UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE CUENCA

CARRERA DE INGENIERÍA ELECTRÓNICA

Trabajo de titulación previo
a la obtención del título de
Ingeniero Electrónico

PROYECTO TÉCNICO CON ENFOQUE INVESTIGATIVO:

DISEÑO DE UN PROTOTIPO ELECTRÓNICO DE BAJO COSTO Y REDUCIDAS DIMENSIONES QUE PERMITA LA ADQUISICIÓN, PROCESAMIENTO Y VISUALIZACIÓN DE SEÑALES CARDÍACAS

AUTORES:

RÓMULO ISMAEL NARVÁEZ MIRANDA

MATEO SANTIAGO VALVERDE JARA

TUTOR:

ING. EDUARDO PINOS VÉLEZ, Ph.D.

CUENCA - ECUADOR

2021

CESIÓN DE DERECHOS DE AUTOR

Nosotros, Rómulo Ismael Narváez Miranda con documento de identificación N° 0605613397 y Mateo Santiago Valverde Jara con documento de identificación N° 0104554449, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación: **"DISEÑO DE UN PROTOTIPO ELECTRÓNICO DE BAJO COSTO Y REDUCIDAS DIMENSIONES QUE PERMITA LA ADQUISICIÓN, PROCESAMIENTO Y VISUALIZACIÓN DE SEÑALES CARDÍACAS"**, mismo que ha sido desarrollado para optar por el título de: *Ingeniero Electrónico,* en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autores nos reservamos los derechos morales de la obra antes citada. En concordancia, suscribimos este documento en el momento que hacemos entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, marzo del 2021

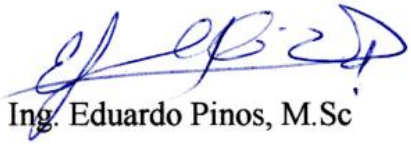
Rómulo Ismael Narváez Miranda
C.I. 0605613397

Mateo Santiago Valverde Jara
C.I. 0104554449

CERTIFICACIÓN

Yo, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: **"DISEÑO DE UN PROTOTIPO ELECTRÓNICO DE BAJO COSTO Y REDUCIDAS DIMENSIONES QUE PERMITA LA ADQUISICIÓN, PROCESAMIENTO Y VISUALIZACIÓN DE SEÑALES CARDÍACAS"**, realizado por Rómulo Ismael Narváez Miranda y Mateo Santiago Valverde Jara, obteniendo el *Proyecto Técnico con enfoque investigativo*, que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana.

Cuenca, marzo del 2021



Ing. Eduardo Pinos, M.Sc

C.I. 0102942190

DECLARATORIA DE RESPONSABILIDAD

Nosotros, Rómulo Ismael Narváez Miranda con documento de identificación N° 0605613397 y Mateo Santiago Valverde Jara con documento de identificación N° 0104554449, autores del trabajo de titulación: **"DISEÑO DE UN PROTOTIPO ELECTRÓNICO DE BAJO COSTO Y REDUCIDAS DIMENSIONES QUE PERMITA LA ADQUISICIÓN, PROCESAMIENTO Y VISUALIZACIÓN DE SEÑALES CARDÍACAS",** certificamos que el total contenido del *Proyecto Técnico con enfoque investigativo*, es de nuestra exclusiva responsabilidad y autoría.

Cuenca, marzo del 2021

Rómulo Ismael Narváez Miranda
C.I. 0605613397

Mateo Santiago Valverde Jara
C.I. 0104554449

AGRADECIMIENTOS

Mi gratitud infinita a Dios, quien con sus bendiciones llena mi vida y me guía a lo largo de mi existencia. Mi profundo agradecimiento a mis padres por su confianza y apoyo incondicional, por ser los principales promotores de mis sueños y metas y por sus consejos, valores y principios que me han inculcado.

Al personal docente y administrativo de la Universidad Politécnica Salesiana por aportar acertadamente a mi formación personal y profesional. A nuestro director de tesis, Eduardo Pinos, por su guía y apoyo durante el desarrollo de este proyecto. A mi buen amigo y compañero de tesis Mateo por su arduo esfuerzo, disciplina y motivación durante el desarrollo de esta tesis.

Rómulo Ismael Narváez Miranda

Doy gracias infinitas a Dios, por haberme dado fuerza y valor a lo largo de mi vida. Agradezco también el apoyo y confianza que mis padres han depositado en mi demostrándome su amor incondicional, corrigiendo mis faltas y celebrando mis triunfos, a mis hermanos por estar presentes a lo largo de los años. A la Universidad Politécnica Salesiana, su personal docente y administrativo quienes con su ardua labor han aportado en mi crecimiento personal y profesional. A nuestro tutor de tesis Eduardo Pinos por su apoyo y guía durante este proyecto, a mi compañero de tesis y amigo Ismael. por haber tenido paciencia y porque no permitió que nos rindamos en el camino, mis amigos y compañeros que con sus constantes preguntas "¿y la tesis para cuándo?", demostraban su apoyo y presionaron para que pueda cumplir con mis metas.

Mateo Santiago Valverde Jara

DEDICATORIAS

A mis padres Mirella y Rómulo quienes con su paciencia, amor, esfuerzo y motivación me han permitido cumplir una meta más y me han dado la oportunidad de formarme en esta prestigiosa universidad. A mis hermanos Déborath, David y Cristian por su apoyo moral y por estar conmigo en todo momento.

Rómulo Ismael Narváez Miranda

En primer lugar, a Dios por haberme dado la vida y permitirme llegar hasta este momento tan importante en mi formación profesional. A mis padres René y Mariana, por ser el pilar más importante en mi vida y porque siempre confiaron y estuvieron a mi lado motivándome y luchando junto a mi para llegar a este día. A mis hermanos mayores Juan, René y Karina por su apoyo incondicional. A mis amigos cercanos Josué J., Jazz V., Belén Q., Fabi V., Sebastián Q., Brian S., en quienes puedo confiar y me han ayudado a levantarme en diferentes etapas de mi vida.

Mateo Santiago Valverde Jara

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

En la actualidad unos de los problemas médicos más destacados son las patologías cardiacas, debido a esto, los instrumentos creados para estudiar el corazón son de gran importancia. Hoy en día, con el avance de la tecnología, se desea que estos instrumentos sean más sencillos de usar, más pequeños, que tengan más prestaciones y que obtengan la mejor calidad de las señales biomédicas.

Para aceptar esta tendencia, varios fabricantes, tales como: Texas Instruments, Analog Devices, Arduino, etc; han desarrollado circuitos integrados que facilitan en gran medida la captación de bioseñales, incluyendo las principales etapas en la adquisición de este tipo de señales. Con esto, se logra reducir el tamaño, peso y complejidad de este tipo de instrumentos, además de que facilitan su diseño, son eficientes en consumo y obteniendo señales de mejor calidad.

En este proyecto de titulación se estudia el diseño y desarrollo de un prototipo de dispositivo de monitorización cardiaca, haciendo uso del circuito integrado ADAS1000 del fabricante Analog Devices, que sirve como circuito de adquisición, filtrado y preprocesamiento de una señal ECG. Este circuito integrado dispone de las entradas necesarias para la conexión de electrodos o sensores, dichas entradas pasan por un amplificador diferencial de ganancia programable, un convertidor analógico – digital y un filtro digital. Este circuito integrado es controlado por un dspic30F4013 que permitirá transmitir los datos de manera alámbrica a una aplicación para Windows y de manera inalámbrica a una aplicación para dispositivos Android usando comunicación bluetooth. Finalmente, se hace una comparación de resultados con dispositivos que existen en el mercado y con los resultados obtenidos se demuestra que, a pesar de que se logró una captación de la señal cardiaca, esta no es aceptable para realizar estudios futuros de patologías, debido al ruido excesivo presente en la señal.

INTRODUCCIÓN

En los últimos años, un campo de investigación de gran interés ha sido el de la adquisición y análisis de señales electrocardiográficas e identificación de eventos patológico y anormales dentro de los registros de ECG. Muchos de estos estudios tienen como base el uso de bases de datos con señales adquiridas como las del MIT, disponibles en el sitio web PHYSIONET [9]. Algunos trabajos de investigación se enfocan en el desarrollo de sistemas computacionales para analizar estas bioseñales [10]. Otros, por su parte se centran en desarrollar algoritmos para identificar patologías referentes a las señales electrocardiográficas [11]. Estas investigaciones han establecido metodologías tanto para la adquisición como para el análisis de las señales ECG, brindando aplicaciones como: segmentar e identificar latidos, filtrar bioseñales, entre otras.

Este proyecto tiene como objetivo principal, desarrollar un prototipo que sea capaz de captar señales ECG por medio de un circuito integrado ADAS1000 que es de bajo costo y poco consumo, para después transmitir los datos a un dispositivo móvil y a una aplicación para Windows y graficar cada una de las derivaciones que el ADAS1000 nos permita en su configuración. Posteriormente, en líneas futuras, se pretende mejorar la etapa de filtrado y adquisición para la captación de señales de alta calidad. Estas señales pueden ser adquiridas directamente de pacientes sanos o quienes presenten algún tipo de patología cardiaca, con el fin de realizar estudios de desarrollo de herramientas que apoyen y faciliten la detección temprana de anomalías cardiacas.

A continuación, se describe el desarrollo del prototipo antes mencionado, los métodos usados para su programación y configuración. Finalmente, se muestran los resultados obtenidos y limitaciones observadas al hacer uso del dispositivo.

ANTECEDENTES DEL PROBLEMA DE ESTUDIO

Durante los últimos 15 años la mayor causa de muertes en el mundo son las cardiopatías o enfermedades del corazón y accidentes cerebrovasculares [1]. En 2015 estas enfermedades causaron alrededor de 15 millones de decesos en el mundo. Es claro que la tasa de mortalidad ha disminuido actualmente pero han ido surgiendo otros problemas de salud como la hipertensión, la obesidad, hábitos sedentarios [2].

Para remediar este problema se debe seguir los tres niveles de prevención que existen:

1. Prevención primaria: informar las causas de la cardiopatía y promover acciones que disminuyan el riesgo de sufrirlas [3].
2. Prevención secundaria: detectar y prevenir el desarrollo de la cardiopatía en estados tempranos. Esto se logra mejorando las tecnologías de detección y diagnóstico. Este proyecto está basado en este nivel [3].
3. Prevención terciaria: Evitar que la enfermedad empeore cuando está instaurada [3].

En el Ecuador existen 1,4 millones de personas que padecen algún tipo de enfermedad cardíaca lo cual implica un gasto 615 millones de dólares a la economía del país [4]. Con estas cifras tan grandes es cierto que existe una tasa de mortalidad alta por lo que detectar una patología cardíaca a tiempo puede significar salvar la vida de alguien.

JUSTIFICACIÓN (IMPORTANCIA Y ALCANCES)

En la actualidad las patologías cardíacas representan un problema relevante, debido a esto, es de gran importancia el desarrollo de dispositivos e instrumentos para estudiar el corazón. Por lo tanto, la continua monitorización de un paciente puede proveer de vital información a los especialistas para anticiparse a la detección y prevención de patologías cardíacas [2]. Sin embargo, los equipos de electrocardiografía que se encuentran en el mercado actual son costosos y generalmente de difícil acceso, en el Capítulo 1 se describe el estado del arte de algunos dispositivos que existen actualmente en el mercado.

Con este proyecto se pretende desarrollar un prototipo independiente de reducidas dimensiones que pueda interactuar con dispositivos con conexión bluetooth, como teléfonos celulares, computadoras.

OBJETIVOS

OBJETIVO GENERAL

- Diseñar un prototipo electrónico de bajo costo y reducidas dimensiones que permita la adquisición, procesamiento y visualización de señales cardíacas

OBJETIVOS ESPECÍFICO

- Obtener información sobre los estudios actuales y pasados referentes a la electrocardiografía.
- Diseñar y elaborar un PCB que permita la interacción entre el ADAS1000 y el PIC24 para el envío de las señales cardíacas a un dispositivo final.
- Desarrollar una aplicación (para dispositivos Android y SO Windows) que permita la visualización de una señal cardíaca en cada una de sus derivaciones.
- Generar una base de datos con registros de las señales obtenidas.

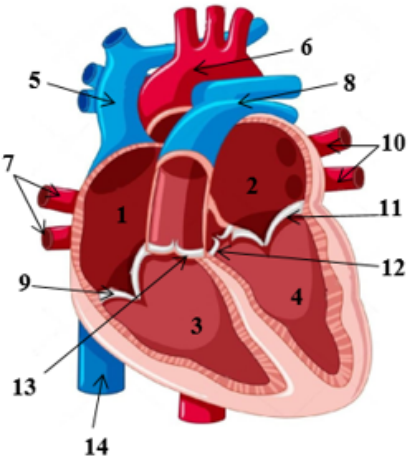
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Para el desarrollo de nuestro prototipo se realizó un estudio de la anatomía del corazón, el electrocardiograma (ECG) y diversas patologías cardíacas, esto con el fin de comprender su funcionamiento, además se muestran los circuitos integrados que utilizaremos como ADAS1000BSTZ, DSPIC30F4013 y el software con el cual diseñaremos la aplicación para celular será Android Studio, el lenguaje de programación para el software de PC utilizamos C# en Visual Studio 2015.

1.1 CONSIDERACIONES ANATÓMICAS

El corazón, es el músculo de mayor importancia y es el órgano central del sistema circulatorio, es encargado de enviar la sangre a todo el cuerpo; está situado dentro de la caja torácica, ubicado detrás del esternón, apoyado sobre el diafragma, situado entre los pulmones. El corazón posee forma piramidal con la punta dirigida hacia abajo, a la derecha y atrás, la base de la pirámide se encuentra hacia arriba, a la izquierda y adelante [5], [6].

Miocardio es la masa muscular que lo constituye, se encuentra compuesto internamente por el endocardio y exteriormente por el epicardio, rodeado una membrana que se denomina pericardio, el corazón se divide en dos partes el corazón derecho y el corazón izquierdo, cada parte del corazón a su vez se subdivide en dos cavidades: una cavidad auricular y una cavidad ventricular para una mejor referencia puede observar la Figura 1 [6].



1. Aurícula derecha	2. Aurícula Izquierda
3. Ventrículo derecho	4. Ventrículo Izquierdo
5. Vena Cava superior	6. Vena cava inferior
7. Vena pulmonar derecha	8. Arteria Pulmonar
9. Válvula tricúspide	10. Vena pulmonar izquierda
11. Válvula mitral	12. Válvula aórtica
13. Válvula Pulmonar	14. Vena Cava Inferior

Figura 1 Partes del Corazón[1]

1.2 FUNCIONAMIENTO DEL CORAZÓN

Las aurículas y los ventrículos conjuntamente con 4 válvulas cuyos nombres son: mitral, tricúspide, aórtica y pulmonar, son las encargadas del flujo de sangre. Cuando el corazón se contrae (sístole) impulsa la sangre a todo el cuerpo mientras que las 4 válvulas se abren, cuando el corazón se relaja (diástole) la sangre se almacena en el corazón mientras que las válvulas se cierran, este proceso se repite rítmicamente, esta secuencia es conocida como ritmo cardiaco, es un movimiento involuntario, es decir se realiza de forma automática [6].

El miocardio está conformado por fibras musculares cuya contracción garantiza un acortamiento del espacio haciendo la función de "bomba" [7], [8].

El corazón es un músculo excitable y presenta estas dos importantes características:

- Células con la capacidad de contraerse sin necesidad ningún estímulo externo [7].
- Células rítmicas, con la función de mantener una frecuencia de contracción constante y armónica [7].

[1] **Fuente:** https://quizlet.com/233943705/partes-del-corazon-diagram/ Última visita a la página: 12/Julio/2019

Propiedades eléctricas

Los miocardiocitos[2] no requieren la presencia de un estímulo externo para generar movimiento como cualquier otro músculo del cuerpo, esta capacidad se denomina "ritmicidad miogénica" [8].

Existen dos tipos de fibras cardiacas con propiedades eléctricas:

- Fibras de respuesta lenta: se caracterizan por generar y conducir el potencial de acción[3] [7].
- Fibras de respuesta rápida: requieren un estímulo externo para su funcionamiento [7].

Potencial de Membrana

El miocito[4] está formado por una membrana celular, esta membrana confiere una resistencia al paso de diversos iones, sin embargo en la membrana existen algunas proteínas que permiten el paso de iones, si se midiera la diferencia de potencial entre el interior y el tejido extracelular de una célula, habrá un potencial negativo, a este valor se le conoce como Potencial de Membrana, que durante la diástole también se le conoce como potencial de reposo [9].

Potencial de acción

En el corazón este fenómeno se inicia por corrientes eléctricas que provienen de células contiguas o puede ser generada de manera espontánea, el potencial presente en las fibras cardíacas puede variar entre 150 y 300 milisegundos, este periodo de tiempo asegura que el músculo no se pueda re-excitar en ningún momento excepto al final de cada ciclo [8], [9].

El potencial de acción cardíaco es diferente en cada porción del corazón y depende de la cinética de sus canales que tengan las células en esa región específica [8], [10].

[2] **Miocardiocitos:** células del músculo cardíaco capaces de contraerse de forma involuntaria.
[3] **Potencial de Acción:** Polarización y despolarización súbita de una célula.
[4] **Miocito:** célula del tejido muscular.

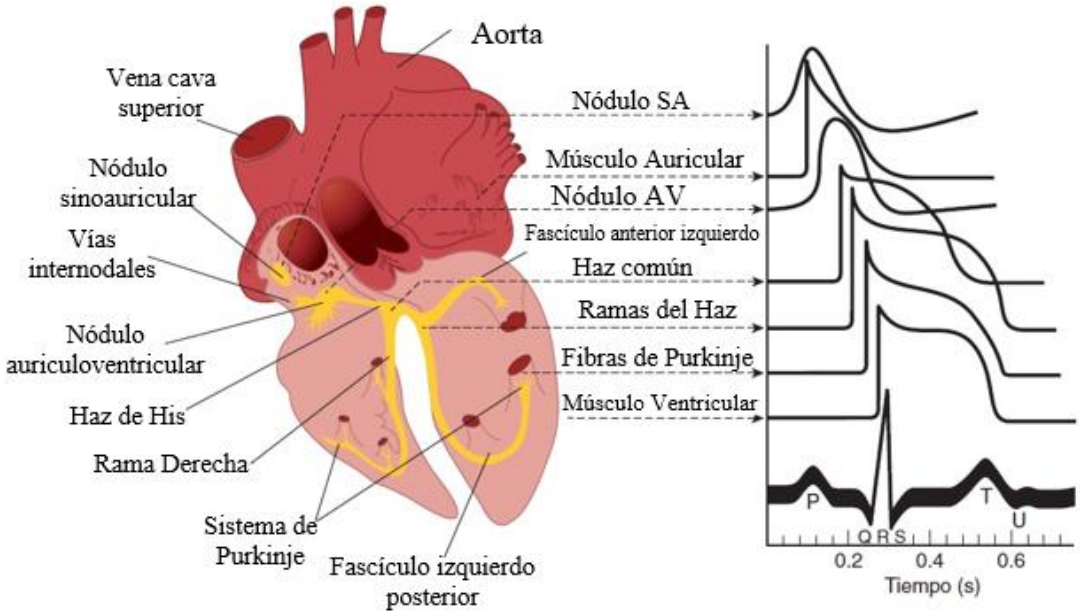


Figura 2 Morfología de los potenciales de acción dependiendo de la región anatómica en que se registran [10].

