UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES

TEMA:

"SISTEMA DE TELEMEDICINA PARA MONITOREO CONTINUO DE CONSTANTES VITALES EN LACTANTES MENORES PARA EVITAR EL SÍNDROME DE MUERTE SÚBITA"

Trabajo de Graduación Modalidad: Proyecto de Investigación, presentado previo la obtención del título de Ingeniero en Electrónica y Comunicaciones

LÍNEAS DE INVESTIGACIÓN: Física y Electrónica.

AUTOR: Luis Alberto Ruiz Narváez

TUTOR: Ing. Edgar Patricio Córdova Córdova, Mg.

Ambato – Ecuador

Abril, 2018

APROBACIÓN DEL TUTOR

En mi calidad de Tutor del Trabajo de Investigación sobre el Tema:

"SISTEMA DE TELEMEDICINA PARA MONITOREO CONTINUO DE CONSTANTES VITALES EN LACTANTES MENORES PARA EVITAR EL SÍNDROME DE MUERTE SÚBITA", del señor, Luis Alberto Ruiz Narváez estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el numeral 7.2 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ambato, Abril de 2018

EL TUTOR

Ing. Edgar Patricio Córdova Córdova, Mg.

AUTORÍA

El presente Proyecto de Investigación titulado: SISTEMA DE TELEMEDICINA PARA MONITOREO CONTINUO DE CONSTANTES VITALES EN LACTANTES MENORES PARA EVITAR EL SÍNDROME DE MUERTE SÚBITA. Es absolutamente original, auténtico y personal, en tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, Abril 2018

Luis Alberto Ruiz Narváez

CC: 1804486122

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación. Cedo los derechos de mi Trabajo de Titulación, con fines de difusión pública, además autorizo su reproducción dentro de las regulaciones de la Universidad.

Ambato, Abril de 2018

Luis Alberto Ruiz Narváez

CC: 1804486122

APROBACIÓN COMISIÓN CALIFICADORES

La Comisión Calificadora del presente trabajo conformada por los señores docentes Ing. Marco Jurado e Ing. Patricio Encalada, revisó y aprobó el Informe Final del trabajo de graduación titulado SISTEMA DE TELEMEDICINA PARA MONITOREO CONTINUO DE CONSTANTES VITALES EN LACTANTES MENORES PARA EVITAR EL SÍNDROME DE MUERTE SÚBITA, presentado por el señor Luis Alberto Ruiz Narváez de acuerdo al numeral 9.1 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ing. Pilar Urrutia

PRESIDENTE DEL TRIBUNAL

Ing. Marco Jurado

DOCENTE CALIFICADOR

Ing. Patricio Encalada

DOCENTE CALIFICADOR

DEDICATORIA

El presente trabajo lo dedico a mis padres quienes han sido mi guía y mi inspiración, los promotores de mi futuro de los que siempre he recibido soporte absoluto.

A mi novia Daniela quien ha sido mi ayuda incondicional a lo largo de mi camino estudiantil tanto en mis triunfos como derrotas, quien supo sacar lo mejor de mi.

A mi familia y amigos en quienes siempre encontré apoyo en todos mis objetivos planteados.

AGRADECIMIENTO

A Dios dueño de mi vida quien me ha permitido ver cada derrota como una oportunidad de ser mejor y superarme, quien ha puesto a las personas indicadas en mi camino para alcanzar mis metas y sin quien nada de lo que me propongo pudiera ser realidad.

A mis padres y hermanas por cada sacrificio realizado en mi nombre y cada consejo útil, a quienes les debo todos mis triunfos.

A mi novia Daniela quien me ha brindado su amor incondicional e inspirado ha ser mejor persona cada día y siempre luchar por mis sueños.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

En el presente proyecto de titulación se describe un sistema de telemedicina para uso doméstico, diseñado para el monitoreo continuo de los signos vitales, capaz de enviar señales de alertas en caso de detectar alteraciones, con lo que previene condiciones clínicas graves o incluso la muerte. Este instrumento presenta información amplia del estado del bebé, lo que es especialmente útil para lactantes prematuros, con poco peso, recientemente operados o con problemas congénitos, ya que son los pacientes con mayor riesgo de muerte súbita.

El impacto de este proyecto es la contribución que se da a la comunidad en general, pues este dispositivo está dirigido principalmente al área de biomedicina, brindando una tecnología usable y adaptable, conformada por la interfaz de visualización, alerta y configurable a las necesidades crecientes en los tiempos actuales, con el objetivo de reducir la mortalidad por Síndrome de Muerte Súbita en lactantes.

