serial.print(x[2]);
 Serial.print(" ");
 Serial.print("x[1]=");Serial.print(x[1]);
 Serial.print(" ");
 Serial.print("x[0]=");Serial.println(x[0]);}
```

A14. Medición del tiempo que tarda la operación del filtro no.1

```
#include <TimerOne.h>
volatile int sensor;
volatile int x[7]={0,0,0,0,0,0,0};//son valores de voltaje decimales.
volatile int y;
void setup() {
Timer1.initialize(1000000); //llamar interrupcion cada 1 segundo
Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupcion Timer ISR
  Serial.begin(9600);} // inicializa el Puerto Serial
unsigned long start, finished, elapsed;

void loop() {}

//Rutina de interrupción
void timerIsr(){
start=micros();  // inicio cronometro
int sensor = analogRead(A0)>>2; // tomar muestra  Analog read
x[0]=sensor;  // pasar este valor a X[0] del buffer
y = x[0]-x[6];   // operación para filtro #1
x[6] = x[5];
x[5] = x[4];
x[4] = x[3];         // Actualizamos los valores del buffer
x[3] = x[2];
x[2] = x[1];
x[1] = x[0];
finished=micros(); // fin cronometro
elapsed=finished-start;  // duración de la operación
Serial.print(elapsed);   // imprimir valores Puerto serial
Serial.println(" micro segundos");}
```

A15. Medición del tiempo que tarda la operación del filtro no.2

```
#include <TimerOne.h>
volatile int sensor;
volatile int x[4];
volatile int y;
unsigned long start, finished, elapsed;

void setup() {
DDRD = 0xFF;  // declaramos como salidas el port D completo
Timer1.initialize(2777); //1/360=.002777 calculo para interrupcion 360Hz
Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupcion Timer ISR
Serial.begin(115200);} // inicializa el Puerto Serial
 void loop() {}

//Rutina de interrupcion
void timerIsr()
{start=micros();  // inicio cronometro
 x[0]= analogRead(A0) >> 2;; // lee valores pin A0
y = x[0]+x[3]; // Ecuacion del filtro

PORTD = (y); // imprime valor y en port D

x[3] = (x[2]);
x[2] = (x[1]);
x[1] = (x[0]);// Actualizamos los valores del buffer
finished=micros(); // fin cronometro
elapsed=finished-start;  // duración de la operación
Serial.print(elapsed);   // imprimir valores Puerto serial
Serial.println(" micro segundos");}
```

A16. Medición del tiempo que tarda la operación del filtro no.3

```
#include <TimerOne.h>
volatile float x[3];
volatile float y;
unsigned long start, finished, elapsed;

void setup() {
 DDRD = 0xFF;
```

```
Timer1.initialize(2777); //1/360=.002777 cálculo para interrupcion 360Hz
  Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupcion Timer ISR
Serial.begin(115200);}
void loop() {}

//Rutina de interrupcion
void timerIsr()
{start=micros();
x[0]= analogRead(A0) >> 2; // lee valores pin A0

y = x[0]+x[1]+x[2]; // Ecuacion del filtro
PORTD = y; // imprime valor y en port D

x[2] = (x[1]);
x[1] = (x[0]);// Actualizamos los valores del buffer
finished=micros();
elapsed=finished-start;
Serial.print(elapsed);
Serial.println(" micro segundos");}
```

A17. Medición del tiempo que tarda la operación del filtro no.4

```
#include <TimerOne.h>
volatile int sensor;
volatile int x[7];
volatile int y[7];
volatile int w;
unsigned long start, finished, elapsed;

void setup() { DDRD = 0xFF; // declaramos como salidas el port D completo
Timer1.initialize(500000); //1/300=.003333 calculo para interrupcion 300Hz
  Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupcion Timer ISR
  Serial.begin(9600);} // inicializa el Puerto Serial
 void loop() {}
//Rutina de interrupcion
void timerIsr()
{start=micros();  // inicio cronometro
  int sensor = analogRead(A0); // tomo valores para x[0]
x[0] = analogRead(A0)>> 2; // muestreo valores para x[0] y paso de 10 bits a 8 bits

y[0] = ((.5)*(x[0]-x[6]+(.5*y[6]))); // Ecuacion del filtro // si la señal es sin el ruido basal  la
ganancia se tiene que dividir .7 la señal si es con ruido basal se deja igual.
w=(y[0]+127); // imprime valor y en port D // pruebas a 127 bits, 155, y 195 fue lo max antes de
saturarse