1.3 ELECTROCARDIOGRAMA

Como ya se explicó en los puntos anteriores el corazón genera un potencial de acción en las diferentes partes de las fibras miocárdicas, la suma algebraica de cada potencial de acción se lo representa mediante el electrocardiograma (EGC). En la Figura 3 se puede observar la conducción del corazón y el electrocardiograma correspondiente a cada etapa [11].

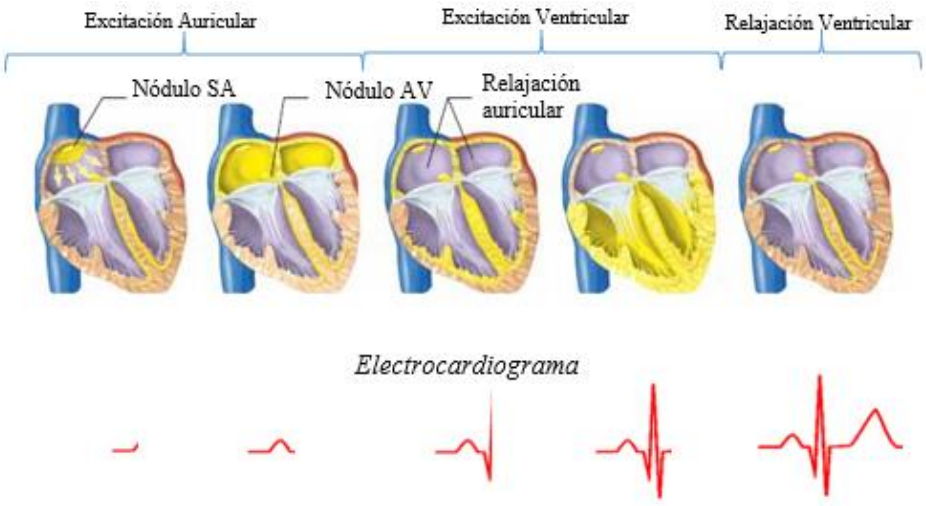


Figura 3 Arriba: Propagación de la actividad Eléctrica del corazón [8]. **Abajo:** Onda producida por la propagación eléctrica del corazón

Un ECG es la representación gráfica y detallada de la toda la actividad eléctrica del corazón para poder visualizarla se utiliza un papel milimétrico en donde el eje vertical se encuentra la amplitud del pulso eléctrico y es medido en [mV][5] cada milímetro (mm[6]) representa 0.1 mV; mientras que el eje horizontal es el tiempo transcurrido y es medido en [s][7] y cada mm representa 0,04s [12]. En la Figura 4 se puede observar la clásica pantalla de un ECG.

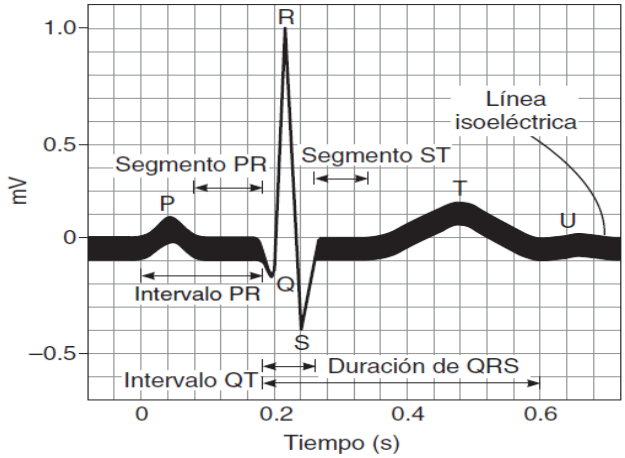


Figura 4 Cuadrícula de un ECG [8].

1.3.1 ONDAS DEL ECG

Es necesario saber que un ECG está formado por diferentes ondas éstas son: P, Q, R, S, T, U que se unen por una línea isoeléctrica [13]. En la Figura 5 se muestra un ECG normal hasta la onda U.

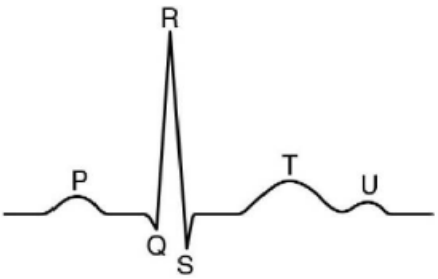


Figura 5 ECG Normal [14].

[5] **mV:** abreviatura de la unidad de medida de voltaje "milivoltio"; 1 mV = 0,001 V.
[6] **mm:** abreviatura de la unidad de medida de distancia 1mm = 0,001 m.
[7] **s:** abreviatura de la unidad de tiempo "segundo".

Onda P

Es el inicio del ciclo cardíaco, la onda P canaliza la contracción de las aurículas[8], está conformada por la adición de toda la actividad eléctrica de las dos aurículas cuya duración es inferior a 0,10 segundos (2,5 mm) y un valor máximo de voltaje de 0,25 mV (2,5 mm) [12], [14].

Onda Q

Generalmente es estrecha y poco profunda aproximadamente 0,04s de ancho y 0,2 mV de amplitud y en general no supera el 25% de amplitud la onda QRS [12], [14].

Complejo QRS

Compuesto por una asociación de ondas que representa contracción de cada ventrículo. El periodo de tiempo de este complejo puede variar entre 0,06s y 0,10s [12], [14].

- **Onda Q:** Se denomina Q si es negativa la primera onda.
- **Onda R:** Es la sección inicial positiva del complejo.
- **Onda S:** Posterior a la onda R existe una onda negativa que se denomina S.
- **Onda QS:** Cuando existe alguna anomalía y el complejo es totalmente negativo se denomina QS.
- **Ondas R' y S':** si hay más de una onda R o S.

Onda T

Indica la expansión de los ventrículos, suele ser de menor amplitud que el complejo QRS, esta onda es asimétrica. Su amplitud máxima llega a 0,5 mV [12], [14].

Onda U

Habitualmente esta onda es positiva, de escaso voltaje y con origen desconocido, se teoriza que indica la repolarización de los músculos papilares [12], [14].

[8] **Contracción de las aurículas:** se dice también que es una despolarización de las aurículas.

1.3.2 INTERVALOS Y SEGMENTOS DEL ECG

Para comenzar a enumerar y describir los intervalos y segmentos debemos saber diferenciarlos.

- **Segmento electrocardiográfico:** es la línea ubicada entre una onda y otra.
- **Intervalo electrocardiográfico:** la parte del ECG que contiene una o más ondas.

Intervalo Rv-R

Representa la distancia que existe entre dos ondas R contiguas. Generalmente, en el ritmo sinusal[9], esta distancia es constante y su duración es determinada por la frecuencia cardiaca [15], [16].

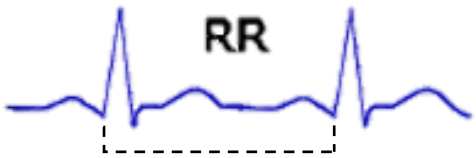


Figura 6 Intervalo Rv-R [12].

Intervalo PR

Indica la contracción de la aurícula y el retraso fisiológico que sucede en este proceso. Comienza con el comienzo de P hasta que las ondas Q o R inicien. Tiene un valor normal que puede variar entre 0.12 segundos y 0.20s [15], [16].

Figura 7 Intervalo PR [12].

[9] **Ritmo sinusal:** ritmo normal del corazón.

Intervalo QRS

Es el tiempo en el que los ventrículos permanecen contraídos. Su duración es desde el comienzo de la onda Q hasta el final de la onda S. Su valor normal es entre 0.06 s y 0.10 s [15], [16].

Figura 8 Intervalo QRS [12].

Intervalo QT

Determina la sístole eléctrica ventricular, es decir, representa cuando los ventrículos entran en despolarización y repolarización, el intervalo QT está entre los 340ms y 440ms en adultos jóvenes y hasta 460ms en mujeres adultas [15], [16].

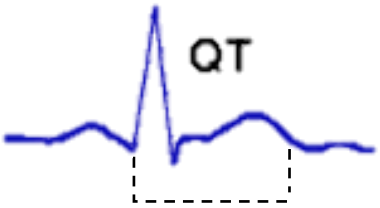


Figura 9 Intervalo QT [12].

Segmento ST

Indica el comienzo de la expansión ventricular y se corresponde con la fase de repolarización lenta de los miocitos ventriculares, su duración se mide desde el final del complejo QRS hasta el inicio de la onda T [15], [16].

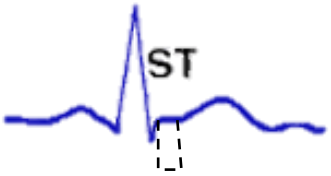


Figura 10 Segmento ST [12].

1.3.3 FRECUENCIA CARDÍACA A PARTIR DEL ECG

Es el inverso del periodo de tiempo entre dos latidos cardíacos consecutivos, este tiempo puede ser medido entre el intervalo Rv-R véase Figura 11. Si el intervalo de tiempo entre dos latidos es de 1s, la frecuencia cardíaca es de 60 latidos/min [17]. Un ritmo sinusal normal está en el rango entre 60 a 100 latidos/min.

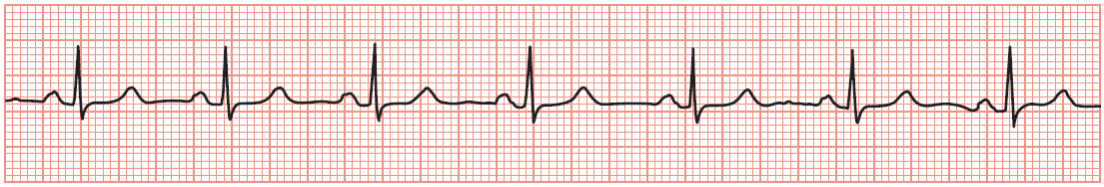


Figura 11 ECG con ritmo sinusal normal [18].

1.3.4 DERIVACIONES

El dipolo cardiaco es una expresión que permite calcular el potencial del corazón en cualquier punto. Si referimos los dipolos cardiacos a todos los instantes de tiempo a un origen común, sus extremos describirán una curva alabeada llamada vectocardiograma[10], correspondientes a un ciclo cardiaco. Mediante unos electrodos en la superficie de la piel, se puede captar la proyección del vectocardiograma y por tanto visualizarlo desde distintas perspectivas [19].

Con distintas ubicaciones de electrodos se pueden captar distintas proyecciones del vector cardiográfico. Se pueden dividir en: derivaciones estándar o llamado también el triángulo de Einthoven11, derivaciones aumentadas y derivaciones precordiales; cada derivación puede explicarse de la siguiente manera:

- **Derivación estándar I:** Es la diferencia de potencial que existe entre el brazo derecho y el brazo izquierdo.
- **Derivación estándar II:** Es la diferencia de potencial que existe entre el brazo derecho. y el pie izquierdo.

[10] **Vectocardiograma:** Registro gráfico de la dirección y magnitud de las fuerzas eléctricas generadas por la actividad del corazón.
[11] **Triángulo de Einthoven:** es una representación gráfica de las derivaciones frontales del electrocardiograma.

- **Derivación estándar III:** Es la diferencia de potencial que existe entre el brazo izquierdo y el pie izquierdo.

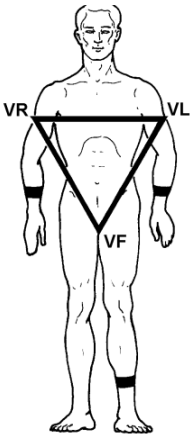


Figura 12 Derivaciones estándar [20].

- **Derivación aumentada aVR:** Es la diferencia de potencial entre la tensión promediada del pie izquierdo y el brazo izquierdo y el brazo derecho.
- **Derivación aumentada aVL:** Es la diferencia de potencial entre la tensión promediada del pie izquierdo y el brazo derecho y el brazo izquierdo.
- **Derivación aumentada aVF:** Es la diferencia de potencial entre la tensión promediada del brazo izquierdo y el brazo derecho con el pie izquierdo.

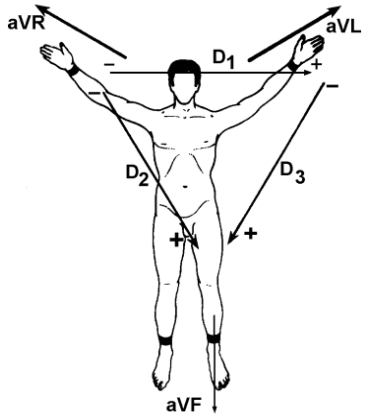


Figura 13 Derivaciones Aumentadas [20].

Las derivaciones precordiales son la diferencia de tensión entre los electrodos situados en el tórax, y el terminal negativo que se lo conoce como CTW (Central Terminal of Wilson[12]) [19].

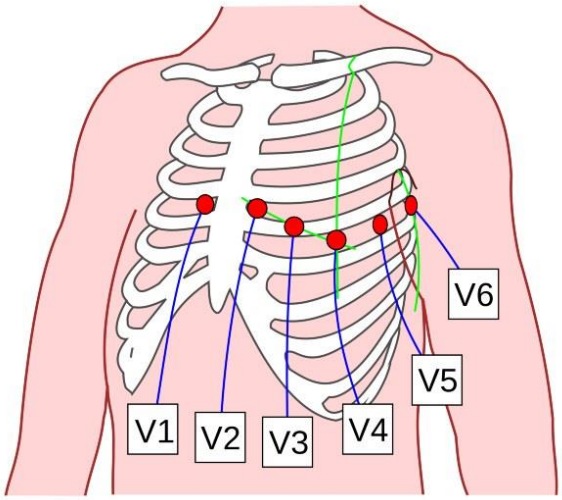


Figura 14 Derivaciones precordiales ubicadas en el tórax (V1, V2, V3, V4, V5, V6) [21].

1.4 ARRITMIAS Y PATOLOGÍAS CARDÍACAS

Anteriormente se describió la propagación eléctrica del músculo cardíaco, el ritmo cardíaco se lo conoce como "ritmo sinusal normal" éste se caracteriza por tener una frecuencia cardíaca entre los 60 y 100 latidos por minuto (lpm), la onda cardíaca compuesta por las diferentes ondas P, Q, R, S, T, U vistas en el apartado 1.3.1.

El ritmo cardíaco puede verse afectado por diferentes anomalías en la amplitud o en la frecuencia cardíaca, éstas anomalías se denominan "arritmia[13]" , esto se debe a diversas patologías que impiden que el impulso eléctrico generado en el corazón se propague de una manera correcta [18].

Las arritmias cardíacas pueden presentarse en diferentes zonas del músculo cardíaco, las arritmias más comunes son las siguientes:

- **Arritmias con ritmo sinusal:** Pueden ser taquicardias sobre los 180 lpm o bradicardia inferior a 40 lpm.

[12] **Central Terminal of Wilson:** tensión promediada entre el brazo izquierdo, brazo derecho y el pie izquierdo.
[13] **Arritmia:** Falta de regularidad o alteración de la frecuencia de los latidos del corazón.

- **Arritmias supraventriculares:** Se genera cuando el latido no se origina en el nodo SA[14].
- **Arritmias ventriculares:** Se dan cuando tiene un ritmo cardíaco no constante
- **Bloqueos:** Son demoras o interrupciones entre cada ciclo cardíaco.

1.5 ESTADO DEL ARTE DEL PROCESAMIENTO DE SEÑALES ECG

Existen diferentes normativas referentes a la seguridad, rendimiento, compatibilidad electromagnética de los equipos y sistemas eléctricos médicos.

En la mayor parte de latinoamérica existe poco interés en desarrollar diferentes tecnologías que puedan aportar algo a esta gran problemática sin embargo, muchas universidades a nivel mundial poseen algún grupo de investigación dedicado al estudio de la electrocardiografía, algunas de las revistas que publican sobre el procesamiento de señales biomédicas son las siguientes:

- Scielo (http://www.scielo.org/php/index.php)
- Revista Ingeniería Biomédica, Medellín – Colombia. (http://revistabme.eia.edu.co/es.html)
- Cuban Health, Centro de Ingeniería Genética y Biotecnología (CIGB). (http://www.cubanhealth.com/)
- Revista Cubana de Investigaciones Biomédicas. (http://bvs.sld.cu/revistas/ibi/indice.html)
- CIBER-BBN, Centro Investigación Biomédica en Red, Bioingeniería, Biomateriales y Nanomedicina. (http://bbn.ciberbbn.es/grupos/detail?id=115&locale=es&show=lineas)
- Corience, Plataforma Europea Independiente sobre cardiopatías congénitas. (http://www.corience.org/es/)
- CREB, Centro de Investigación en Ingeniería Biomédica de la Universidad Politécnica de Cataluña (UPC). (http://www.creb.upc.es/es)

[14] **Nodo SA:** Nodo Sinoauricular, genera una señal eléctrica que permite la contracción (latido) del corazón, llamado “marcapasos natural”

- CTB, Centro de Tecnología Biomédica de la Universidad Politécnica de Madrid (UPM). (http://www.ctb.upm.es/)

Actualmente se están desarrollando e implementando dispositivos de monitorización ECG que son completamente portátiles y autónomos, de las tecnologías más llamativas son los productos que son compatibles con smartphones que utilizan el protocolo de comunicación bluetooth.

1.5.1 QARDIOCORE

Es un dispositivo que posee dos bandas que se colocan sobre el pecho y mediante Bluetooth se visualiza la señal ECG en la aplicación móvil, en la Figura 15 se muestra el dispositivo QARDIOCORE con su APP móvil y su respectiva banda [22].

Algunas de las características de este dispositivo son:

- Respuesta en frecuencia entre 0,05 Hz y 40 Hz.
- 600 muestras por segundo.
- Resolución de muestreo: 16 bits
- Bluetooth 4.0
- Precio (oficial): 499€

Figura 15 Dispositivo QARDIOCORE [22].

1.5.2 KARDIA

Esta banda del tamaño de una tarjeta de crédito, como se puede observar en la Figura 16 juntamente con la aplicación para smartphones o, en este caso también, para relojes inteligentes (de momento solamente Apple Watch) registra el ECG en

30 segundos y lo transmite al dispositivo. Está distribuido por la marca AliveCor por un precio de 149 €. Mide los impulsos a partir de los dedos colocados en las almohadillas y está certificado para uso pediátrico en Europa [23].

Figura 16 Banda de detección KARDIA [23].

1.5.3 SISTEMA DE ECG HOLTER BASADO EN PC - CC-HOLTER

Es un sistema con un costo aproximado de $2875,00. En la Figura 17 se muestra una imagen del sistema.

El sistema de ECG Holter basado en PC Nasiff CardioCard con redes integradas, base de datos ilimitada, registrador digital de 24 a 96 horas con detección de marcapasos, pantalla de 3 canales para obtener una vista previa de la forma de onda del ECG [24].