Palabra clave— Síndrome; la muerte súbita; usable; infantes menores de edad; saturación de oxígeno; pulsaciones; signos vitales.

ABSTRACT

In this project to present a telemedicine system for home use, designed for the continuous monitoring of vital signs, capable of sending warning signals in case of detecting alterations, thus preventing serious clinical conditions or even death. This instrument presents extensive information on the baby's condition, which is especially useful for premature, light-weight, recently operated or congenitally-born infants, since they are the patients with the highest risk of sudden death.

The impact of this project is the contribution that is given to the community in general, because this device is directed mainly to the area of biomedicine, providing a usable and adaptable technology, shaped by the visualization interface, alert and configurable to the growing needs in current times, with the aim of reducing mortality due to Sudden Death Syndrome in infants.

Keyword— Syndrome; sudden death; usable; infants under age; oxygen saturation; pulsations; Vital signs.

GLOSARIO DE TÉRMINOS Y ACRÓNIMOS

- ADC: Analog to Digital Converter
- Android: Sistema operativo destinado para teléfonos móviles basado en un núcleo Linux.
- Arduino: Plataforma de hardware libre, constituida por una placa con un microcontrolador AVR con pines de entrada y salida y un entorno de desarrollo de libre acceso de baja complejidad de uso.
- BLE: Bluetooth Low Energy ECG: Electrocardiograma
- EEPROM: Electrical: Electrically Erasable Programmable Read-Only Memory
- GUI: Graphical User Interface
- Hardware: Conjunto de elementos o materiales físicos que constituyen un sistema electrónico informático
- IDE: Integrated Development Environment INEC: Instituto Nacional de Estadísticas y Censos
- Interfaz: Medio que permite comunicar dos o más dispositivos o maquinas. Extensión para la comunicación entre un sistema o subsistemas.
- Microcontrolador: Circuito integrado programable, capaz de almacenar y ejecutar ordenes pre grabadas en su memoria.

- MSP: Ministerio de Salud Pública del Ecuador
- PCB: Printed Circuit Board
- HR: Pulsos Cardiacos (Hearth rate)
- SMSL: Síndrome de muerte súbita del lactante
- Sensor: dispositivo capaz de detectar magnitudes físicas o químicas y convertirlas en variables de naturaleza eléctrica.
- SMART: Self Monitoring Analysis and Reporting Technology SRAM: Static Random Access Memory
- UART: Universal Asynchronous Receiver-Transmitter
- WBAN: Wireless Body Area Network
- Wearable Computing: prendas electrónicas usables que permiten al individuo usarlas por arriba o debajo de la vestimenta
- WPAN: Wireless Personal Area Network
- SPO2: medida del porcentaje de moléculas de hemoglobina unidas al oxígeno.

INTRODUCCIÓN

Los últimos avances tecnológicos han permitido la medición de signos vitales de forma no invasiva, con rápido crecimiento en micro/nano circuitos integrados de baja potencia y comunicación móvil, permitiendo el uso de una nueva generación de redes de sensores inalámbricos que generan indicadores de normalidad y brindan al paciente la capacidad de un monitoreo continuo domiciliario.

El presente proyecto de investigación se encuentra estructurado de tal manera que, en el Capítulo I se describe el problema generado por el sindrome de muerte súbita en lactantes menores y su impacto social, asi como la justificación que sustenta el desarrollo del proyecto.

El Capítulo II esta dedicado al análisis de trabajos investigativos relacionados con el monitoreo de signos vitales en lactantes por medio de dispositivos electrónicos. Además de la fundamentación que sustentará teóricamente el presente proyecto de titulación; finalizando con la propuesta que daría solución a la problemática presentada en el Capítulo I.

El Capítulo III engloba el desarrollo de la metodología utilizada para el cumplimiento del proyecto propuesto.

En el Capítulo IV se acota la descripción minuciosa del desarrollo del sistema de telemedicina para el monitoreo de signos vitales y las características de cada uno de sus componentes.

Finalmente en el Capítulo V se presentan las conclusiones obtenidas posterior al desarrollo del trabajo y las respectivas recomendaciones, que orienten a futuras investigaciones que se deriven de este trabajo.

Capítulo I El problema

1.1. Tema de Investigación

"Sistema de telemedicina para monitoreo continuo de constantes vitales en lactantes menores para evitar el síndrome de muerte súbita."