PORTD = w;
x[6] = (x[5]);
x[5] = (x[4]);
x[4] = (x[3]);   // Actualizamos los valores del buffer
x[3] = (x[2]);
x[2] = (x[1]);
x[1] = (x[0]);

y[6] = (y[5]);
y[5] = (y[4]);
y[4] = (y[3]);   // Actualizamos los valores del buffer
y[3] = (y[2]);
y[2] = (y[1]);
y[1] = (y[0]);
finished=micros(); // fin cronometro
elapsed=finished-start;  // duración de la operación
Serial.print(elapsed);   // imprimir valores Puerto serial
Serial.println(" micro segundos");}
```

A18. Filtro #1

```
#include <TimerOne.h>
volatile int sensor;
volatile int x[7];
volatile int y;
void setup() {
   DDRD = 0xFF; // declaramos como salidas el port D completo
Timer1.initialize(2777); //1/300=.003333 cálculo para interrupcion 300Hz
  Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupcion Timer ISR
}
 void loop() {}
//Rutina de interrupcion
void timerIsr()
```

```
{int sensor = analogRead(A0); // tomo valores para x[0]
x[0] = analogRead(A0)>> 2; // tomo valores para x[0] y paso de 10 bits a 8 bits
  // Desplazar dos bits es mucho más rápido que una división
y = ((x[0]-x[6])); // Ecuación del filtro // si la señal es sin el ruido basal  la ganancia se tiene que
dividir .7 la señal si es con ruido se deja igual.
PORTD = (y+195); // imprime valor y en port D // pruebas a 127 bits, 155, y 195 fue lo máx. antes de
saturarse
x[6] = (x[5]);
x[5] = (x[4]);
x[4] = (x[3]);   // Actualizamos los valores del buffer
x[3] = (x[2]);
x[2] = (x[1]);
x[1] = (x[0]);}
```

A19. Filtro #2

```
#include <TimerOne.h>
volatile int sensor;
volatile int x[4];
volatile int y;
void setup() {
  DDRD = 0xFF;  // declaramos como salidas el port D completo
Timer1.initialize(2777); //1/360=.002777 cálculo para interrupción 360Hz
  Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupción Timer ISR
  }
void loop()
{}
//Rutina de interrupcion
void timerIsr()
{x[0]= analogRead(A0) >> 2;; // lee valores pin A0
y = x[0]+x[3]; // Ecuación del filtro
PORTD = (y); // imprime valor y en port D
x[3] = (x[2]);
x[2] = (x[1]);
x[1] = (x[0]);// Actualizamos los valores del buffer
}
```

A20. Filtro #3

```
#include <TimerOne.h>
volatile float x[3];
volatile float y;
void setup() {
 DDRD = 0xFF;
Timer1.initialize(2777); //1/360=.002777 cálculo para interrupcion 360Hz
  Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupcion Timer ISR
}
void loop()
{}
//Rutina de interrupcion
void timerIsr()
{x[0]= analogRead(A0) >> 2; // lee valores pin A0
y = x[0]+x[1]+x[2]; // Ecuación del filtro
PORTD = y; // imprime valor y en port D
x[2] = (x[1]);
x[1] = (x[0]);// Actualizamos los valores del buffer
}
```

A21. Filtro #4

```
#include <TimerOne.h>
volatile int sensor;
volatile int x[7];
volatile int y[7];
volatile int w;

void setup() {
   DDRD = 0xFF; // declaramos como salidas el port D completo
Timer1.initialize(2777); //1/300=.003333 cálculo para interrupción 300Hz
  Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupción Timer ISR
}
 void loop() {}