Figura 17 ECG Holter basado en PC - CC-HOLTER [24].

1.5.4 PC NASIF CARDIO SUITE STRESS ECG Y HOLTER MONITOR

Es un sistema con un costo aproximado a $5790.00. En la Figura 18 se muestra una imagen de dicho sistema [24].

Figura 18 PC Nasif Cardio Suite Stress ECG y Holter Monitor [24].

CAPÍTULO 2: MARCO METODOLÓGICO

El prototipo ser realizó en 3 etapas. En primer lugar, la elección de los circuitos integrados y entornos de programación; como segunda etapa, la configuración del ADAS1000; y la tercera etapa, la comunicación entre el microcontrolador con la computadora y con el smartphone.

2.1 ESTRUCTURA Y FUNCIONAMIENTO DEL ADAS1000

Para el diseño de este prototipo se utilizó el circuito integrado de Analog Devices ADAS1000. Este circuito integrado está diseñado para simplificar la adquisición de señales biomédicas, específicamente de señales electrocardiográficas y señales de respiración a través de electrodos superficiales. Opera con una frecuencia de 8.192MHz proporcionada por un oscilador externo conectado a los pines respectivos.

El ADAS1000 posee varios pines digitales que aseguran la flexibilidad en la comunicación con microcontroladores y de esta manera realizar un procesamiento posterior de la señar cardíaca adquirida.

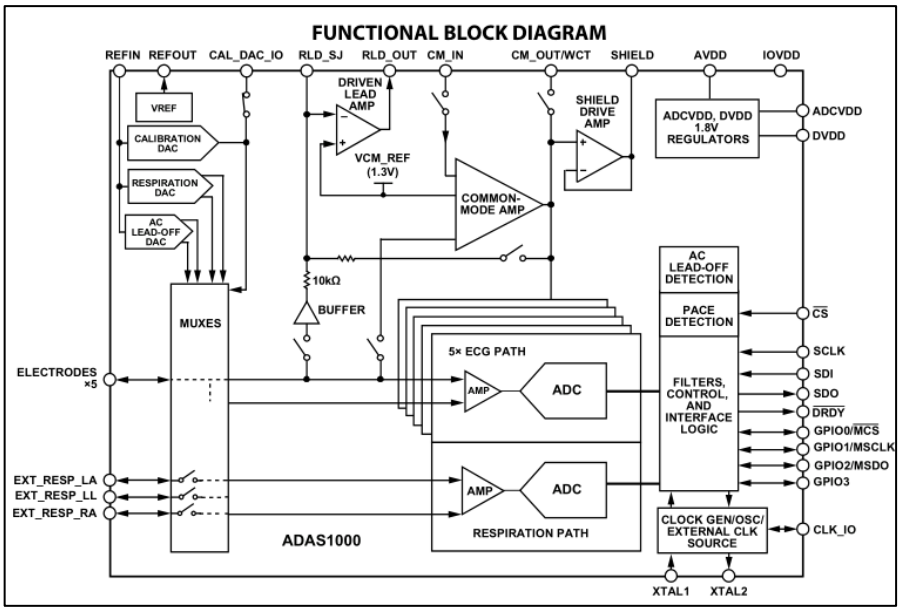


Figura 19 Esquema de funcionamiento del ADAS1000 [25].

La estructura de entrada del ADAS1000 es un amplificador diferencial, esto permite a los usuarios elegir entre una variedad de opciones de configuración que mejor se adapte a su aplicación.

También posee 5 entradas analógicas (conexión de los electrodos LA, LL, RA, V1, V2) usadas para la adquisición de las bioseñales, además de 6 entradas para medir el ritmo de respiración, y por último una unidad de referencia (RLD).

Analog Devices propone un ejemplo de conexión al cual se debe regir para su correcto funcionamiento véase Figura 20.

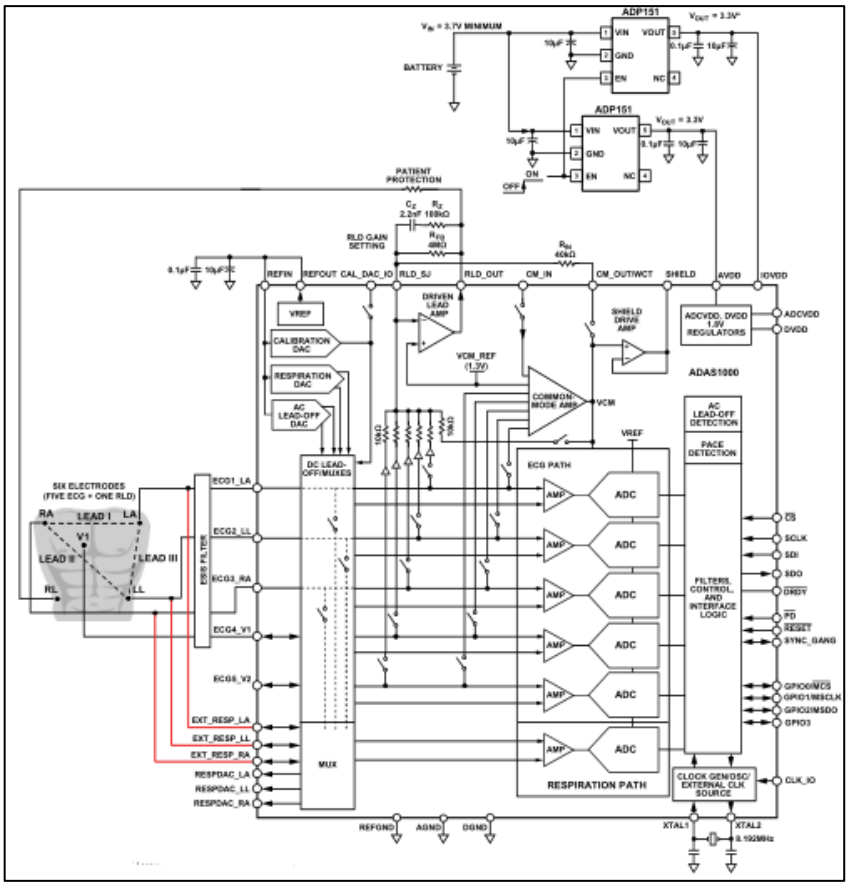


Figura 20 Esquema propuesto por Analog Device [25].

2.1.1 CONEXIÓN DE LOS ELECTRODOS

La conexión de los electrodos se realiza directamente en las entradas correspondientes. Si el dispositivo se va a usar en conjunto con un desfibrilador, se recomienda conectar una protección externa entre el electrodo y el pin de conexión, que puede ser unas bombillas de neón, las cuales conducen la corriente a masa para impedir afectar al dispositivo. En este proyecto no se usará esta protección sino directamente los electrodos al ADAS1000.

2.1.2 CANALES DE ECG

Cada canal de ECG en el interior del ADAS1000 está formado por:

- Diodos que sirven para protección de altas tensiones y corrientes elevadas.
- Un amplificador operacional de bajo ruido con ganancia programable.

- Un filtro antialiasing con ganancia fija.
- Un buffer de datos.
- Un conversor analógico digital de 14 bits.

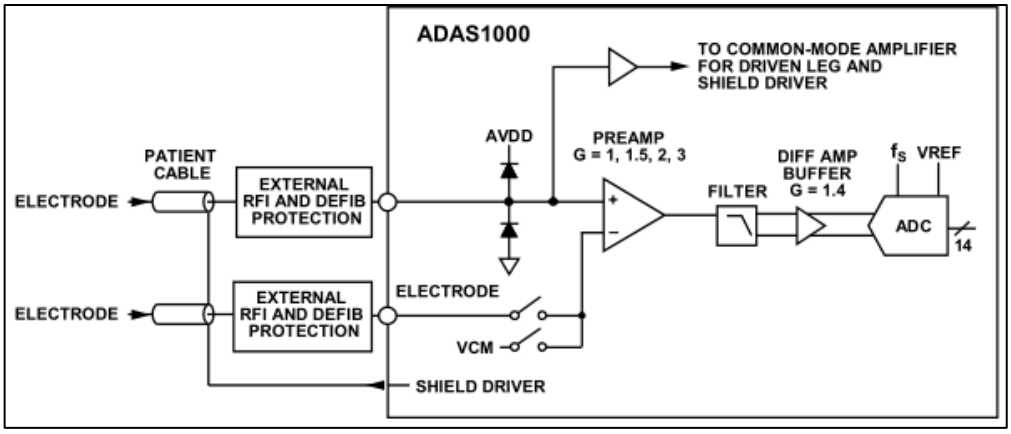


Figura 21 Canales internos del ADAS1000 [25].

2.1.3 SELECTOR DE MODO COMÚN

La señal de modo común se puede derivar de cualquier combinación de una o más de las entradas de los electrodos, de la referencia interna fija de modo común o de una fuente externa conectada al terminal correspondiente, Esto es útil en la medición de señales de calibración del ADAS1000 o cuando se pretende recoger la señal de un solo electrodo.

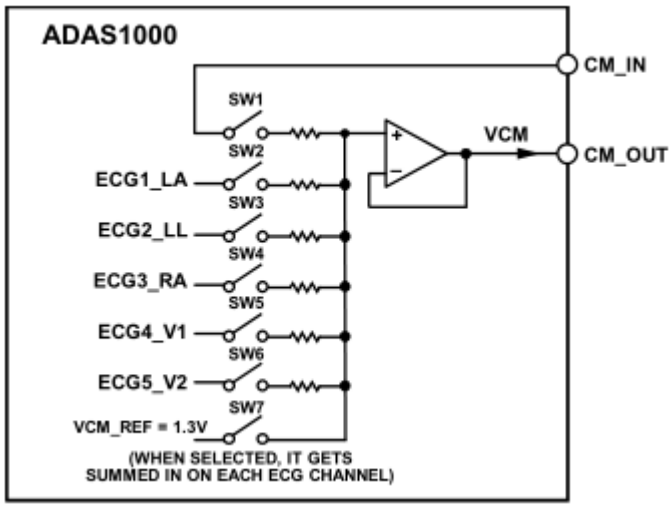


Figura 22 Selector de modo común [25].

2.1.4 DERIVADA DE REFERENCIA

La derivada de referencia o de pierna derecha se utiliza como realimentación del circuito. Con esto se genera una tensión de modo común en el paciente lo más cerca posible al nivel interno de voltaje del ADAS1000. También sirve para la reducción de ruido generado por otros instrumentos conectados al paciente.

2.1.5 FILTRADO

El ADAS1000 tiene una etapa de filtrado digital antes de enviar la señal. Existen 4 filtros de paso bajo con frecuencia seleccionable: 40 Hz, 150 Hz, 250 Hz y 7 Hz usado para calibración.

2.2 DSPIC30F4013

Es un controlador digital de señales que trabaja a 16 bits, adecuado para varias aplicaciones de propósito general y de audio, con una memoria de programa de 48 Kb, un conversor analógico-digital de 12 bits y comunicaciones: I2C, SPI, CAN y 2 UARTs. Este dispositivo fue elegido para el prototipo debido a su sencillo manejo en la programación y comunicación con distintos periféricos

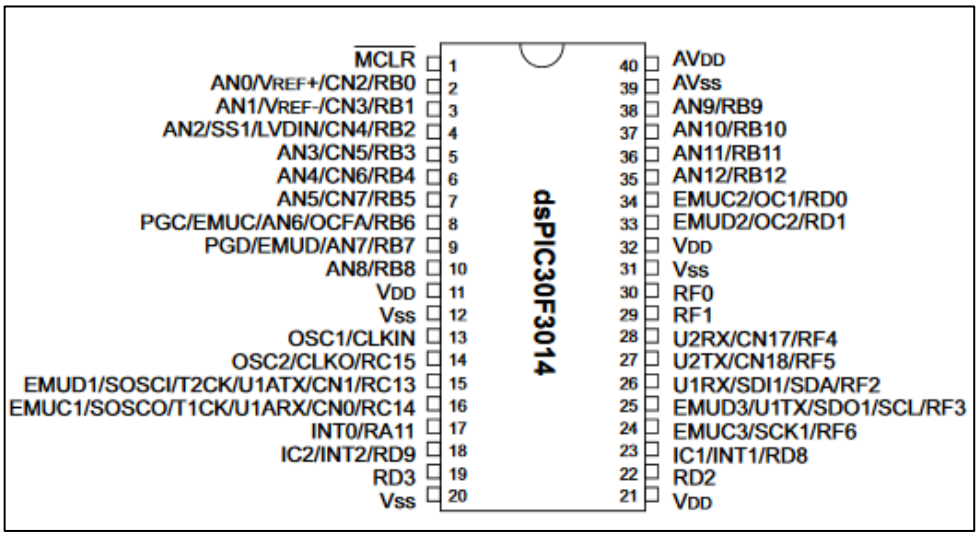


Figura 23 Diagrama de pines de dsPIC30F4014 [26].

2.3 FIRMWARE

Para la programación de los diferentes circuitos integrados se divide en 3 partes: comunicación entre DSPIC y ADAS1000, comunicación entre DSPIC y PC, comunicación entre DSPIC y SMARTPHONE.

El compilador utilizado para la programación del DSPIC es MPLAB XC16, aquí se crearon las librerías y funciones respectivas para configurar y controlar el ADAS1000. Para crear la aplicación de escritorio en Windows se utilizó Visual Studio 2015 en el lenguaje C# (sharp); y para la aplicación para smartphone Android se utilizó Android Studio.

2.3.1 PROTOCOLO DE COMUNICACIÓN ENTRE DSPIC Y ADAS1000

La comunicación entre DSPIC y ADAS1000 se realizó mediante SPI, siendo el DSPIC el maestro y el ADAS1000 el esclavo. El ADAS1000 se controla por medio de registros de 32 bits. Debido a que el DSPIC30F4013 posee un SPI de 8 bits y el ADAS1000 trabaja con palabras de 32 bits, se estableció un protocolo de comunicación entre estos dos dispositivos que consiste en enviar 4 palabras de 8 bits para cada instrucción.

La trama que acepta el ADAS1000 se muestra en la Tabla 1:

Tabla 1 Trama que recibe ADAS1000

Bits	31	30:24	23:0
Función	Bit de lectura o escritura	Bits de dirección	Bits de datos (MSB primero)

El bit 31 indica al dispositivo si la instrucción va a ser de escritura o de lectura. Los bits del 30 al 24 especifican la dirección de la instrucción y los bits restantes indican la instrucción a ejecutarse.

El ADAS1000 posee varios registros para ser configurado de muchas maneras, para este proyecto solo se utilizarán los siguientes registros: CMREFCTL, FRMCTL, ECGCTL y FRAMES.

CMREFCTL: configura el modo común y la referencia.

FRMCTL: configura y controla la cabecera de datos.

ECGCTL: controla el ECG.

FRAMES: registro para empezar a transmitir.

Para este prototipo se configurará el ADAS de la siguiente manera:

- 5 electrodos
- Ganancia de x1.4
- Modo común con todos los electrodos

El ADAS1000 posee un pin CS (chip select), activo a bajo, para ser activado o desactivado por el maestro. Cada que se requiera enviar un registro desde el microcontrolador al ADAS, se establece un cero lógico a este pin.

También se tiene un pin READY para indicar que el esclavo está listo para enviar datos, el microcontrolador por tanto debe ser capaz de leer este pin para saber cuan es posible recibir los datos desde el ADAS1000

La trama de salida desde el ADAS1000 es la siguiente:

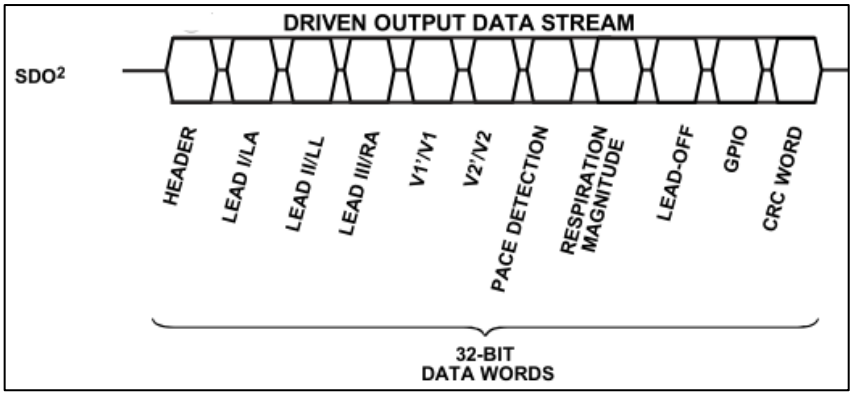


Figura 24 Trama de salida de datos del ADAS1000 [25].

Esta trama fue modificada al configurar el registro FRMCTL de esta forma se reduce a:

- HEADER
- LA
- LL
- RA

- V1
- V2
- Detección de pulso

Cabe recalcar que cada palabra es de 32 bits de dimensión, por lo tanto, cada dato se guardará en 4 variables de 8 bits para ser enviados por comunicación UART hacia la PC o un SMARTHPHONE.

2.3.2 PROTOCOLO DE COMUNICACIÓN ENTRE DSPIC Y PC

Al recibir los datos desde el ADAS1000 se los guarda en variables de 8 bits, estos datos se envían por UART hacia la PC.

Se desarrolló una aplicación para Windows que nos permita conectarnos de manera serial a nuestro prototipo, recibir los datos y graficar las señales obtenidas.

Para el desarrollo de este software se utilizó Visual Studio, en la Figura 27 se muestra la interfaz de la aplicación de escritorio.

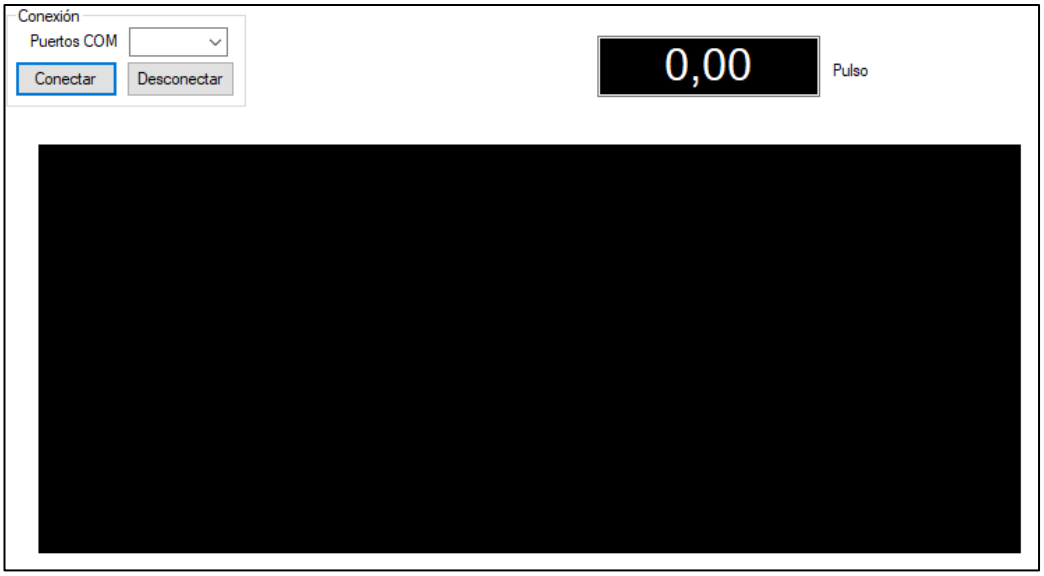


Figura 25 Aplicación para Windows desarrollada en Visual Studio

2.3.3 COMUNICACIÓN ENTRE DSPIC Y SMARTPHONE

Para la comunicación entre el dsPIC y un teléfono celular Android utilizamos el protocolo de comunicación UART que se conecta a un módulo bluetooth HC-05 y desarrollamos una APP en Android Studio para la visualización de la señal cardiaca.