1.2. Planteamiento del Problema

En el trabajo realizado por Emely Licet Morales Rúa y colaboradores refieren que el síndrome de la muerte súbita del lactante (SMSL) constituye la principal causa de mortalidad en infantes entre un mes y un año de vida, y es la tercera causa de mortalidad infantil en los Estados Unidos. Su ocurrencia presenta un pico máximo entre los 2 y 4 meses de edad y además de ello los hermanos en futuras gestas tiene 5.9 veces más de riesgo de SMSL. A pesar de ser un diagnóstico de exclusión, puede tomarse como causa básica de muerte luego de comprobar que se trató de un lactante en quien no existió hallazgo patológico que explicase su muerte. [1], [2]

La primera causa de muerte súbita en niños es cardiovascular y el trabajo en equipo de forenses, patólogos, cardiólogos, pediatras, microbiólogos, investigadores básicos y especialistas en metabolopatías es clave, tanto para el correcto diagnóstico de la causa última de la muerte como para evitar nuevas muertes súbitas en las familias ya afectadas y en la población general. [3]

El INEC en el 2014, Ecuador presentó una tasa de mortalidad infantil de 17,93 por 1000 nacidos vivos, sin diferencia significativa entre sexos. Las principales causas de mortalidad infantil están directamente asociadas con complicaciones que ocurren en el período neonatal, además reportaron para el mismo año 15 muertes súbitas en menores de 1 año, 10 hombres y 5 mujeres, con mayor incidencia en la edad

comprendida entre 28 días y 11 meses. Adicionalmente, de las 1431 muertes en el período neonatal precoz registradas por el INEC en 2008, un 33,2% sucedieron en el primer día del nacimiento y 63,8 %, entre los días 0 y 3. [4], [5], [6]

Según el Anuario de Estadísticas Vitales - Nacimientos y Defunciones del 2014 en la zona tres del país, conformada por las provincias de Cotopaxi, Chimborazo, Pastaza y Tungurahua, se ha detectado la siguiente mortalidad en menores de 1 año, como se muestra en la Tabla 1.1:

Tabla 1.1. Defunciones menores de un año por sexo, según regiones, provincias y áreas de residencia habitual [7]

Regiones, provincias y áreas	**Defunciones en menores de un año**		
	Total	**Hombres**	**Mujeres**
Cotopaxi	85	47	38
Urbana	46	32	14
Rural	39	15	24
Chimborazo	121	62	59
Urbana	72	39	33
Rural	49	23	26
Pastaza	17	12	5
Urbana	13	8	5
Rural	4	4	0
Tungurahua	92	50	42
Urbana	50	24	26
Rural	42	26	16
Total	315	171	144
Total Urbana	181	103	78
Total Rural	134	68	66

Dado que en la zona 3 se registra un total de 315 fallecimientos en menores de un año, en relación a una natalidad de 22978, es importante destacar que la prevención de la

mortalidad en este grupo etario es de suma importancia, ya que un descuido a este nivel representaría un mal funcionamiento del sistema de salud.

Se debe sopesar dentro de los factores de riesgo los embarazos mal controlados, partos domiciliarios, oportunidades perdidas de educación para la salud y falta de nivel de alarma. En este contexto, la muerte súbita del lactante podría tener una alta incidencia y aunque no sea la primera causa de mortalidad infantil, representa una patología potencialmente prevenible. [7]

1.3. Delimitación

DELIMITACIÓN DE CONTENIDOS

Área Académica: Física y Electrónica.

Línea de Investigación: Sistemas de Control.

Sublínea de Investigación: Sistema Embebidos.

DELIMITACIÓN ESPACIAL

El presente trabajo investigativo se llevó a cabo en la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato.

DELIMITACIÓN TEMPORAL

El proyecto de investigación descrito, se ha desarrollado en el período Marzo 2017 - Agosto 2017 de acuerdo a lo establecido en el Reglamento de graduación para obtener el título terminal de tercer nivel de la Universidad Técnica de Ambato.

1.4. Justificación

El INEC menciona que dentro de nuestro país a pesar de que la mortalidad infantil en menores de un año ha disminuido en los últimos 14 años, aún el porcentaje del 4,43% es importante, siendo el grupo etario más afectado el comprendido entre 28 días y 11 meses. Es importante destacar que tan solo se diagnosticaron 15 muertes al año por síndrome de muerte súbita pero también existieron 153 muertes con causa de muerte

mal definidas y las no especificadas, mismos que si hubieran tenido la investigación adecuada se englobarían dentro del síndrome de muerte súbita. [7]

En vista de que la etiología de la muerte súbita es compleja y de origen cardiovascular en su mayoría, y que los casos de muerte por este síndrome se han incrementado a nivel mundial y en nuestro medio, este síndrome representa una enfermedad de alto impacto social y epidemiológico cuya prevención es imperativa. [2]