//Rutina de interrupción
void timerIsr()
{int sensor = analogRead(A0); // tomo valores para x[0]
x[0] = analogRead(A0)>> 2; // tomo valores para x[0] y paso de 10 bits a 8 bits
y[0] = ((.9)*(x[0]-x[6]+(.9*y[6]))); // Ecuación del filtro // si la señal es sin el ruido basal  la
ganancia se tiene que dividir .7 la señal si es con ruido basal se deja igual.
```

```
w=(y[0]+127); // imprime valor y en port D // pruebas a 127 bits, 155, y 195 fue lo máx. antes de
saturarse
PORTD = w;
x[6] = (x[5]);
x[5] = (x[4]);
x[4] = (x[3]);   // Actualizamos los valores del buffer
x[3] = (x[2]);
x[2] = (x[1]);
x[1] = (x[0]);
y[6] = (y[5]);
y[5] = (y[4]);
y[4] = (y[3]);   // Actualizamos los valores del buffer
y[3] = (y[2]);
y[2] = (y[1]);
y[1] = (y[0]);
}
```

A22. Filtro para eliminar ruido 60Hz en transmisión Serie

```
#include <TimerOne.h>
volatile int sensor;
volatile int x[4];// número de datos del arreglo
volatile int y;// variable salida
void setup(){
Serial.begin(9600);
Timer1.initialize(2777); //1/360=.002777 cálculo para interrupcion 300Hz
Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupción Timer ISRD
}
void loop() {}

//Rutina de interrupción
void timerIsr()
{sensor= analogRead(A0); // lee valores pin A0
 x[0]= sensor; // lee valores pin A0
y = x[0]+x[3]; // Ecuación del filtro
Serial.print(y); // imprime valores
//Serial.print("v"); // imprime letra "v" LABview la utiliza para saber que hay un nuevo valor a
graficar.
x[3] = (x[2]);
x[2] = (x[1]);  // buffer circular, actualiza los valores para  aplicar el filtro.
x[1] = (x[0]);}
```

A23. Filtro para eliminar ruido 60Hz en data logger

```
#include <SPI.h>
#include <SD.h>
File myFile;
#include <DS3231.h>
#include <Wire.h>
#include <TimerOne.h>
DS3231 Clock;
byte year, month, date, DoW, hour, minute, second;
char linea[64]; // ¿64 caracteres por línea (máximo) serán suficientes?
char voltaje[10]; // Se usa para obtener el resultado de la conversión float a texto
volatile int x[4];// número de datos del arreglo
volatile int y;// variable salida

#define LED1    A1
#define LED2    A2
#define chipSelect      10
#define LED3    4
#define LED4    3
#define LED5    2
#define LED6    A3

void setup()
{TIMSK0 = 0;
  Wire.begin();
  pinMode(7, INPUT_PULLUP);
  pinMode(6, INPUT_PULLUP);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(chipSelect, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
  pinMode(LED5, OUTPUT);
  pinMode(LED6, OUTPUT);
digitalWrite(LED1, LOW);//Prende LED 1 Grabar
```

```
digitalWrite(LED2, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED3, HIGH);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar

  if (!SD.begin(chipSelect))
  {digitalWrite(LED1, LOW);//Prende LED 1 Grabar
digitalWrite(LED2, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED3, LOW);//Prende LED 1 Grabar
digitalWrite(LED4, HIGH);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
    return; }
digitalWrite(LED1, LOW);//Prende LED 1 Grabar
digitalWrite(LED2, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED3, HIGH);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
  Timer1.initialize(2777); // 1/500= .002 =2000 micros para 500 muestras
}
void loop()
{if (!digitalRead(7)) grabar();
 //Clock.getTime(year, month, date, DoW, hour, minute, second);
}
void grabar() {
  while (!digitalRead(7)); // Espera a que el botón se suelte
  myFile = SD.open("datalog1.txt", FILE_WRITE);//abrimos  el archivo
  if (!myFile) {
    return;}
digitalWrite(LED1, HIGH);//Prende LED 1 Grabar
digitalWrite(LED2, LOW);//Apaga LED 2 FinalizardigitalWrite(LED3, HIGH);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar

  Clock.getTime(year, month, date, DoW, hour, minute, second);
  myFile.write(linea, sprintf(linea, "Iniciado el %02d/%02d/20%02d, a las %02d:%02d:%02d\r\n", date,
month, year, hour, minute, second)); //
   Timer1.attachInterrupt (timerIsr); // attach service
  while (digitalRead(7)){
     unsigned long tAnterior = 0;
    unsigned long tActual = millis();
    if (tActual >= tAnterior) { // Pide información nueva al RTC cada segundo
      tAnterior = tActual + 1000;
      Clock.getTime(year, month, date, DoW, hour, minute, second);
    }
  Timer1.detachInterrupt(); // detach service
  while (!digitalRead(7)); // Espera a que el botón se suelte
  Clock.getTime(year, month, date, DoW, hour, minute, second);
  myFile.write(linea, sprintf(linea, "Finalizado el %02d/%02d/20%02d, a las %02d:%02d:%02d\r\n", date,
month, year, hour, minute, second)); // Pie
myFile.close(); //cerramos el archivo
digitalWrite(LED1, LOW);//Prende LED 1 Grabar
digitalWrite(LED2, HIGH);//Apaga LED 2 Finalizar
digitalWrite(LED3, LOW);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, HIGH);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
}
void timerIsr()
{int voltaje= analogRead(A0);
 x[0]= voltaje; // lee valores pin A0
y = x[0]+x[3]; // Ecuación del filtro
    myFile.print(y);
    myFile.write(linea, sprintf(linea, "\t%02d:%02d:%02d\r\n",  hour, minute, second));
x[3] = (x[2]);
x[2] = (x[1]);  // buffer circular, actualiza los valores para  aplicar el filtro.
x[1] = (x[0]);}
```

A24. Filtro para eliminar ruido 60Hz en bluetooth celular

```
#include <SoftwareSerial.h>
#include <TimerOne.h>
SoftwareSerial BT(9,10);//RX,TX
volatile int sensor;
volatile int x[4];// número de datos del arreglo
volatile int y;// variable salida

void setup(){
BT.begin(115200);
```

```
Timer1.initialize(2777); //1/360=.002777 cálculo para interrupcion 300Hz
Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupción Timer ISRD
}
void loop() {}
//Rutina de interrupción
void timerIsr()
{sensor= analogRead(A0); // lee valores pin A0
 x[0]= sensor; // lee valores pin A0
y = x[0]+x[3]; // Ecuación del filtro
BT.print("E"); // imprime letra "v" LABview la utiliza para saber que hay un nuevo valor a graficar.
BT.println(y); // imprime valores
x[3] = (x[2]);
x[2] = (x[1]);  // buffer circular, actualiza los valores para  aplicar el filtro.
x[1] = (x[0]);}
```

A25. Filtro para eliminar ruido 60Hz en bluetooth PC

```
#include <SoftwareSerial.h>
#include <TimerOne.h>
SoftwareSerial BT(9,10);//RX,TX
volatile int sensor;
volatile int x[4];// número de datos del arreglo
volatile int y;// variable salida
void setup(){
BT.begin(115200);
Timer1.initialize(2777); //1/360=.002777 cálculo para interrupcion 300Hz
Timer1.attachInterrupt (timerIsr); // llama la Rutina de interrupcion Timer ISRD
}
void loop() {}
//Rutina de interrupcion
void timerIsr()
{sensor= analogRead(A0); // lee valores pin A0
 x[0]= sensor; // lee valores pin A0
y = x[0]+x[3]; // Ecuación del filtro
BT.print(y); // imprime valores
BT.print("v"); // imprime letra "v" LABview la utiliza para saber que hay un nuevo valor a graficar.
x[3] = (x[2]);
x[2] = (x[1]);  // buffer circular, actualiza los valores para  aplicar el filtro.
x[1] = (x[0]);}
```

A26. ECG monitor de uso personal y uso prolongado. (Interfaz con LEDS)