Para poder recibir los datos en el smartphone, la APP realiza un escaneo de los dispositivos disponibles, hecho esto ya estarán emparejados el prototipo con el smartphone, solamente pedirá una clave la primera vez que se conecte, esta clave es: "1234".

Cada vez que el dsPIC termine de adquirir una trama de datos, se procede a enviar dicha trama por el puerto serial que estará conectado al módulo Bluetooth. La tecnología Bluetooth presenta las siguientes ventajas sobre otras tecnologías:

- El 100% teléfonos celulares inteligentes poseen tecnología Bluetooth.
- Los teléfonos celulares no requieren de permisos especiales o pagos extra para acceder a la tecnología Bluetooth, además siempre y cuando tengamos en cuenta la duración de la batería del teléfono celular podremos estar enlazados el tiempo que lo requiera.
- No representa costos adicionales para el usuario por usar esta tecnología.

Para lograr la transmisión, utilizamos el módulo UART[15] del microcontrolador dsPIC30F4013, el módulo UART comúnmente se lo conoce como "puerto serial". Este módulo es el encargado de enviar y/o recibir datos en el microcontrolador; el envío o recepción de datos se produce un bit tras otro [26]. En la Figura 26 podemos observar un diagrama de bloques del módulo USART del dsPIC utilizado.

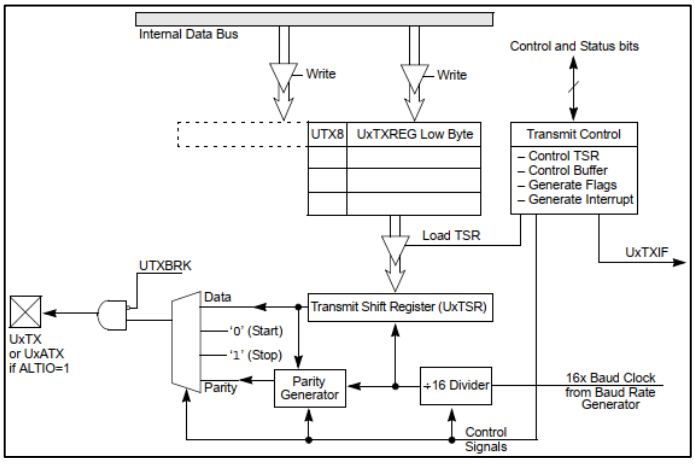


Figura 26 Diagrama de bloques de transmisión de UART [26].

[15] **UART:** *Universal Asynchronous Receiver Transmitter*

Para poder utilizar de manera correcta este protocolo de comunicación y conectarlo con el módulo HC-05 (véase Figura 27) se debe tomar en cuenta algunos aspectos que son:

- **Tasa de baudios:** indica la cantidad de símbolos que se puede transmitir en una comunicación serial.
- **Tamaño del dato:** El protocolo de comunicación USART soporta una trama de 8bits para el envío o recepción de datos, además en la hoja de datos nos indica que el bit menos significativo es el que se transmite en primer lugar [25].

Figura 27 Imagen de referencia de módulo Bluetooth HC-05[16].

Para el desarrollo de la APP para el smartphone se utilizó Android Studio, el código utilizado en su mayoría es JAVA, la parte de la interfaz gráfica puede ser creado con las herramientas que nos ofrece Android Studio o mediante el lenguaje de programación XML.

Una vez decidido el entorno de programación deberemos buscar las librerías necesarias que se utilizarán para el desarrollo de la APP, utilizamos solamente una única librería para la gráfica en tiempo real, se utilizó la librería A Chart Engine (ACE)[17], ya que esta tiene varias ventajas sobre otras librerías, una de ellas es la facilidad de uso y compatibilidad con el Bluetooth.

[16] **HC-05:** Imagen de referencia del módulo Bluetooth tomado de: https://www.geekfactory.mx/tienda/radiofrecuencia/hc-05-modulo-bluetooth-maestro-esclavo/

[17] **A Chart Engine (ACE):** Enlace de descarga y documentación: https://github.com/ddanny/achartengine

La APP posee dos ventanas principales, la primera en donde se muestra los dispositivos BT vinculados con el smartphone (véase Figura 28) y la segunda ventana en donde se va a graficar la señal ECG (véase Figura 29).

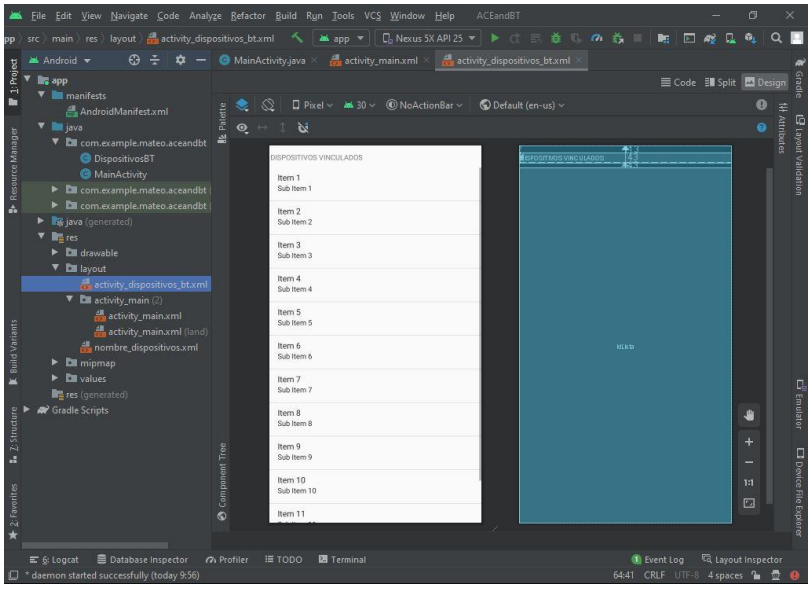


Figura 28 Vista Design + Blueprint de Android Studio del layour Dispositivos BT

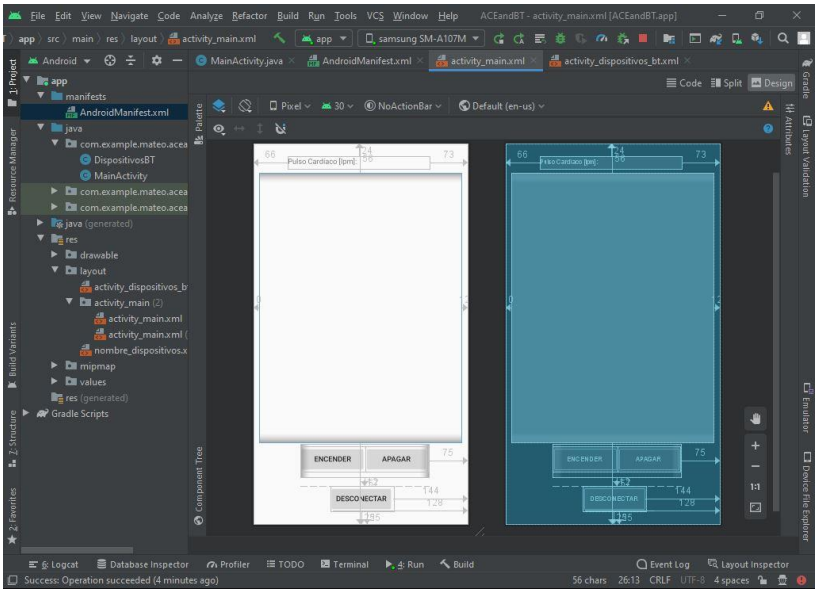


Figura 29 Vista Design + Blueprint de Android Studio del layout Monitor ECG

CAPÍTULO 3: IMPLEMENTACIÓN Y ANÁLISIS DE RESULTADOS

La implementación de este proyecto se dividió en varias etapas, desde el diseño, pruebas en Project Board, realización de la placa, diseño de las APP tanto para Windows como para Android las cuales serán explicadas a continuación.

3.1 PCB EN ALTIUM DESIGNER

Para el diseño del PCB del prototipo seguimos las recomendaciones proporcionadas por el fabricante del ADAS1000 ya que, la parte digital puede introducir fácilmente ruido en la parte analógica y afectar en gran medida la calidad de la señal. La señal analógica, por tratarse de una señal fisiológica en muy sensible al ruido, por lo que pequeñas variaciones pueden afectar a la señal original. Por otra parte, la electrónica de un circuito integrado digital (reloj de la CPU y otros componentes) puede interferir en la parte analógica, el fabricante recomienda separar lo más posible estas dos partes.

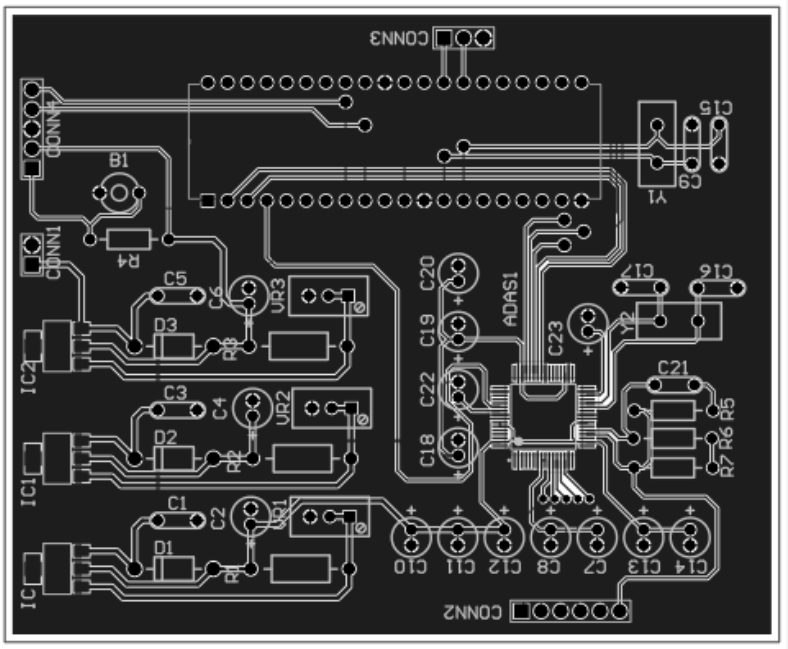


Figura 30 PCB diseñado en Altium Designer, el diseño se puede observar en el Apéndice A, página 42

El fabricante recomienda un arreglo de capacitores y resistencias ubicados lo más cerca posible del ADAS1000, los valores de estos componentes vienen especificados en su hoja de datos.

Finalmente, con el PCB fabricado y para facilitar el manejo y manipulación del prototipo se elaboró un case de madera adaptado para las diferentes conexiones (véase Apéndice B).

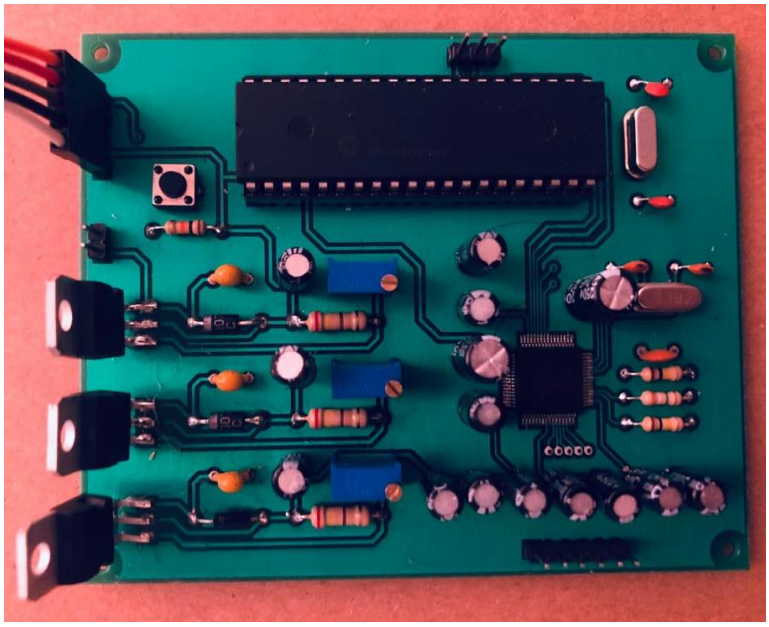


Figura 31 Placa física terminada

3.2 APLICACIÓN PARA WINDOWS

En primer lugar, el prototipo debe conectarse con el ordenador, para ello, se debe seleccionar el puerto al que está conectado:

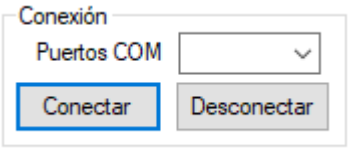


Figura 32 Seleccionar el puerto al que está conectado el prototipo

Al presionar en "Conectar" comenzará la transmisión de los datos y posteriormente se empezará a graficar la señal cardíaca.

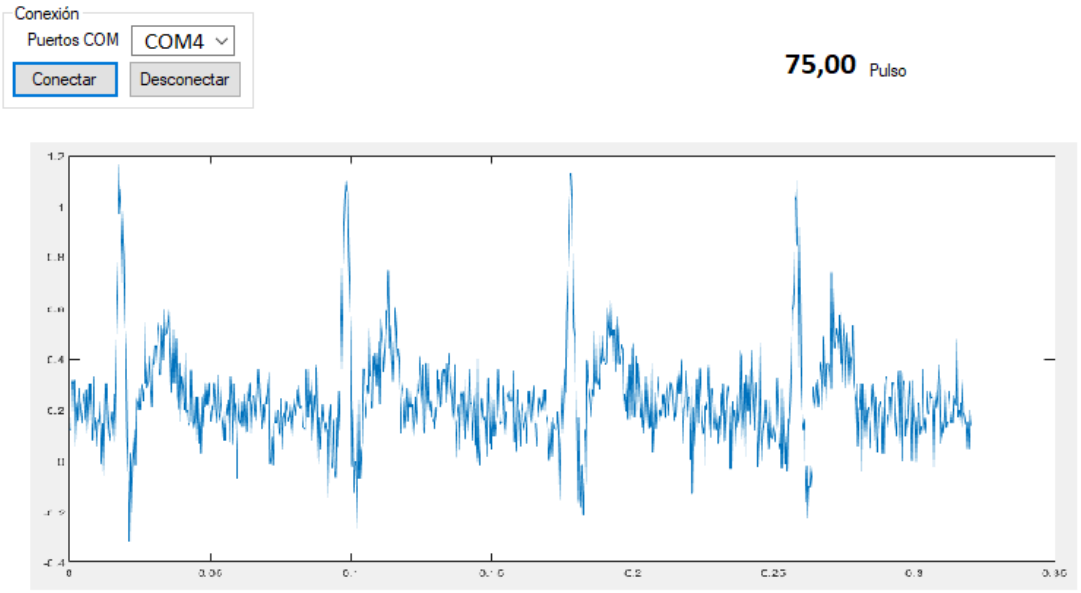


Figura 33 Gráfica de la señal cardíaca

3.3 APLICACIÓN PARA ANDROID

Para la Aplicación en Android se debe instalar un APK en el smartphone la aplicación tiene el nombre "Monitor ECG", para poder usarla necesitamos un dispositivo Android 5.0 o superior, adicionalmente debemos asegurarnos que el teléfono celular esté funcionando correctamente incluyendo la tecnología Bluetooth.

En el Apéndice D se muestra el manual de usuario, en el cual se explica el proceso de instalación y la manera correcta de poner en marcha la APP, en la Figura 34 se muestra la señal obtenida en esta APP.

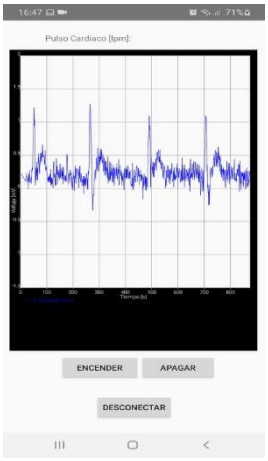


Figura 34 Señal ECG obtenida en APP Android**.**

3.4 RESULTADOS DE LAS SEÑALES OBTENIDAS

Una vez con el PCB realizado, continuamos con la adquisición de la señal cardíaca, a continuación se muestra diversos resultados obtenidos variando el parámetro de la ganancia en el ADAS1000, para ello se varía el registro ECGCTL, este registro se encuentra explicado en la hoja de datos, tabla 25, página 55 [25].

Para cada ganancia realizamos pruebas con diferentes electrodos para saber si hay variaciones en la señal, pero para este prototipo no hubo cambios ni mejoras significativas en la señal, a continuación, se muestran las señales obtenidas con las diferentes ganancias.

3.4.1 Ganancia 4.2

Señal obtenida con una ganancia en el ADAS1000 de 4.2.

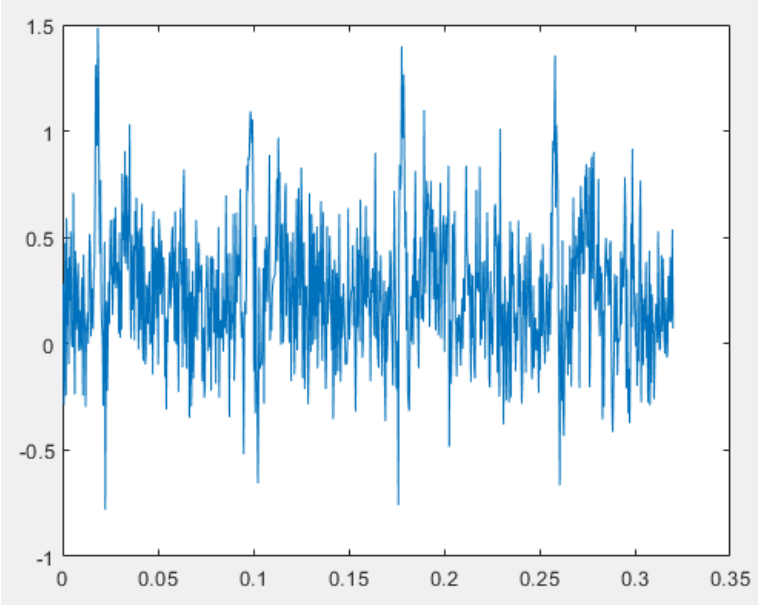


Figura 35 Ganancia en el ADAS1000 de 4.2.

3.4.2 Ganancia 2.8

Señal obtenida con una ganancia en el ADAS1000 de 2.8.

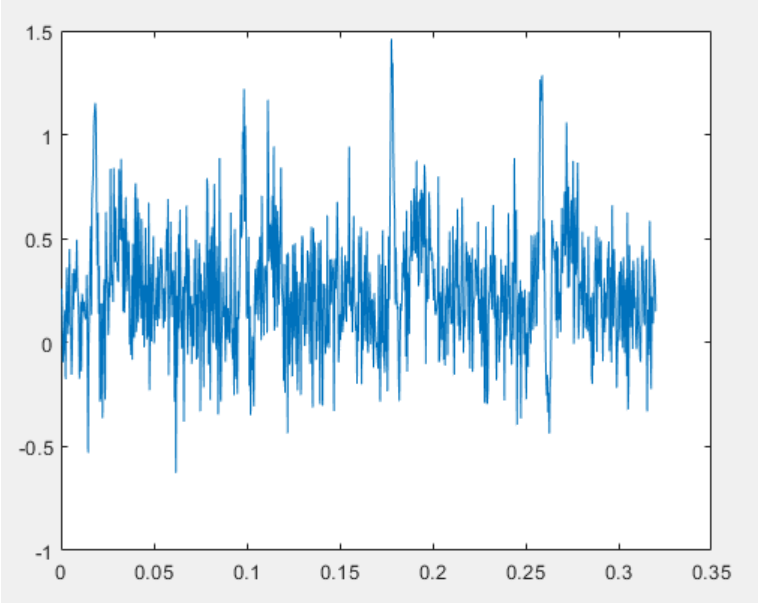


Figura 36 Ganancia en el ADAS1000 de 2.8.

3.4.3 Ganancia 2.1

Señal obtenida con una ganancia en el ADAS1000 de 2.1.

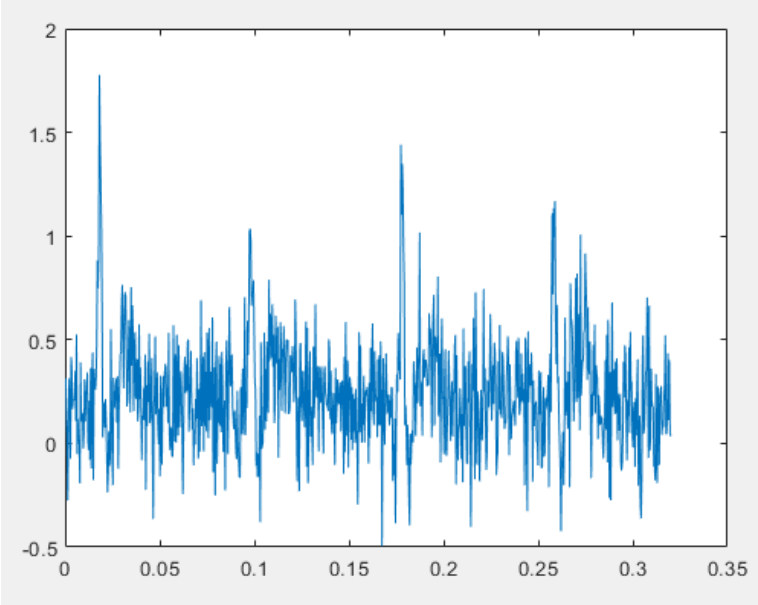


Figura 37 Ganancia en el ADAS1000 de 2.1

3.4.4 Ganancia 1.4

Señal obtenida con una ganancia en el ADAS1000 de 1.4. En esta ganancia es la mejor señal que logramos obtener, llegados a este punto optamos aplicar filtros digitales ya que el ADAS1000 ya posee un filtrado previo, sin embargo, al aplicar dichos filtros la señal pierde la forma característica de una señal cardíaca, por lo que el uso de un filtro digital en esta aplicación no es lo recomendable.

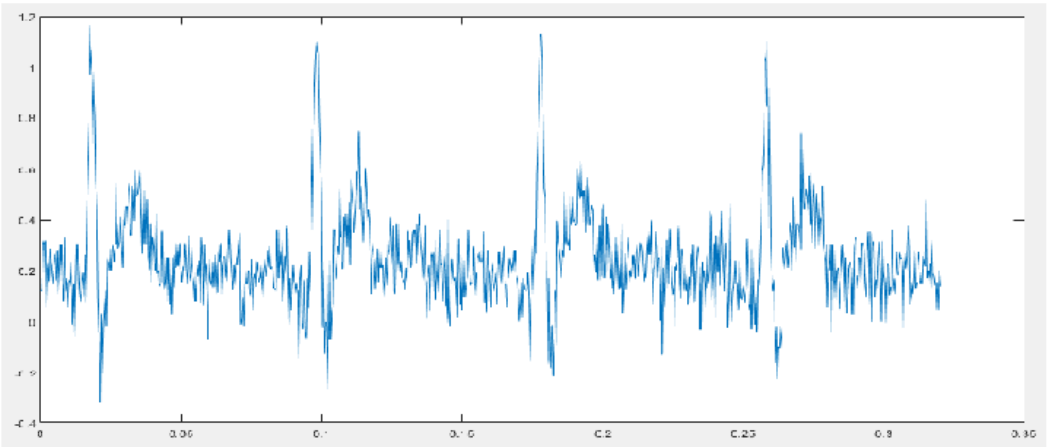


Figura 38 Ganancia en el ADAS1000 de 1.4

3.5 ANÁLISIS DE RENTABILIDAD

3.5.1 Costo de fabricación del prototipo

Para el análisis de costo de fabricación se deberá tomar en cuenta la mano de obra, materia prima directa como indirecta. Considerando que la materia prima directa son todos aquellos elementos que forma parte de la construcción del prototipo. La materia prima indirecta son todos los componentes utilizados para la fabricación del prototipo, pero no se puede cuantificar en el producto terminado.

Tabla 2 Costos materia prima directa.

Materia prima directa			
Componentes	Cantidad	Costo Unitario	Costo Total
ADAS1000	1	$35,00	$35,00
dsPIC30F4013	1	$15,00	$15,00
Regulador de voltaje LM317	2	$0,75	$1,50
Condensadores electrolíticos	15	$0,057	$0,85
Condensadores cerámicos	8	$0,055	$0,44
Diodos	2	$0,053	$0,11
Resistencias	6	$0,047	$0,28
Pulsante	1	$0,089	$0,09
Zócalo	1	$1,00	$1,00
Peinetas	2	$0,50	$1,00
Protector térmico 2m	1	$1,50	$1,50
Cables de Conexión 2m	1	$2,56	$2,56
Módulo HC-05	1	$10,00	$10,00
Fuente de voltaje	1	$5,75	$5,75
Paquete de electrodos	1	$20,00	$20,00
Set de cables	1	$20,00	$20,00
Fabricación Case con corte láser	1	$10,00	$10,00
Fabricación de PCB	1	$58,00	$58,00
	TOTAL		**$183,07**

Tabla 3 Costos materia prima indirecta.

Materia prima indirecta			
Componentes	Cantidad	Costo Unitario	Costo Total
Estaño por metro	2	$0,09	$0,18
Pasta para soldar	1	$5,00	$5,00
Gastos adicionales	1	$3,00	$3,00
	TOTAL		**$8,18**

En la Tabla 2 se especifican los costos de materia prima directa, mientras que en la Tabla 3 se muestran los costos de materia prima indirecta. Para calcular el costo total del prototipo se emplea la siguiente ecuación:

$$Materia_{prima} = Materia_{Prima\ directa} + Materia_{Prima\ indirecta}$$

$$Materia_{prima} = \$\ 183{,}07 + \$\ 8{,}18$$

$$Materia_{prima} = \$\ 191.25$$

3.5.2 Rentabilidad en bajo costo

Se han establecido los costos indirectos para una duración de seis meses, en estos costos se incluyen los sueldos de programadores, servicios básicos, costos operativos y de uso de equipos como se detalla en la Tabla 4.

Tabla 4 Inversión del prototipo a 6 meses.

Inversión del prototipo 6 meses		
Elementos	Costo	
Sueldo de investigadores	$4.600,00	
Programación	$280,00	
Servicios básicos	$310,00	
Comida	$800,00	
Transporte	$200,00	
Computadora	$650,00	
Inmueble	$200,00	
Pruebas fallidas	$80,00	
TOTAL	$7.120,00	POR UNIDAD
POR MES	$1.186,67	$79,11

De esta manera, se logró establecer un precio de venta del prototipo considerando una utilidad del 20%, quedando un precio de $364,99 por unidad véase Tabla 5.

Tabla 5 Precio de venta del prototipo.

COSTO MATERIA PRIMA	$191,25
COSTOS INDIRECTOS	$79,11
COSTO POR UNIDAD	$270,36
UTILIDAD	0,35
PRECIO DE VENTA	**$364,99**

A partir de estos resultados se propone un análisis de rentabilidad del proyecto para los próximos 6 meses, considerando una inversión inicial de $7120,00, una tasa de aumento mensual del 20% y una tasa de descuento VAN del 12%. Estos valores son considerados como estándar para el análisis de estos proyectos.

Tabla 6 Análisis de rentabilidad.

MESES	0	1	2	3	4	5	6
INGRESOS							
VENTA DISPOSITIVO		5474,87	6569,84	7883,81	9460,57	11352,68	13623,22
EGRESOS							
COSTO DE DISPOSITIVO		4055,46	4866,55	5839,86	7007,83	8409,40	10091,28
ACTIVIDADES DE INVERSION							
FLUJO DE CAJA NETO	7120,00	1419,41	1703,29	2043,95	2452,74	2943,29	3531,95
VAN	1978,28						
TIR	**20%**						

Al observar los valores de VAN y TIR se aprecia que el proyecto cuenta con rentabilidad positiva, obteniendo una ganancia de $1978,28 a los 6 meses si se venden como mínimo 15 productos mensuales.

Al comparar el precio de venta de este prototipo con los precios de los prototipos mencionados en el Capítulo 1, se puede constatar que entra dentro de los prototipos de bajo costo.

3.5.3 Rentabilidad en reducidas dimensiones

El prototipo presentado en este proyecto cuenta con unas medidas de 10 cm de largo, 7 cm de alto y 8 cm de alto. Si bien no es el prototipo más pequeño en comparación con los presentados en el Capítulo 1, sus dimensiones y diseño permiten que sea de manipulación y uso sencillo.

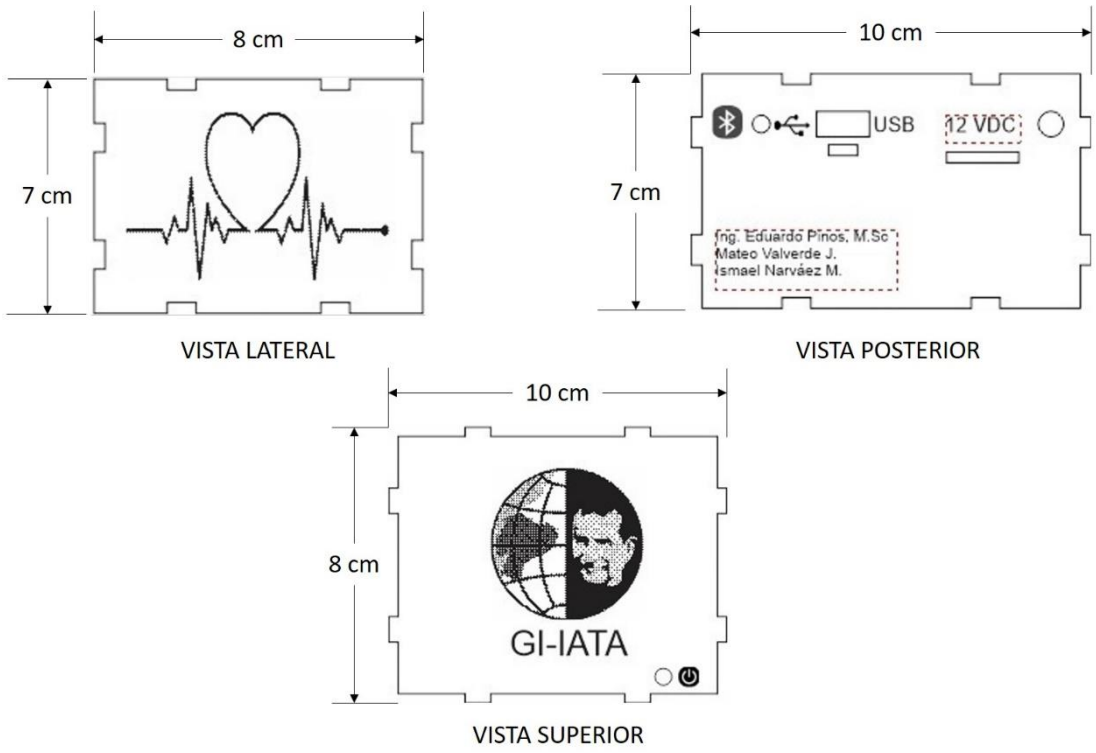


Figura 39 Dimensiones del prototipo

Cabe recalcar que en el PCB que se realizó, se usaron componentes disponibles en nuestro país, la mayoría, componentes de agujero pasante, esto debido al impedimento de importaciones debido a la emergencia sanitaria que atraviesa el mundo. La mayoría de los componentes se los pueden cambiar por componentes de montaje superficial; esto reduciría considerablemente el tamaño del PCB y por ende las medidas del prototipo.

CAPÍTULO 4: CONCLUSIONES Y RECOMENDACIONES

Se ha cumplido con el objetivo principal de este proyecto de titulación, el cual fue **"DISEÑO DE UN PROTOTIPO ELECTRÓNICO DE BAJO COSTO Y REDUCIDAS DIMENSIONES QUE PERMITA LA ADQUISICIÓN, PROCESAMIENTO Y VISUALIZACIÓN DE SEÑALES CARDÍACAS"**. Para esto, el diseño se fundamentó en el circuito integrado del fabricante Analog Devices, el ADAS1000, un circuito integrado que funciona como primera fase en la adquisición de señales cardiacas. Se diseñó el circuito, los esquemas eléctricos y los demás componentes para el desarrollo de este prototipo. De igual manera, se diseñó una placa PCB con las recomendaciones y especificaciones establecidas por el fabricante. Asimismo, se creó un firmware que permita el control y configuración del ADAS desde un dsPIC30F4013. Se elaboró un case de madera que se puede observar en el Apéndice C, este case fue realizado para facilitar la manipulación y uso del prototipo.

Se ha conseguido elaborar un dispositivo capaz de captar señales cardiacas en tiempo real, las mismas que pueden ser enviadas de manera alámbrica e inalámbrica hacia dispositivos finales para su visualización. El dispositivo es pequeño y de bajo consumo en comparación con los existentes en el mercado. Sin embargo, al observar las señales obtenidas y compararlas con otros dispositivos similares, se concluye que estas señales son de baja calidad, proveen poca información y presentan altos niveles de ruido. La configuración del circuito integrado ADAS1000 resulta de alta complejidad debido a que la documentación existente es deficiente y generalizada, el

fabricante proporciona una hoja de datos y esquemas de conexión, pero recomienda adquirir su placa de evaluación (EVALUATION BOARD ADAS1000)[18], antes que programar desde cero como se lo hizo en esta tesis; sin mencionar que existe poca información en foros de discusión en internet.

Si bien las señales obtenidas no tienen la calidad suficiente para ser estudiadas por el ámbito clínico, ya que deberían superar una serie de normativas y exigencias más detalladas que las nombradas en este proyecto; el prototipo podría de gran utilidad dentro del grupo de investigación de la universidad, para que en futuros trabajos se pueda desarrollar algún filtro o circuito que aclare la señal.

Hoy en día en el mercado podemos encontrar monitores de diferentes dimensiones y precios, los cuales se describen en el estado del arte del Capítulo 1en la página 12, los precios de estos sistemas oscilan entre los $200 y $6000 dólares mientras que nuestro monitor ECG llega hasta los $364,99; claramente estos dispositivos resultan más viables, ya que presentan muchas más prestaciones y ventajas, y lo más importante que los sistemas anteriores están certificados para el uso clínico.

En definitiva, la monitorización de la actividad cardiaca en tiempo real de un paciente es una tarea de gran relevancia en los últimos años. Este proyecto ha intentado ofrecer una alternativa de bajo costo con el prototipo propuesto, no obstante, la complejidad y extensión del proyecto hace que pueda trabajarse en investigaciones futuras, dotando de mayores prestaciones y funciones al prototipo.

Otro de los objetivos alcanzados es la realización de la APP en Android Studio, para ello se tienen dos recomendaciones si se va a programar en JAVA existen varias librerías[19] para realizar gráficas por lo que es difícil seleccionar alguna, en este proyecto se utilizó la librería ACE (A Chart Engine) Por las prestaciones que posee sobre las demás librerías que se han encontrado, una de las mayores ventajas es el

[18] **EVALUATION BOARD ADAS1000:** "La placa de evaluación es ideal para explorar conceptos y adoptar el ADAS1000 en sistemas médicos avanzados". Enlace en donde se puede leer las recomendaciones de Analog Device: https://www.analog.com/media/en/technical-documentation/user-guides/UG-426.pdf

[19] **Librerías:** Enlace en donde se pueden encontrar librerías para graficar en Android Studio: https://www.tecnopedia.net/android-mobile/5-librerias-gratis-para-incluir-graficas-en-tus-aplicaciones-android/

poder realizar gráficas en tiempo real, que es indispensable para este tipo de aplicaciones.

El objetivo 4 “Generar una base de datos con registros de las señales obtenidas”, no se lo completó debido a dos razones, la primera es que las señales obtenidas no son de buena calidad y poseen demasiado ruido, y la finalidad de este objetivo era que la base de datos generada sea de utilidad para estudios futuros. Por lo tanto, no fue conveniente generar dicha base de datos con señales ECG ya que no aportan información contundente. La segunda razón se debe a la emergencia sanitaria, debido a esto no se pudo realizar pruebas a pacientes y solamente se tomaron muestras entre quienes escribimos esta tesis.

Recomendamos el uso de entornos de programación multiplataforma para dispositivos Android e iOS, esto porque en los entornos laborales piden personas con experiencia en dichos entornos de programación, adicionalmente los usuarios de Android e iOS, que son dos de los grandes Sistemas Operativos (SO) se van equilibrando entre sí, y a cada día la brecha de usuarios se va cerrando más.

Para trabajos a futuro en esta misma rama, el uso de nuevas tecnologías como Machine Learning o Deep Learning, pueden ser de utilidad para la detección de patologías cardíacas a tiempo real, esto puede ser un avance significativo y se pondría delante de muchos dispositivos existentes en el mercado.

REFERENCIAS BIBLIOGRÁFICAS

[1] "OMS | ¿Qué son las enfermedades cardiovasculares?," *WHO*, 2015, Accessed: Feb. 02, 2019. [Online]. Available: https://www.who.int/cardiovascular_diseases/about_cvd/es/.

[2] D. Mozaffarian *et al.*, "Heart Disease and Stroke Statistics—2016 Update," *Circulation*, vol. 133, no. 4, pp. e38-360, Jan. 2016, doi: 10.1161/CIR.0000000000000350.