```
#include <RTClib.h>
RTC_DS1307 RTC;
#include <SoftwareSerial.h>
#include <TimerOne.h>  // declarar  uso de libreria
SoftwareSerial BT(8,9);//RX,TX
#include <SPI.h>
#include <SD.h>
File myFile;
//#include <DS3231.h>
#include <Wire.h>
//DS3231 Clock;
byte year, month, date, DoW, hour, minute, second;
char linea[15];
char voltaje[4];


// --- Mapeamento de Hardware ---
#define butDown 7
#define butP    5
#define butM    6

#define LED1    A1
#define LED2    A2
#define CS      10
#define LED3    4
#define LED4    3
#define LED5    2
#define LED6    A3

// Funciones  Menu
void changeMenu();
void dispMenu();
void Modo1();
void Modo2();
void Modo3();
void menu4();
```

```
// Variables Globales
char menu = 0x01;  //Variable para seleccionar el menú
char set1 = 0x00; //Control de los LEDS
boolean t_butDown, t_butP, t_butM; //Banderas para almacenar el estado de los botones

void setup()
{ Wire.begin();
 RTC.begin();
  pinMode(7, INPUT_PULLUP);
  pinMode(5, INPUT_PULLUP);
  pinMode(6, INPUT_PULLUP);

  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(CS, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
  pinMode(LED5, OUTPUT);
  pinMode(LED6, OUTPUT);

  t_butDown = 0x00;//Bandera down 0
  t_butP    = 0x00;//Bandera grabar 0
  t_butM    = 0x00; //Bandera stop 0
  } //end setup

void loop()
{
   changeMenu();
   dispMenu();
   getTime(&year, &month, &date, &DoW, &hour, &minute, &second);
 //Clock.getTime(year, month, date, DoW, hour, minute, second);
  //delayMicroseconds(100);
}

void getTime(byte* year, byte* month, byte* date, byte* DoW, byte* hour, byte* minute, byte* second) {
  DateTime now = RTC.now();
  //if (!now) return; // No actualiza si por alguna razón no puede obtener los datos nuevos
  *year = now.year()- 2000;
  *month = now.month();
  *date = now.day();
  *DoW = now.dayOfWeek();
  *hour = now.hour();
  *minute = now.minute();
  *second = now.second();
}

//FN del menu
void changeMenu() //Cambia menu Actual
{
if(!digitalRead(butDown)) t_butDown = 0x01; //Down presionado? activar Bandera
if(digitalRead(butDown) && t_butDown)  //Up Suelto y bandera activa?
{ t_butDown = 0x00;  //Limpia bandera
 menu++;

if(menu > 0x04) menu = 0x01; //Si menu es mayor a 4 regresar menu a 1.
} //end butUp
} //end changeMenu

void dispMenu() //Mostrar menu actual
{
switch(menu) //(cambiar menu) (Control de la variable de menú)
{
case 0x01:  //Caso 1
Modo1(); //llama Modo 1 Serial port COM
digitalWrite(LED3, HIGH);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
break;
case 0x02:  //Caso 2
Modo2(); //llama Modo 2 SD Card
   getTime(&year, &month, &date, &DoW, &hour, &minute, &second);
digitalWrite(LED3, LOW);//Prende LED 1 Grabar
digitalWrite(LED4, HIGH);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
break;
case 0x03:  //Caso 3
Modo3();  //llama Modo 3 Bluetooth Celular
digitalWrite(LED3, LOW);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
```

```
digitalWrite(LED5, HIGH);//Apaga LED 2 Finalizar
digitalWrite(LED6, LOW);//Apaga LED 2 Finalizar
break;
case 0x04: //Caso 4
Modo4(); //Llama modo 4 Bluetooth PC
digitalWrite(LED3, LOW);//Prende LED 1 Grabar
digitalWrite(LED4, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED5, LOW);//Apaga LED 2 Finalizar
digitalWrite(LED6, HIGH);//Apaga LED 2 Finalizar
break;
} //end Cambiar menu
} //end Mostrar Menu

void Modo1() //Modo 1 Serial port
{
if(!digitalRead(butP))    t_butP    = 0x01; // grabar presionado? Activar bandera
if(!digitalRead(butM))    t_butM    = 0x01; //stop presionado? Activar bandera
if(digitalRead(butP) && t_butP) //Boton suelto y bandera activa?
{
t_butP = 0x00; //limpiar bandera
Puertoserie(); //Llama Funcion Puerto Serie
digitalWrite(LED1, HIGH);//Prende LED 1 Grabar
digitalWrite(LED2, LOW);//Apaga LED 2 Finalizar

   } //end butP

   if(digitalRead(butM) && t_butM) //Boton stop suelto y bandera activa?
{
t_butM = 0x00; //Limpia bandera
Serial.end(); // Finaliza Comunicacion Serial
digitalWrite(LED1, LOW); //Apaga LED 1 Grabar
digitalWrite(LED2, HIGH); //Prende LED 2 Finalizar
delay(1500); // Espera 1.5 segundos para  iniciar nuevo modo.
digitalWrite(LED2, LOW); //Apaga LED 2 Para iniciar nuevo modo
} //end butM
} //end Modo1

void Modo2()  //Modo 2 SD card
{
if(!digitalRead(butP))    t_butP    = 0x01; // Boton grabar presionado? Activar bandera
 if(!digitalRead(butM))    t_butM    = 0x01; //Boton stop presionado? Activar bandera
 if(digitalRead(butP) && t_butP) //Boton grabar suelto y bandera activa?
 {
    if (!SD.begin(CS))
  {}
 t_butP = 0x00;  //limpiar bandera
   set1++;

      if(set1 > 2) set1 = 0x01;

      switch(set1)
      {
          case 0x01:                                      //Caso 1
digitalWrite(LED1, LOW); //Prende LED 1 Grabar
digitalWrite(LED2, LOW); //Apaga LED 2 FinalizarmyFile = SD.open("datalog1.txt", FILE_WRITE);//abrimos
el archivo
myFile = SD.open("datalog1.txt", FILE_WRITE);//abrimos  el archivo
//Clock.getTime(year, month, date, DoW, hour, minute, second);
getTime(&year, &month, &date, &DoW, &hour, &minute, &second);
myFile.write(linea, sprintf(linea, "Iniciado el %02d/%02d/20%02d, a las %02d:%02d:%02d\r\n", date, month,
year, hour, minute, second));
 break;                                  //Break


          case 0x02:                                       //Caso 2
Timer1.initialize(2000);
Timer1.attachInterrupt (timerIsr2);
 digitalWrite(LED1, HIGH); //Prende LED 1 Grabar
 digitalWrite(LED2, LOW); //Apaga LED 2 Finalizar
  } //end switch set1
 }//end but P


   if(digitalRead(butM) && t_butM) //Boton stop suelto y bandera activa?
   {
    Timer1.detachInterrupt(); // detach service
  t_butM = 0x00; //Limpia bandera
  getTime(&year, &month, &date, &DoW, &hour, &minute, &second);
   // Clock.getTime(year, month, date, DoW, hour, minute, second);
  myFile.write(linea, sprintf(linea, "Finalizado el %02d/%02d/20%02d, a las %02d:%02d:%02d\r\n", date,
month, year, hour, minute, second)); // Pie
  myFile.close(); //cerramos el archivo
```

```
digitalWrite(LED1, LOW); //Apaga LED 1 Grabar
digitalWrite(LED2, HIGH); //Prende LED 2 Finalizar
delay(1500); // Espera 1.5 segundos para  iniciar nuevo modo.
digitalWrite(LED2, LOW); //Apaga LED 2 Para iniciar nuevo modo
} //end butM
} //end Modo 2()

void Modo3()  //Modo 3 cel Bluetooth
{
if(!digitalRead(butP))    t_butP    = 0x01;
if(!digitalRead(butM))    t_butM    = 0x01;