[3] L. Gaztañaga, F. E. Marchlinski, and B. P. Betensky, "Mecanismos de las arritmias cardiacas," *Rev. Española Cardiol.*, vol. 65, no. 2, pp. 174–185, Feb. 2012, doi: 10.1016/J.RECESP.2011.09.018.

[4] C. Coello, "Ecuador recién se interesa por las enfermedades del corazón." https://www.redaccionmedica.ec/secciones/salud-publica/el-objetivo-para-este-milenio-es-controlar-la-enfermedad-cardiovascular-91029.

[5] I. José, S. Pérez, and R. P. Galán, "Anatomía Y Fisiología Del Corazón," pp. 11–25.

[6] A. Sgarlatta, "Sensor inalámbrico de ECG conectado vía Bluetooth a aplicación de análisis automático en el teléfono móvil," p. 138, 2016.

[7] M. J. Noriega, "Fisiologia Humana," 2017. https://ocw.unican.es/mod/page/view.php?id=534.

[8] K. Barret, S. Barman, S. Boitano, and H. Brooks, *Ganong. Fisiología médica*, 24th ed. 2013.

[9] L. Rubio, "Curso de Electrocardiografía," 2010. http://www.lrubio.es/ECG/tema1/tema1-2.html.

[10] J. Ramirez, "Potencial de Acción Cardíaco," no. 7, 2009, [Online]. Available: http://paginas.facmed.unam.mx/deptos/fis/wp-content/uploads/2018/11/UT-II-Guia10.pdf.

[11] D. Dubin, *Dubin: Interpretación de ECG*, 1st ed. 2011.

[12] My EKG, “My EKG la web del electrocardiograma,” 2013. https://www.my-ekg.com/index.html (accessed Jul. 15, 2019).

[13] J. R. Hampton and D. Adlam, *ECG en la práctica*, 6th ed. Elsevier, 2014.

[14] J. R. Hampton, *ECG fácil*, 8th ed. 2013.

[15] A. L. Goldberger, Z. D. Goldberger, and A. Shvilkin, *Clinical Electrocardiography A Simplified Approach*, 9th ed. 2018.

[16] B. Surawicz, T. K. Knilans, L. E. Gering, and M. E. Tavel, *Chou’s Electrocardiography in clinical practice*, 6th ed. 2008.

[17] J. E. Hall, *Tratado de Fisiología Médica*, 3rd ed. 2016.

[18] A. Houghton and D. Gray, *Making Sense of the ECG a hands on guide*, 4th ed. 2014.

[19] J. M. Ferrero Corral, J. Saiz Rodriguez, and A. Arnau Vives, *Bioelectrónica: Señales Bioeléctricas*. 1994.

[20] A. E. García, *Análisis y procesamiento de la señal auscultada del corazón para el diagnóstico presuntivo de soplos cardíacos y arritmia cardíaca*. 2017.

[21] G. F. Salazar, *Electrocardiografía*. 2005.

[22] Qardio Inc., “QARDIO,” 2015. https://www.getqardio.com/.

[23] AliveCor Inc, “AliveCor,” [Online]. Available: https://www.alivecor.com/.

[24] Henan Corp, “HENAN MEDICAL,” [Online]. Available: https://shop-henanmedical.com/products/pc-based-holter-ecg-system-cc-holter.html.

[25] Analog Devices, *Low Power, Five Electrode Electrocardiogram (ECG) Analog Front End. Datasheet ADAS100 - ADAS1001 - ADAS1002*. 2013.

[26] Microchip, *Data Sheet dsPIC30F4013*, vol. 31402, no. September 2004. 2012.

APÉNDICES

APÉNDICE A: ESQUEMA DEL PCB DISEÑADO EN ALTIUM DESIGNER.

El esquema utilizado para este proyecto se muestra dividido en 3 partes, la primera es la fuente de alimentación, la segunda, la configuración del ADAS1000 y la tercera del Microcontrolador dsPIC30F4013

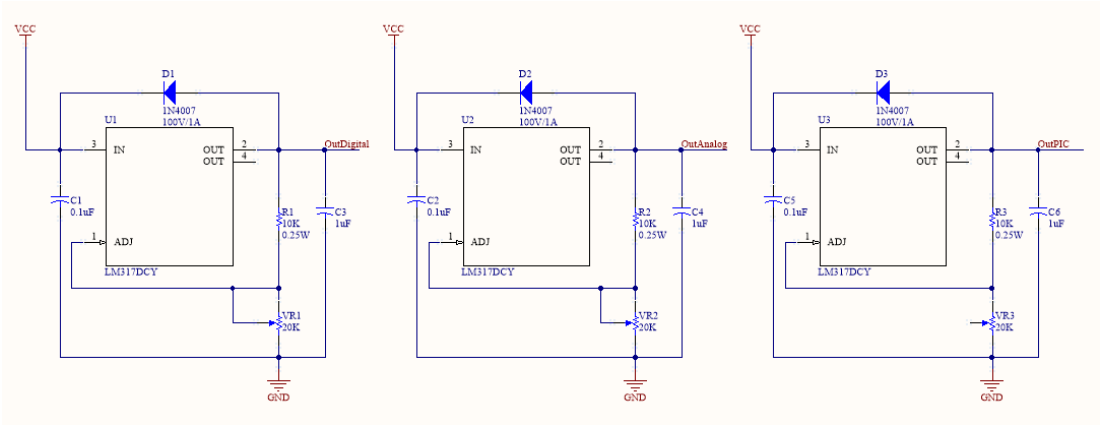


Figura 40 Esquema de los reguladores de voltaje, parte 1.

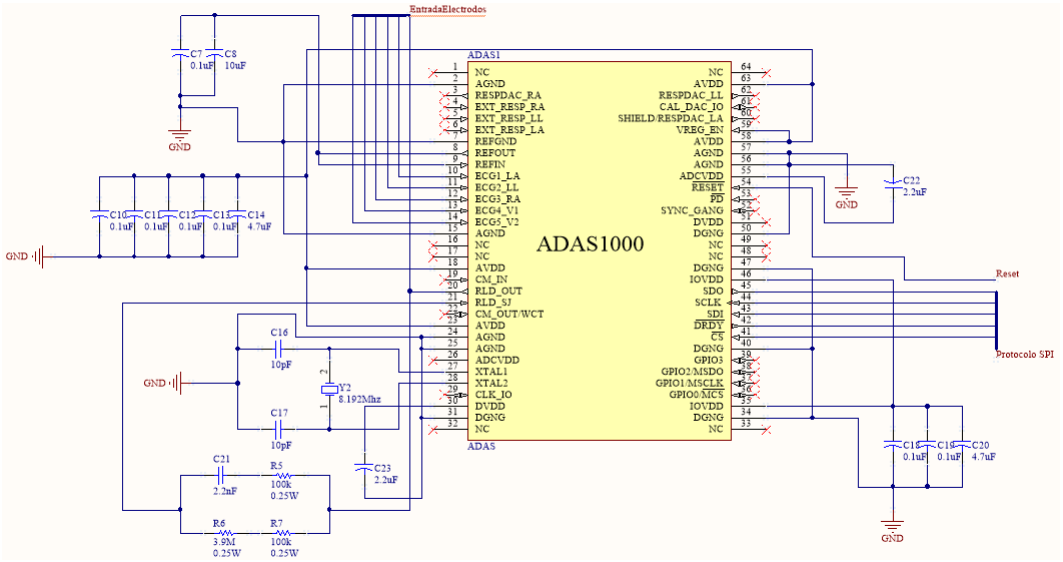


Figura 41 Esquema de conexión para ADAS1000, parte 2.

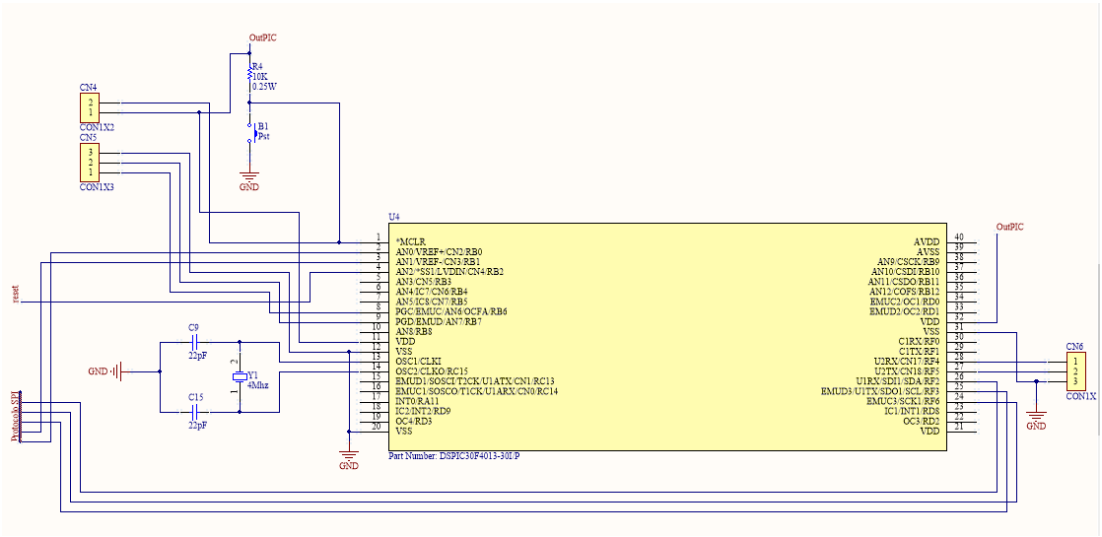


Figura 42 Esquema de conexión dsPIC, parte 3.

APÉNDICE B: DISEÑO DEL CASE

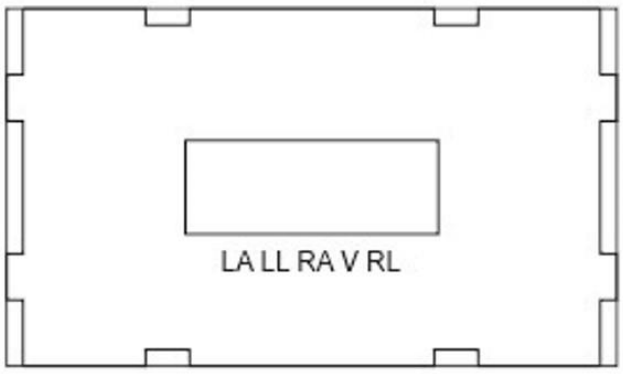


Figura 43 Vista Frontal

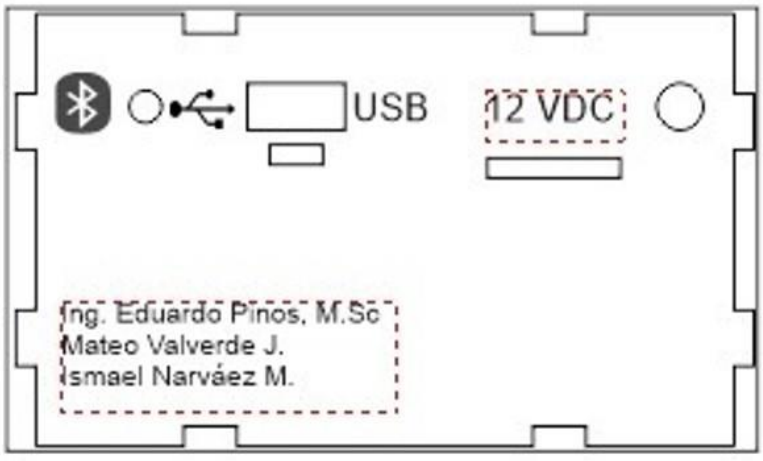


Figura 44 Vista Posterior

Figura 45 Vista Superior

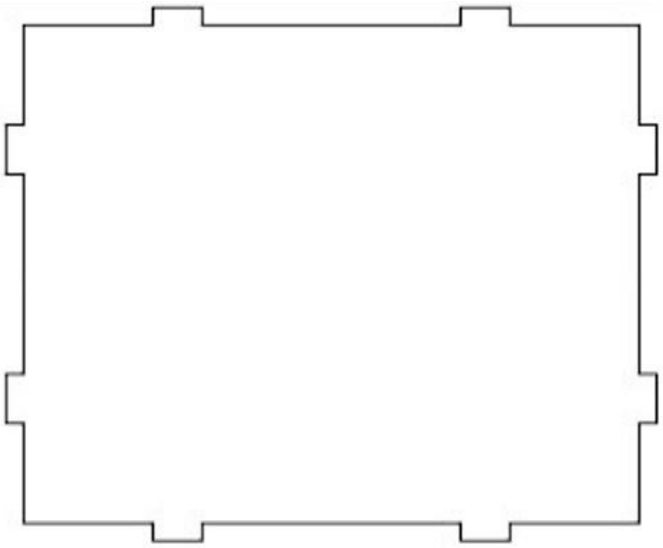


Figura 46 Vista Inferior

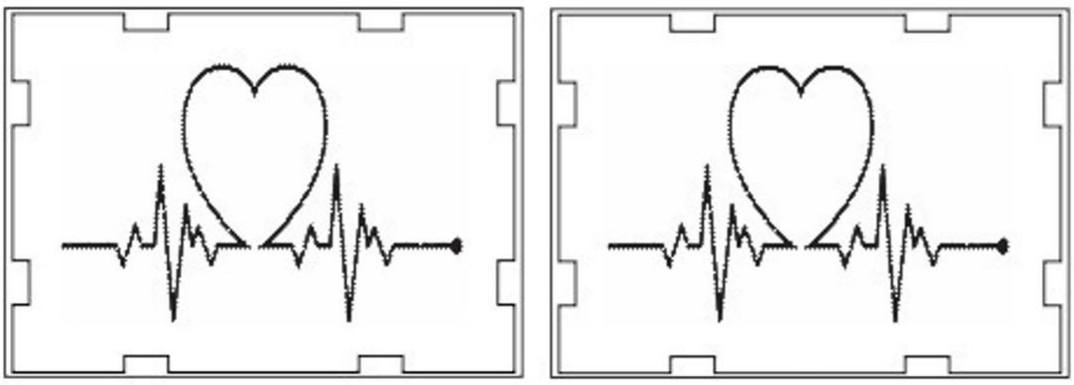


Figura 47 Vista lateral izquierda y lateral derecha

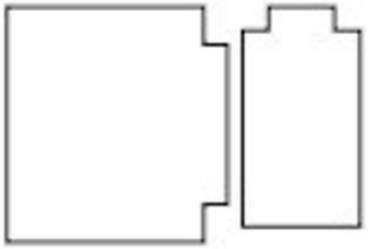


Figura 48 Soportes internos

APÉNDICE C: FOTOGRAFÍAS DEL PROTOTIPO

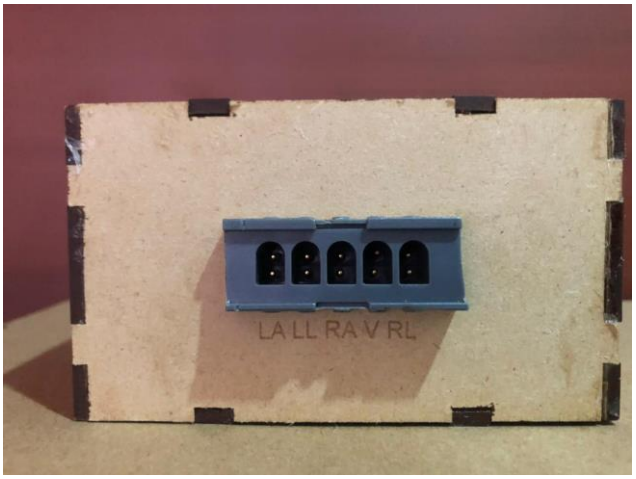


Figura 49 Vista frontal del case montado

Figura 50 Vista posterior del case montado

Figura 51 Vista del case montado

APÉNDICE D: MANUAL DE USUARIO

1. Requerimientos

Para el correcto funcionamiento de este Monitor ECG es necesario contar tanto con software y hardware, que se presentan a continuación:

1.1 Hardware

- Electrodos desechables para ECG (5 en cada uso).
- Cables para electrodos.
- Cable USB – USB.
- Fuente de 12V, 2A (valores máximos que puede tolerar el prototipo).
- Computadora con SO Windows 8/8.1/10.
- Smartphone con SO Android 5.1 o superior.

1.2 Software

- El software que necesitará en el computador es “ECG IDE”.
- La APP para el dispositivo Android es “Monitor ECG”.

1.3 Instalación / funcionamiento

Este manual se dividió en tres partes, la primera sobre el manejo de la APP para Windows, la segunda parte se explica el funcionamiento de la APP para Android y finalmente tenemos como manipular el prototipo físico para su correcto funcionamiento.

1.3.1 Aplicación para Windows

Esta aplicación es un ejecutable, por lo que no requiere ninguna instalación, solamente se debe tener dicho ejecutable en cualquier carpeta del computador véase Figura 52.

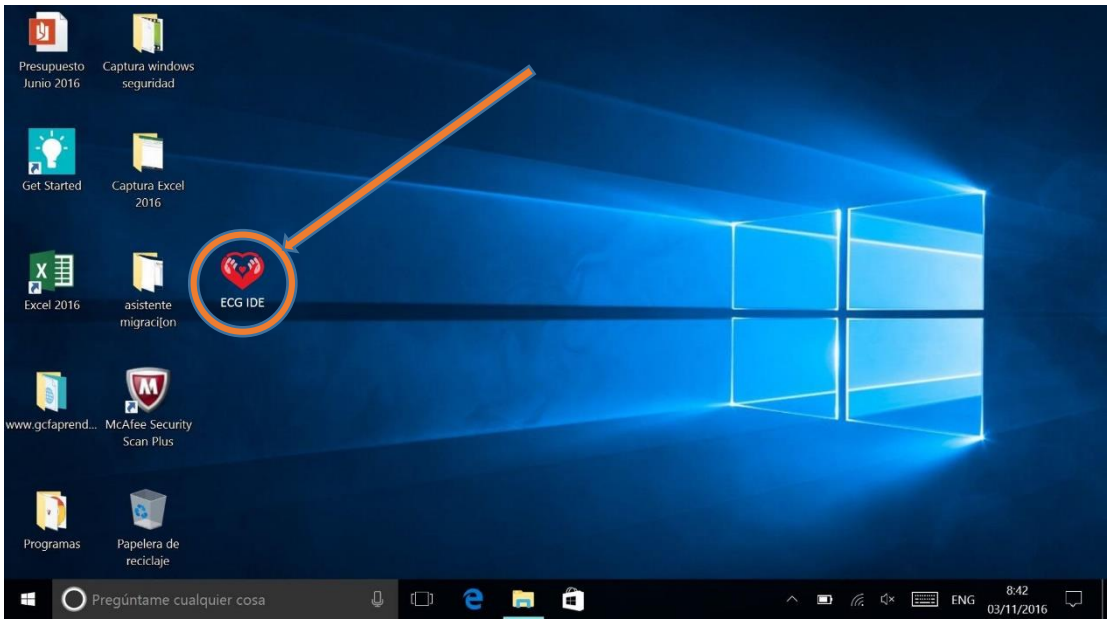


Figura 52 ECG IDE, ejecutable para el monitor ECG en PC

Al abrir dicho ejecutable, prototipo debe conectarse con el ordenador, para ello, se debe seleccionar el puerto al que está conectado.