if(digitalRead(butP) && t_butP)
{
t_butP = 0x00;
BluetoothCel();
digitalWrite(LED1, HIGH);
digitalWrite(LED2, LOW);
} //end butP

if(digitalRead(butM) && t_butM)
{
t_butM = 0x00;
BT0CEL();
digitalWrite(LED1, LOW);
digitalWrite(LED2, HIGH);
delay(1500);
digitalWrite(LED2, LOW);
} //end butM
} //end Modo3

void Modo4() //Transmision Bluetooth PC
{
  if(!digitalRead(butP))    t_butP    = 0x01;
 if(!digitalRead(butM))    t_butM    = 0x01;

 if(digitalRead(butP) && t_butP)
 {
 t_butP = 0x00;
 BluetoothPC();
 digitalWrite(LED1, HIGH);
 digitalWrite(LED2, LOW);
 } //end butP

if(digitalRead(butM) && t_butM)
{
 t_butM = 0x00;
 BT0PC();
 digitalWrite(LED1, LOW);
 digitalWrite(LED2, HIGH);
 delay(1500);
 digitalWrite(LED2, LOW);
} //end butM
} //end Modo4

void Puertoserie() {
  Serial.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr);
}
void timerIsr()
{volatile int sensor= analogRead(A0);
Serial.print(sensor);
Serial.print("v");
}
void timerIsr2()
{      //dtostrf(analogRead(A0) * FACTOR_VOLTAJE, 0, 2, voltaje);
      dtostrf(analogRead(A0) * 1, 1, 0, voltaje);
     //   myFile.write(linea,   sprintf(linea,   "%s\t%02d:%02d:%02d\r\n",   voltaje,   hour,   minute,
second));//(SIN FECHA)
     myFile.write(linea, sprintf(linea, "%s\t%02d/%02d/20%02d\t%02d:%02d:%02d\r\n", voltaje, date, month,
year, hour, minute, second));
}
void BluetoothCel() {
BT.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr1);
  }
void timerIsr1()
{
volatile int sensor= analogRead(A0);
//int sensorx=((sensor*-1)+485);
```

```
int sensorx=sensor;
BT.print("E");
BT.println(sensorx);
}
void BluetoothPC(){
BT.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr3);
}
void timerIsr3()
{ volatile int sensor= analogRead(A0);
BT.print("v");
BT.print(sensor);
}
void BT0PC(){
BT.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr4);
}
//Rutina de interrupcion
void timerIsr4()
{ volatile int sensor=0;
BT.print("v");
BT.print(sensor);
}
void BT0CEL(){
BT.begin(115200);
Timer1.initialize(2777);
Timer1.attachInterrupt (timerIsr5);
}
void timerIsr5()
{ volatile int sensor=0;
BT.print("E");
BT.println(sensor);
}
```

A27. Código para generación de señales y simulación de filtros digitales en Matlab

```
x=ecgsim(3,9)
figure
subplot(2,1,1)
plot(x)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Señal de ECG de 300 muestras');

%señan original
y=resample(x,12,10); %señal muestreada a 360
subplot(2,1,2)
plot(y)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Señal de ECG a 360 muestras');

z=[y,y,y,y,y]; %vector a*20
n = 0:1799;
figure
subplot(2,2,1);
plot(z)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Señal ECG original');
grid on

%ruido a 60 hertz
ruido_60 = 0.25*sin((pi/3)*n);
%ruido a 120 hertz
ruido_120 = 0.1*sin(2*(pi/3)*n);
%ruido a 180 hertz
ruido_180 = 0.08*sin(pi*n);
%offset
offset = 0.5*sin((1/900)*pi*n);

%suma de señal ECG ideal + ruido_60
ecg_ruido_60 = z + ruido_60;
subplot(2,2,2);
plot(ecg_ruido_60)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Señal ECG ideal con ruido de 60 Hertz');
```

```
grid on

%suma de los ruidos
suma_ruido = ruido_60 + ruido_120 + ruido_180;

%suma de la señal original más el ruido
ecg_ruido = z + suma_ruido;
subplot(2,2,3);
plot(ecg_ruido)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Señal de ECG ideal con ruido de 60 Hertz y sus armonicos');
grid on