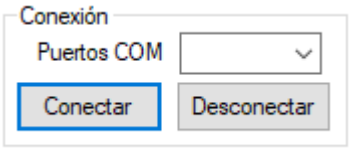


Figura 53 Seleccionar el puerto al que está conectado el prototipo

Al presionar en Conectar comenzará la transmisión de los datos y posteriormente se empezará a graficar la señal cardiaca.

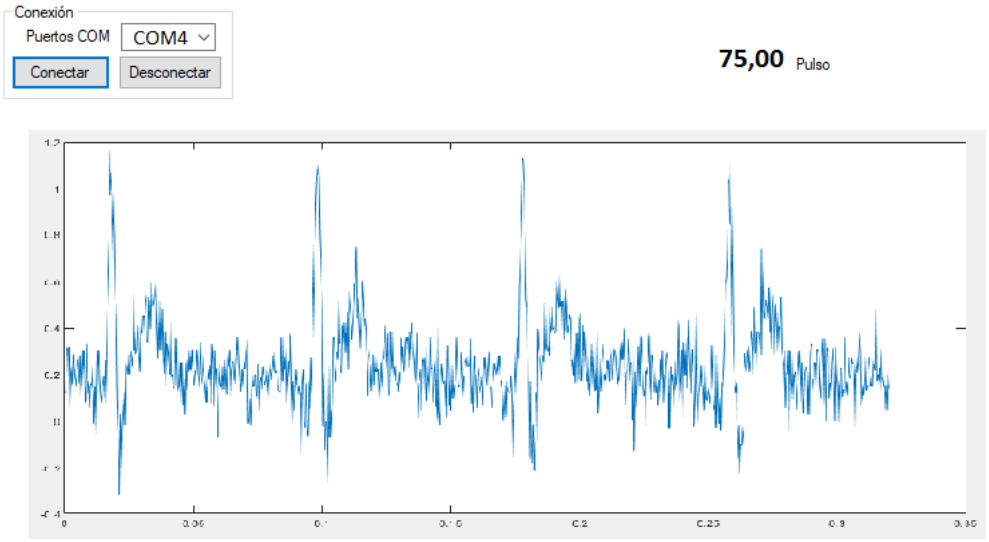


Figura 54 Gráfica de la señal cardíaca

1.3.2 Aplicación para Android

Para la Aplicación en Android vamos a instalar el APK en el smartphone, una vez instalado tendremos en la pantalla inicial algo similar a la Figura 55.

La Aplicación tiene el nombre “Monitor ECG”, para poder usarla necesitamos un dispositivo Android 5.0 o superior, adicionalmente debemos asegurarnos que el teléfono celular esté funcionando correctamente incluyendo la tecnología Bluetooth.

Teniendo en cuenta estas recomendaciones procedemos a presionar el ícono y esperamos a la siguiente pantalla.

Figura 55 Icono de la APP "Monitor ECG"

Si al presionar el ícono no se ha encendido Bluetooth, esta es la imagen que aparecerá, por lo que debemos presionar el botón “Permitir” para otorgar los permisos necesarios y que la APP pueda funcionar de manera correcta, si presionamos el botón “Rechazar” la APP procederá a cerrarse.

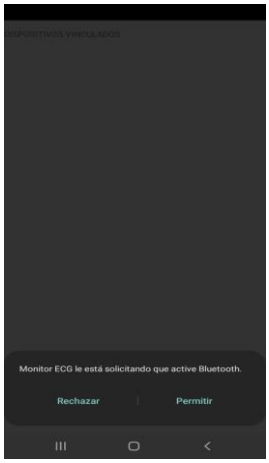


Figura 56 Permisos para Bluetooth

La siguiente pantalla muestra los dispositivos vinculados con nuestro dispositivo, por lo que si no hemos enlazado anteriormente el prototipo con el teléfono móvil debemos conectarlo y cuando pida una contraseña ingresamos “1234”, una vez realizado esto la pantalla se actualizará y aparecerá el nombre “MonitorECG”, procedemos a presionar y el dispositivo se conectará y pasará a la pantalla que se muestra en la Figura 57.

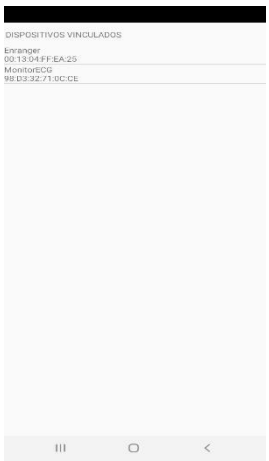


Figura 57 Pantalla de selección de dispositivos BT

Ahora procedemos a tener la última pantalla (véase Figura 58) de la aplicación en donde se va a mostrar la gráfica del ECG, aquí tenemos 3 botones que son fáciles de intuir, con el botón encender inicia el proceso de transmisión de nuestra señal cardíaca, el botón Apagar permite dejar de transmitir la señal, y el botón desconectar permite que dejemos de estar enlazados vía Bluetooth con el prototipo.

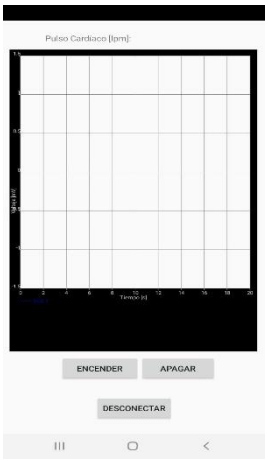


Figura 58 Pantalla principal de la APP

A continuación, se muestra la gráfica de una señal que hemos obtenido en una de las pruebas realizadas.

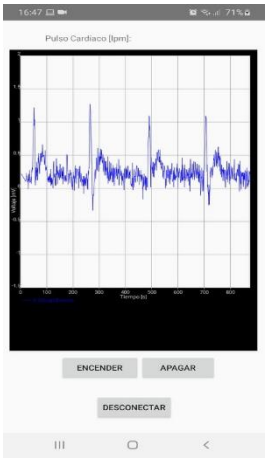


Figura 59 Señal Obtenida

1.3.3 Conexión y uso del prototipo

Para poder utilizar el dispositivo en primer lugar debemos presionar el botón de encendido que se encuentra en la parte superior.

Figura 60 Vista superior del prototipo

En la parte posterior tenemos un selector en donde podemos configurar si deseamos visualizar la señal en el teléfono móvil o en una computadora, además tenemos el ingreso de voltaje.

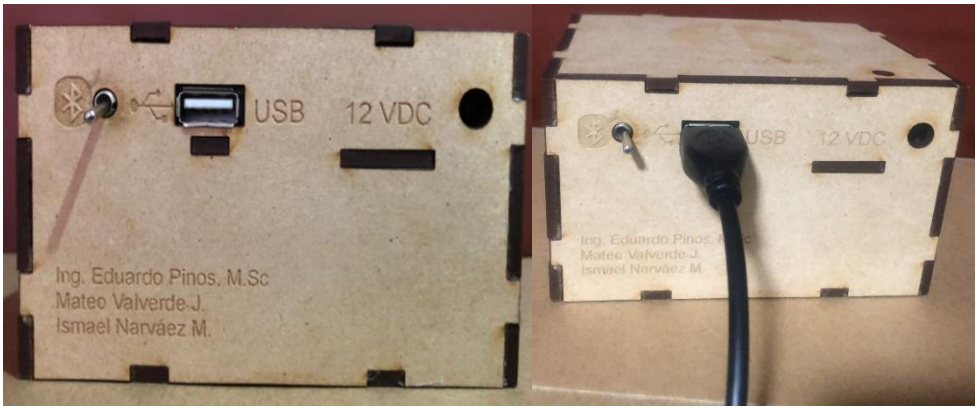


Figura 61 Vista posterior en donde se encuentra el selector y el ingreso de voltaje.

En la Parte Frontal tenemos el ingreso de los electrodos LA, LL, RA, V y RL, debidamente señalizados.

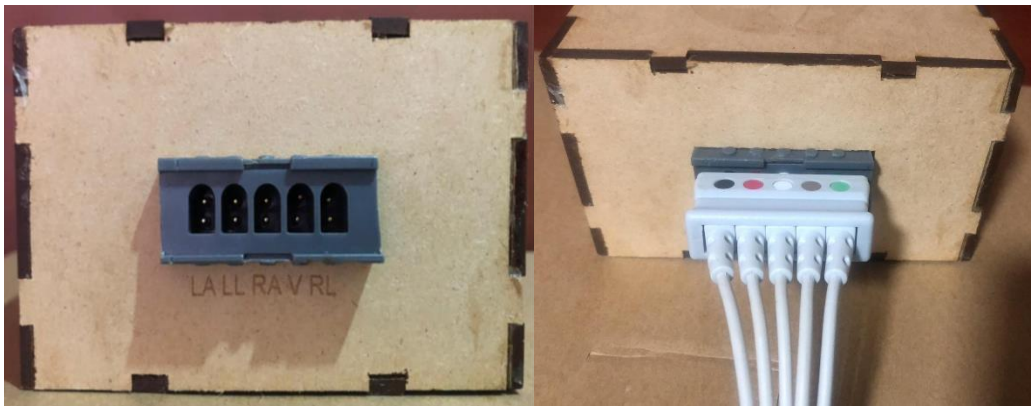


Figura 62 Ingreso de los electrodos

Finalmente se presenta la manera en la cual se ubican los electrodos en la caja torácica Figura 63, para ello tendremos en cuenta los colores y las etiquetas usadas en nuestro prototipo.

- **Color Negro:** Etiqueta LA.
- **Color Rojo:** Etiqueta LL.
- **Color Blanco:** Etiqueta RA
- **Color Café:** Etiqueta V
- **Color Verde:** Etiqueta RL

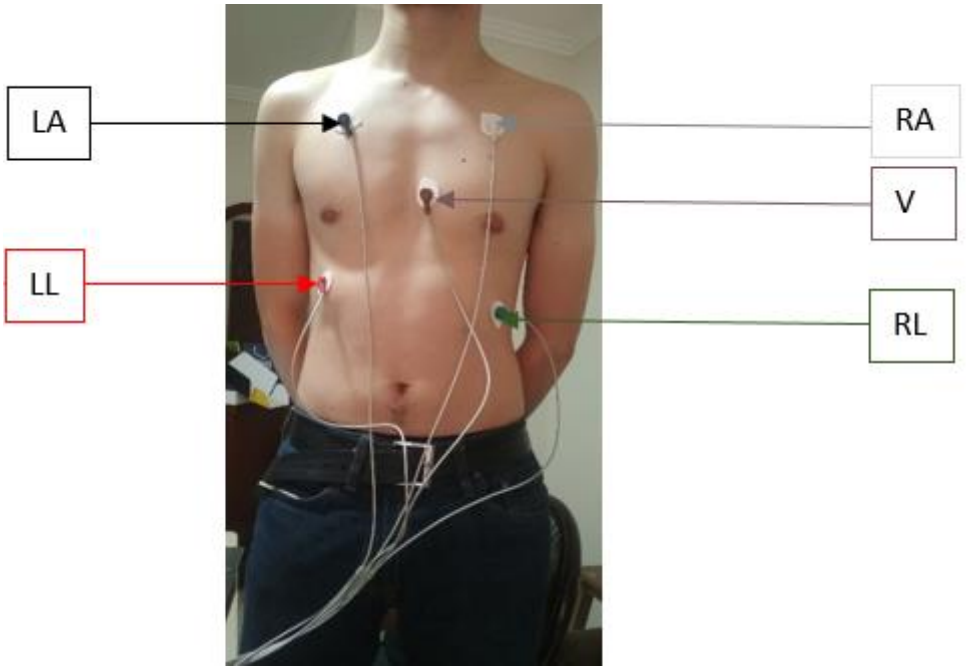


Figura 63 Disposición de electrodos en la caja torácica

1.4 Consideraciones / Recomendaciones

- Recordar que la fuente de alimentación tiene que ser 12V 2A, si sobrepasa este voltaje se puede llegar a quemar la placa principal.
- Para cada uso es necesario usar 5 electrodos desechables.

APÉNDICE E: CÓDIGO DE PROGRAMACIÓN EN MPLAB PARA DSPIC

```
#include "xc.h"
#include "config.h"
#include "reloj.h"
#include <libpic30.h>
//#include "spi.h"

//#define _XTAL_FREQ 4000000
#define cs1 LATBbits.LATB0
#define ddry LATBbits.LATB1
#define reset LATBbits.LATB2

unsigned char spi_write(unsigned char spi_dat);
//void write_adas(unsigned char spi_dat);
```

```
void InitUART2(void);
void config_adas_testtone(void);
void config_adas_ecg(void);
void WriteUART2(unsigned int data);
void send_data(void);
void read_adas(void);
void WriteUART2dec2string(unsigned int data);
void WriteStringUART2(const char * s);

unsigned char head [4]={0,0,0,0};
unsigned char lead1 [4]={0,0,0,0};
unsigned char lead2 [4]={0,0,0,0};
unsigned char lead3 [4]={0,0,0,0};
unsigned char lead4 [4]={0,0,0,0};
unsigned char lead5 [4]={0,0,0,0};
unsigned char ritmo [4]={0,0,0,0};
unsigned char resp [4]={0,0,0,0};
unsigned char leadoff [4]={0,0,0,0};

char *p;
unsigned char buf[30];
unsigned char da;

void main(void) {
   TRISFbits.TRISF2=1; //SDI
   TRISFbits.TRISF3=0; //SDO
   TRISFbits.TRISF6=0; //SCK
   TRISBbits.TRISB0=0; //CS/SS
   TRISDbits.TRISD0=0;
   TRISBbits.TRISB1=1;
   TRISBbits.TRISB2=0; //DDRY//DDRY

   LATBbits.LATB0=1;

   //Configurar el SPI
   SPI1STAT=0;
   SPI1STATbits.SPIEN=1;
   SPI1CON=0;
   SPI1CONbits.SMP=1; //muestrea a la mitad del reloj
   SPI1CONbits.CKE=0; //recibe o transmite cunado pasa de reposo a activo
```

```
SPI1CONbits.CKP=0; //el estado bajo es reposo
SPI1CONbits.MODE16=0;
SPI1CONbits.MSTEN=1;
SPI1CONbits.SPRE=0b000; //x8 para 500KHz
SPI1CONbits.PPRE=0b11; //x1

InitUART2();
__delay_ms(1000);
config_adas_testtone();
__delay_ms(100);
/*config_adas_ecg();
__delay_ms(1000);*/
while(1){
/*cs1=0;
head[1] = spi_write(0x00);
head[2] = spi_write(0x00);
head[3] = spi_write(0x00);
head[4] = spi_write(0x00);
cs1=1
__delay_ms(2);

cs1=0;
lead1[1] = spi_write(0x00);
lead1[2] = spi_write(0x00);
lead1[3] = spi_write(0x00);
lead1[4] = spi_write(0x00);
cs1=1;

__delay_ms(2);

cs1=0;
lead2[1] = spi_write(0x00);
lead2[2] = spi_write(0x00);
lead2[3] = spi_write(0x00);
lead2[4] = spi_write(0x00);
cs1=1;

__delay_ms(2);

cs1=0;
```

```
lead3[1] = spi_write(0x00);
lead3[2] = spi_write(0x00);
lead3[3] = spi_write(0x00);
lead3[4] = spi_write(0x00);
cs1=1;

__delay_ms(2);

cs1=0;
lead4[1] = spi_write(0x00);
lead4[2] = spi_write(0x00);
lead4[3] = spi_write(0x00);
lead4[4] = spi_write(0x00);
cs1=1;

__delay_ms(2);

cs1=0;
lead5[1] = spi_write(0x00);
lead5[2] = spi_write(0x00);
lead5[3] = spi_write(0x00);
lead5[4] = spi_write(0x00);
cs1=1;

__delay_ms(2);

cs1=0;
ritmo[1] = spi_write(0x00);
ritmo[2] = spi_write(0x00);
ritmo[3] = spi_write(0x00);
ritmo[4] = spi_write(0x00);
cs1=1;

__delay_ms(5);


sprintf(buf,"A%i;%i;%i;%i\n\r",lead1[1],lead1[2],lead1[3],lead1[4]);
WriteStringUART2(buf);
sprintf(buf,"B%i;%i;%i;%i\n\r",lead2[1],lead2[2],lead2[3],lead2[4]);
WriteStringUART2(buf);
```

```
    sprintf(buf,"C%i;%i;%i;%i\n\r",lead3[1],lead3[2],lead3[3],lead3[4]);
    WriteStringUART2(buf);
    sprintf(buf,"D%i;%i;%i;%i\n\r",lead4[1],lead4[2],lead4[3],lead4[4]);
    WriteStringUART2(buf);
    sprintf(buf,"E%i;%i;%i;%i\n\r",lead5[1],lead5[2],lead5[3],lead5[4]);
    WriteStringUART2(buf);
    sprintf(buf,"E%i;%i;%i;%i\n\r",ritmo[1],ritmo[2],ritmo[3],ritmo[4]);
    WriteStringUART2(buf);

        __delay_ms(100);*/

        while(ddry==1);

        cs1=0;
    lead1[1] = spi_write(0x00);
    lead1[2] = spi_write(0x00);
    lead1[3] = spi_write(0x00);
    lead1[4] = spi_write(0x00);
    cs1=1;

    if(lead1[1]==0x00){
        sprintf(buf,"A%i\n\r",0x01);
        WriteStringUART2(buf);
    }
    else{
    sprintf(buf,"A%i;%i;%i;%i\n\r",lead1[1],lead1[2],lead1[3],lead1[4]);
    WriteStringUART2(buf);
    }
    }

    return;
}

unsigned char spi_write(unsigned char spi_dat){

    SPI1BUF=spi_dat;
    while(SPI1STATbits.SPIRBF==0);
    return(SPI1BUF);
    //WriteUART2(SPI1BUF);
    //WriteUART2(59);
```

```
}
void config_adas_testtone(void){

   cs1=0;
   spi_write(0x85);
   spi_write(0x00);
   spi_write(0x00);
   spi_write(0x0B);

   cs1=1; __delay_us(1000);

   cs1=0;
   spi_write(0x88);
   spi_write(0xF8);
   spi_write(0x00);
   spi_write(0x0D);

   cs1=1; __delay_us(1000);

   cs1=0;
   spi_write(0x8B);
   spi_write(0x00);
   spi_write(0x00);
   spi_write(0x08);

   cs1=1; __delay_us(1000);

   cs1=0;
   spi_write(0x8A);
   spi_write(0x07);
   spi_write(0x96);
   spi_write(0x10);

   cs1=1; __delay_us(1000);

   cs1=0;
   spi_write(0x81);
   spi_write(0xF8);
   spi_write(0x00);
   spi_write(0xAE);
```

```
    cs1=1; __delay_us(1000);

    cs1=0;
    spi_write(0x40);
    spi_write(0x00);
    spi_write(0x00);
    spi_write(0x00);

    cs1=1; __delay_us(1000);
}

void config_adas_ecg(void){
    cs1=0;
    spi_write(0x0B);
    spi_write(0x00);
    spi_write(0xE0);
    spi_write(0x85);
    cs1=1; __delay_ms(50);

    cs1=0;
    spi_write(0x00);
    spi_write(0xBE);
    spi_write(0x07);
    spi_write(0x8A);
    cs1=1; __delay_ms(50);

    cs1=0;
    spi_write(0xAE);
    spi_write(0x06);
    spi_write(0xF8);
    spi_write(0x81);
    cs1=1; __delay_ms(50);

    cs1=0;
    spi_write(0x00);
    spi_write(0x00);
    spi_write(0x00);
    spi_write(0x40);
    cs1=1; __delay_ms(50);
```

```
}
//UART2

void InitUART2(void){
   U2BRG = 25; // baud rate 9600
   //U2STA &= 0xfffc;
   //IEC1bits.U2RXIE = 1; // enable RX2 interrupt
   U2MODEbits.UARTEN = 1; // UARTEN: UART Enable bit
   U2STAbits.UTXEN = 1; // transmition ON
}
void WriteUART2 (unsigned int data){
   U2TXREG = data;
   while(!U2STAbits.TRMT){}
}

void WriteStringUART2(const char * s){
   while(*s)
         WriteUART2(*s++);
}
```

APÉNDICE F: CÓDIGO DE PROGRAMACIÓN EN VISUAL STUDIO

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Windows.Forms.DataVisualization.Charting;

namespace GraficaTemp{
```

```
public partial class Temperatura : Form{
    System.IO.Ports.SerialPort puerto;
    String[] listado_puerto = System.IO.Ports.SerialPort.GetPortNames();
    string datos_puerto;
    double serie1 = 0.0, serie2 = 0.0, serie3 = 0.0, serie4 = 0.0;
    int lead1 = 0;
    double tiempo = 0.0;
    int res = 0;
    int x = 0;
    int y = 0;
    string Xg;
    string Yg;
    string LEAD_1 = "";
    System.Drawing.Rectangle r1 = new System.Drawing.Rectangle();
    public Temperatura(){
      InitializeComponent();
      this.comboBox1.DataSource = listado_puerto; // listado de puertos
    }
    public void serial(){
      try{
        this.puerto = new System.IO.Ports.SerialPort("" + comboBox1.SelectedItem, 9600, System.IO.Ports.Parity.None, 8, System.IO.Ports.StopBits.One);
        this.puerto.DataReceived += new System.IO.Ports.SerialDataReceivedEventHandler(recepcion);
      }
      catch (Exception){
        MessageBox.Show("Verifique:" + System.Environment.NewLine + "- Voltage" + System.Environment.NewLine + "- Conexion del puerto", "Error de puerto COM", MessageBoxButtons.OK, MessageBoxIcon.Error);
      }
    }
    public void recepcion(object sender, System.IO.Ports.SerialDataReceivedEventArgs e){
      try{
        Thread.Sleep(10);
```

```
        datos_puerto = this.puerto.ReadLine();

        datos_puerto = datos_puerto.Remove(0, 1);
        Console.WriteLine(datos_puerto);
        if (datos_puerto.StartsWith("A")){ this.Invoke(new EventHandler(actualizar)); }
        else {
            //this.Invoke(new EventHandler(actualizar));
        }
    }
    catch (Exception) { }
}

private void timer1_Tick(object sender, EventArgs e){
    tiempo++;
    res++;
    chart2.Series[0].Points.AddXY((tiempo / 2), serie1);
    textBox1.Text = "" + serie2;
    if (res == 200){
        chart2.Series[0].Points.Clear();
        res = 0;
    }
}

private void button1_Click(object sender, EventArgs e){
    try{
      serial();
        puerto.Open();
        timer1.Start();
    }
    catch (Exception){ }
}
private void button5_Click(object sender, EventArgs e){
    puerto.Close();
```

```
        timer1.Stop();
    }
    private void chart2_PostPaint(object sender, System.Windows.Forms.DataVisualization.Charting.ChartPaintEventArgs e){
        System.Drawing.Font drawFont = new System.Drawing.Font("Verdana", 8);
        System.Drawing.SolidBrush drawBrush = new System.Drawing.SolidBrush(System.Drawing.Color.Red);
        System.Drawing.StringFormat drawFormat = new System.Drawing.StringFormat();
        if (!(string.IsNullOrEmpty(Xg) && string.IsNullOrEmpty(Yg))){
            e.ChartGraphics.Graphics.DrawString("ELong=" + Xg + "\n" + "Fuerza=" + Yg, drawFont, drawBrush, x + 5, y - 2);
            e.ChartGraphics.Graphics.DrawRectangle(new Pen(Color.Red, 3), r1);
        }
    }
    private void chart2_MouseMove(object sender, MouseEventArgs e){
        var pos1 = e.Location;
        if (prevPosition.HasValue && pos1 == prevPosition.Value)
            return;
        tooltip.RemoveAll();
        prevPosition = pos1;
        var results1 = chart2.HitTest(pos1.X, pos1.Y, false,
                                      ChartElementType.DataPoint);
        foreach (var result1 in results1){
            if (result1.ChartElementType == ChartElementType.DataPoint){
                var prop1 = result1.Object as DataPoint;
                if (prop1 != null){
                    var pointXPixel = result1.ChartArea.AxisX.ValueToPixelPosition(prop1.XValue);
                    var pointYPixel = result1.ChartArea.AxisY.ValueToPixelPosition(prop1.YValues[0]);

                    // check if the cursor is really close to the point (2 pixels around)

                    tooltip.Show("Tiempo=" + prop1.XValue + ", Fuerza=" + prop1.YValues[0], this.chart2,
                                 pos1.X, pos1.Y - 15);
```

```
            }
          }
        }
    }
    System.Drawing.Point? prevPosition = null;
    ToolTip tooltip = new ToolTip();

    private void chart2_MouseDown(object sender, MouseEventArgs e){
        chart2.Invalidate();
        r1.X = x;
        r1.Y = y;
        r1.Width = 3;
        r1.Height = 3;
    }

    public void actualizar(object s, EventArgs e){
        if (datos_puerto.Substring(0, 1) == "A"){
            datos_puerto = datos_puerto.Remove(0, 1); }
        string[] arreglo = datos_puerto.Split(';');
        if ((int.Parse(arreglo[0]) > 180 && int.Parse(arreglo[1]) > 180 && int.Parse(arreglo[2]) > 180
&& int.Parse(arreglo[3]) > 180) || (arreglo[0] == "0" && arreglo[1] == "0" && arreglo[2] == "0" &&
arreglo[3] == "0")){
            serie1 = 0.0;
        }
        else{
            serie1 = BinToDec(DecToBin(int.Parse(arreglo[0])) + DecToBin(int.Parse(arreglo[1])) +
DecToBin(int.Parse(arreglo[2])) + DecToBin(int.Parse(arreglo[3])));
        }
        //serie1 = Convert.ToDouble(arreglo[0].Replace(".", ","));
        //serie1 = serie1 * 1048575 / (Math.Pow(2, 32) - 1);
        serie1 = serie1 * 0.002 / (Math.Pow(2, 32) - 1);
        Console.WriteLine(serie1);
        serie2 = Convert.ToDouble(arreglo[1].Replace(".", ","));
    }
```

```
        public string DecToBin(int a){
            string[] binario = new string[8];
            int divisiones = 0;
            int numero = a;
            int indice = 0;
            do{
                if (numero % 2 == 0){
                    binario[indice] = "0";
                }
                else {
                    binario[indice] = "1";
                }
                divisiones++;
                numero = numero / 2;
                indice++;

            } while (numero > 0);
            while (indice < 8){
                binario[indice] = "0";
                indice++;
            }
            Array.Reverse(binario);
            string cadena = binario[0] + binario[1] + binario[2] + binario[3] + binario[4] + binario[5] + binario[6] + binario[7];
            return cadena;
        }
        public double BinToDec(string c) {
            string ca;
            string[] num = new string[32];
            double[] num2 = new double[32];
            double res = 0.0;
            int j = 0;
            for (int i = 31; i >= 0; i--){
```

```
            num[31-i] =c.Substring(31-i, 1);// *Math.Pow(2,i);
            num2[31-i]=Convert.ToDouble(num[31-i])*Math.Pow(2,Convert.ToDouble(i));
            res = res + num2[31-i];
           }
         return res;
         }
   }
}
```

APÉNDICE G: CÓDIGO DE PROGRAMACIÓN EN ANDROID STUDIO

```
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import android.content.Context;
import android.graphics.Color;
import android.graphics.Paint;
import android.os.AsyncTask;
import android.widget.LinearLayout;
```

```
import org.achartengine.ChartFactory;
import org.achartengine.GraphicalView;
import org.achartengine.model.TimeSeries;
import org.achartengine.model.XYMultipleSeriesDataset;
import org.achartengine.model.XYSeries;
import org.achartengine.renderer.XYMultipleSeriesRenderer;
import org.achartengine.renderer.XYSeriesRenderer;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.UUID;

public class MainActivity extends AppCompatActivity {

    //1)
    Button IdEncender, IdApagar,IdDesconectar;
    TextView IdBufferIn;
    //-----------------------------------------
    Handler bluetoothIn;
    final int handlerState = 0;
    private BluetoothAdapter btAdapter = null;
    private BluetoothSocket btSocket = null;
    private StringBuilder DataStringIN = new StringBuilder();
    private ConnectedThread MyConexionBT;
    // Identificador unico de servicio - SPP UUID
    private static final UUID BTMODULEUUID = UUID.fromString("00001101-0000-1000-8000-
00805F9B34FB");
    // String para la direccion MAC
    private static String address = null;

    //----------------------  PARA GRAFICA ------------------------//
    private XYMultipleSeriesDataset mDataset = new XYMultipleSeriesDataset();
    private XYMultipleSeriesRenderer mRenderer = new XYMultipleSeriesRenderer();
    private XYSeries mCurrentSeries;
    private XYSeriesRenderer mCurrentRenderer;
    private String mDateFormat;
    private GraphicalView mChartView;

    static double x = 0;
```

```
static double y = 0;
private int MaxSize = 200;

private double xx;
private double yy;

protected Update mUpdateTask;

@Override
protected void onRestoreInstanceState(Bundle savedState) {
   super.onRestoreInstanceState(savedState);
   mDataset = (XYMultipleSeriesDataset) savedState
         .getSerializable("dataset");
   mRenderer = (XYMultipleSeriesRenderer) savedState
         .getSerializable("renderer");
   mCurrentSeries = (XYSeries) savedState
         .getSerializable("current_series");
   mCurrentRenderer = (XYSeriesRenderer) savedState
         .getSerializable("current_renderer");
   mDateFormat = savedState.getString("date_format");
}

@Override
protected void onSaveInstanceState(Bundle outState) {
   super.onSaveInstanceState(outState);
   outState.putSerializable("dataset", mDataset);
   outState.putSerializable("renderer", mRenderer);
   outState.putSerializable("current_series", mCurrentSeries);
   outState.putSerializable("current_renderer", mCurrentRenderer);
   outState.putString("date_format", mDateFormat);
}



@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   //Orientación de la pantalla HORIZONTAL
   setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
```

```
//PANTALLA COMPLETA
this.getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
      WindowManager.LayoutParams.FLAG_FULLSCREEN);

//Enlaza los controles con sus respectivas vistas
IdEncender = (Button) findViewById(R.id.IdEncender);
IdApagar = (Button) findViewById(R.id.IdApagar);
IdDesconectar = (Button) findViewById(R.id.IdDesconectar);
IdBufferIn = (TextView) findViewById(R.id.IdBufferIn);

mRenderer.setAxisTitleTextSize(16);
mRenderer.setChartTitleTextSize(20);
mRenderer.setLabelsTextSize(15);
mRenderer.setLegendTextSize(15);

mRenderer.setYLabelsAlign(Paint.Align.RIGHT);
mRenderer.setXAxisMax(20);
mRenderer.setXAxisMin(0);
mRenderer.setYAxisMax(1.5);
mRenderer.setYAxisMin(-1.5);

mRenderer.setZoomEnabled(false, false);
mRenderer.setZoomButtonsVisible(false);
mRenderer.setPanEnabled(false);

mRenderer.setXTitle("Tiempo [s]");
mRenderer.setYTitle("Voltaje [mV]");

mRenderer.setShowGrid(true);
mRenderer.setGridColor(Color.GRAY);
mRenderer.setXLabels(11); // numero de lineas en el eje X
mRenderer.setYLabels(7);  // numero de lineas en el eje Y

String seriesTitle = "ECG " + (mDataset.getSeriesCount() + 1);
XYSeries series = new XYSeries(seriesTitle);

mDataset.addSeries(series);
mCurrentSeries = series;

XYSeriesRenderer renderer = new XYSeriesRenderer();
```

```
mRenderer.addSeriesRenderer(renderer);
mCurrentRenderer = renderer;

mUpdateTask = new Update();
mUpdateTask.execute(this);

// interruopción para interactuar con  los datos de ingreso
bluetoothIn = new Handler() {
   public void handleMessage(android.os.Message msg) {
      if (msg.what == handlerState) {
         String readMessage = (String) msg.obj;
         DataStringIN.append(readMessage);

         int endOfLineIndex = DataStringIN.indexOf("#");

         if (endOfLineIndex > 0) {
            String dataInPrint = DataStringIN.substring(0, endOfLineIndex);
            IdBufferIn.setText("Dato: " + dataInPrint);//<-<- PARTE A MODIFICAR >->->

            y = Double.parseDouble(dataInPrint);

            DataStringIN.delete(0, DataStringIN.length());
         }
      }
   }
};

btAdapter = BluetoothAdapter.getDefaultAdapter(); // get Bluetooth adapter
VerificarEstadoBT();

// Configuracion onClick listeners para los botones
// para indicar que se realizara cuando se detecte
// el evento de Click
IdEncender.setOnClickListener(new View.OnClickListener() {
   public void onClick(View v)
   {
      MyConexionBT.write("1");
   }
});
```

```
    IdApagar.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            MyConexionBT.write("0");
        }
    });

    IdDesconectar.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            if (btSocket!=null)
            {
                try {btSocket.close();}
                catch (IOException e)
                { Toast.makeText(getBaseContext(), "Error", Toast.LENGTH_SHORT).show();;}
            }
            finish();
        }
    });
}

private BluetoothSocket createBluetoothSocket(BluetoothDevice device) throws IOException
{
    //crea un conexion de salida segura para el dispositivo
    //usando el servicio UUID
    return device.createRfcommSocketToServiceRecord(BTMODULEUUID);
}

@Override
protected void onResume()
{
    super.onResume();

    if (mChartView == null) {
        LinearLayout layout = (LinearLayout) findViewById(R.id.chart);
        mChartView = ChartFactory.getLineChartView(this, mDataset,
                mRenderer);
        mRenderer.setClickEnabled(true);
        mRenderer.setSelectableBuffer(100);

        layout.addView(mChartView, new LinearLayout.LayoutParams(
```

```
                LinearLayout.LayoutParams.FILL_PARENT,
LinearLayout.LayoutParams.FILL_PARENT));
            boolean enabled = mDataset.getSeriesCount() > 0;
        } else {
            mChartView.repaint();
        }

        //Consigue la direccion MAC desde DeviceListActivity via intent
        Intent intent = getIntent();
        //Consigue la direccion MAC desde DeviceListActivity via EXTRA
        address = intent.getStringExtra(DispositivosBT.EXTRA_DEVICE_ADDRESS);//<-<- PARTE
A MODIFICAR >->->
        //Setea la direccion MAC
        BluetoothDevice device = btAdapter.getRemoteDevice(address);

        try{
            btSocket = createBluetoothSocket(device);
        } catch (IOException e) {
            Toast.makeText(getBaseContext(), "La creación del Socket fallo",
Toast.LENGTH_LONG).show();
        }
        // Establece la conexión con el socket Bluetooth.
        try{
            btSocket.connect();
        } catch (IOException e) {
            try {
                btSocket.close();
            } catch (IOException e2) {}
        }
        MyConexionBT = new ConnectedThread(btSocket);
        MyConexionBT.start();
    }

    protected class Update extends AsyncTask<Context, Integer, String> {
        @Override
        protected String doInBackground(Context... params) {

            TimeSeries dat = new TimeSeries("ECG");

            int i = 0;
```

```
    while (true) {
        try {
            Thread.sleep(50);

            x = x + 1;
            yy = (y * 1.5)/1024;
            xx = x * 0.1;

            publishProgress(i);
            i++;

            if (x == MaxSize){
                x = 0;
                mCurrentSeries.clear();
                //dat.clear();
            }

        } catch (Exception e) { }
    }
    // return "COMPLETE!";
}

// -- gets called just before thread begins
@Override
protected void onPreExecute() {
    super.onPreExecute();

}

@Override
protected void onProgressUpdate(Integer... values) {
    super.onProgressUpdate(values);

    LinearLayout layout = (LinearLayout) findViewById(R.id.chart);

    mCurrentSeries.add(xx, yy);

    if (mChartView != null) {
        mChartView.repaint();
    }
```

```
        }

        // -- called if the cancel button is pressed
        @Override
        protected void onCancelled() {
            super.onCancelled();

        }

        @Override
        protected void onPostExecute(String result) {
            super.onPostExecute(result);
        }
    }


    @Override
    public void onPause()
    {
        super.onPause();
        try
        { // Cuando se sale de la aplicación esta parte permite
            // que no se deje abierto el socket
            btSocket.close();
        } catch (IOException e2) {}
    }

    //Comprueba que el dispositivo Bluetooth Bluetooth está disponible y solicita que se active si está
desactivado
    private void VerificarEstadoBT() {

        if(btAdapter==null) {
            Toast.makeText(getBaseContext(), "El dispositivo no soporta bluetooth",
Toast.LENGTH_LONG).show();
        } else {
            if (btAdapter.isEnabled()) {
            } else {
                Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(enableBtIntent, 1);
            }
```

```
    }
}

//Crea la clase que permite crear el evento de conexion
private class ConnectedThread extends Thread
{
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;

    public ConnectedThread(BluetoothSocket socket)
    {
        InputStream tmpIn = null;
        OutputStream tmpOut = null;
        try
        {
            tmpIn = socket.getInputStream();
            tmpOut = socket.getOutputStream();
        } catch (IOException e) { }
        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }

    public void run()
    {
        byte[] buffer = new byte[256];
        int bytes;

        // Se mantiene en modo escucha para determinar el ingreso de datos
        while (true) {
            try {
                bytes = mmInStream.read(buffer);
                String readMessage = new String(buffer, 0, bytes);
                // Envia los datos obtenidos hacia el evento via handler
                bluetoothIn.obtainMessage(handlerState, bytes, -1, readMessage).sendToTarget();
            } catch (IOException e) {
                break;
            }
        }
    }
    //Envio de trama
```

```
    public void write(String input)
    {
      try {
        mmOutStream.write(input.getBytes());
      }
      catch (IOException e)
      {
        //si no es posible enviar datos se cierra la conexión
        Toast.makeText(getBaseContext(), "La Conexión fallo", Toast.LENGTH_LONG).show();
        finish();
      }
    }
  }
}
```