%suma de offset y señal con ruido
ecg_final = (ecg_ruido + offset);
subplot(2,2,4);
plot(ecg_final)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Suma de señal de offset a señal de ECG con ruido');
grid on

%filtro peine y[n] = x[n] - x[n-6] probado con señal de ECG con offset y ruido
b = [1, 0, 0, 0, 0, 0, -1];
a = 1;
c = filter(b,a,ecg_final);
figure
plot(c)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Señal filtrada con filtro comb y[n] = x[n] - x[n-6]');
grid on
fvtool(b,a)
title('Señal filtrada con filtro comb y[n] = x[n] - x[n-6]');


%filtro peine y[n] = x[n] + x[n-3] probado con la señal de ECG con ruido de 60 Hertz
b2 = [1, 0, 0, 1];
a2 = 1;
c2 = filter(b2,a2,ecg_ruido_60);
plot(c2)
xlabel('Numero d Muestras');
ylabel('Amplitud (V)');
title('Señal filtrada con filtro comb y[n] = x[n] + x[n-3]');
grid on
fvtool(b2,a2)
title('Señal filtrada con filtro comb y[n] = x[n] + x[n-3]');


%filtro peine y[n] = x[n] + x[n-3]
b3 = [1, 0, 0, 1];
a3 = 1;
c3 = filter(b3,a3,ecg_final);
plot(c3)
xlabel('Numero d Muestras');
ylabel('Amplitud (V)');
title('Señal filtrada con filtro comb y[n] = x[n] + x[n-3]');
grid on
fvtool(b3,a3)
title('Señal filtrada con filtro comb y[n] = x[n] + x[n-3]');


%filtro peine y[n] = x[n] + ax[n-1] + bx[n-2] probado con la señal de ECG con ruido de 60 Hertz
b4 = [1, -1, 1];
a4 = 1;
c4 = filter(b4,a4,ecg_ruido_60);
plot(c4)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Señal filtrada con filtro y[n] = x[n] - ax[n-1] + bx[n-2]');
grid on
fvtool(b4,a4)
title('Señal filtrada con filtro y[n] = x[n] - ax[n-1] + bx[n-2]');


%filtro peine y[n] = x[n] + ax[n-1] + bx[n-2] probado con la señal de ECG final
b5 = [1, -1, 1];
a5 = 1;
c5 = filter(b5,a5,ecg_final);
plot(c5)
xlabel('Numero de Muestras');
```

```
ylabel('Amplitud (V)');
title('Señal filtrada con filtro y[n] = x[n] - ax[n-1] + bx[n-2]');
grid on
fvtool(b5,a5)
title('Señal filtrada con filtro y[n] = x[n] - ax[n-1] + bx[n-2]');


%filtro Notch periodico y[n] = x[n] - x[n-N] + polo en z=0.5 con señal de ECG final
b6 = [1, 0, 0, 0, 0, 0, -1];
a6 = [1, 0, 0, 0, 0, 0, -0.5];
c6 = filter(b6,a6,ecg_final);
plot(c6)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Señal filtrada con filtro Filtro Notch periodico y[n] = x[n] - x[n-6] + ay[n-6] con a = 0.5');
grid on
fvtool(b6,a6)
title('Señal filtrada con filtro comb y[n] = x[n] - x[n-6], alfa 0.5');


%filtro Notch periodico y[n] = x[n] - x[n-N] + polo en z=0.9 con la señal de ECG final
b7 = [1, 0, 0, 0, 0, 0, -1];
a7 = [1, 0, 0, 0, 0, 0, -0.9];
c7 = filter(b7,a7,ecg_final);
plot(c7)
xlabel('Numero de Muestras');
ylabel('Amplitud (V)');
title('Señal filtrada con filtro Filtro Notch periodico y[n] = x[n] - x[n-6] + ay[n-6] con a = 0.9');
grid on
fvtool(b7,a7)
title('Señal filtrada con filtro comb y[n] = x[n] - x[n-6], alfa 0.9');
```