La presente investigación está enfocada en el monitoreo continuo de las señales vitales, con el fin de presentar un panorama más amplio, de cómo se encuentra el estado del lactante menor y a su vez de generar una tranquilidad a sus padres o cuidadores; ya que se van a encontrar alertados en caso de anormalidades detectadas; es así que este proyecto se realizará por medio de tres etapas, la primera comprende en la captación de los signos vitales a travez de dispositivos open-source, la segunda etapa es el procesamiento y envio de datos y la tercera etapa es la interfaz de visualización y almacenamientos de datos, asi como la generación de alertas en caso de detección de alteraciones en los rangos preestablecidos de normalidad según la Asociación Americana del Corazón en la cual se ha basado.

El aporte que genera este dispositivo es de alto impacto para la sociedad, ya que al permitir un monitoreo continuo del lactante es capaz de reducir la mortalidad neonatal y el impacto psicológico en el núcleo familiar, además de ser un equipo innovador ya que en el mercado local no se encuentra dispositivos capaces de realizar lo que en este proyecto se propone, con enfoque en lactantes menores.

Los principales beneficiarios de este dispositivo serán: recién nacidos prematuros, con bajo peso al nacer, neonatos en recuperación posquirurgica o con malformaciones congénitas, ya que son los que más riesgo de muerte súbita presentan.

1.5. Objetivos

1.5.1. General

Implementar el sistema de telemedicina para monitoreo continuo de constantes vitales, de lactantes menores para evitar el síndrome de muerte súbita.

1.5.2. Específicos

- Analizar las características del síndrome muerte súbita del lactante menor.
- Diseñar un dispositivo de monitoreo, alerta y registro de las señales vitales y las alteraciones de manera inalámbrica del lactante menor.
- Desarrollar una aplicación para dispositivos móviles, que permita alertar oportunamente de las alteraciones de las constantes vitales del lactante.

Capítulo II Marco Teórico

2.1. Antecedentes Investigativos

Revisando varios trabajos de investigación, relacionados con tecnologías destinadas al monitoreo médico electrónico y equipos destinados a la detección del Síndrome de Muerte Súbita en Lactante (SMSL), se procede a describir varias prácticas de mayor relevancia en los últimos tiempos:

En el 2013 la revista Horizons "A Supplement to biomedical instrumetation & technology" de SPRING, menciona que un grupo de estudiantes de Brigham Young University (BYU), han desarrollado un dispositivo que reducirá el número de casos de SMSL cada año, el cual describe una Wireless Body Area Network (WBAN). Representando un modelo de red implantable en el medio del lactante en forma de un calcetín el cual estará atado a su pie, utilizando la oximetría de pulso para monitorizar la frecuencia cardiaca y los niveles de oxígeno en la sangre, a su vez si el lactante deja de respirar o hay un cambio en el marcado en la frecuencia cardiaca, el monitor avisará a los padres a través de un teléfono inteligente. [8]

En el artículo "Mobile Wearable Nano-Bio Health Monitoring Systems with Smartphones as Base Stations" de Vijay K. Varadan and Linfeng Chen en el 2012, mencionan el desarrollo y la aplicación de móviles en nano-bio, sistemas de monitoreo para la telemedicina y que en un sistema de este tipo, los biosensores basados en nanomateriales se utilizan para medir señales fisiológicas, tales como electrocardiograma (ECG), electroencefalograma (EEG), electromiograma (EMG) y electrooculograma (EOG). Las señales fisiológicas obtenidas son filtradas, amplificadas y transmitidas a un servidor de almacenamiento remoto, utilizando teléfonos inteligentes como estaciones base. Los recursos de computación en nube se

utilizan para cálculos complejos, como la extracción de características y el diagnóstico automático. La información en el servidor de almacenamiento remoto puede ser accedida instantáneamente por los proveedores de atención médica, y el asesoramiento médico también se puede enviar al instante al paciente a través del sistema de comunicación inalámbrica. [9]

En el paper "An Innovative Approach for Infant Monitoring System using Pulse Rate and Oxygen Level" desarrollado por Mrudula Borkar, Neha Kenkre, Harshada Patke Ankita Gupta, consta de una metodología la cual trata de superar las limitaciones de sistemas convencionales, los cuales se compone principalmente de sensores, unidad de hardware, servidor de nube y la aplicación móvil en los smartphones de los padres. El sistema tomará dos entradas de los sensores que son la frecuencia del pulso y el nivel de oxígeno y a su vez servidor Cloud enviará estos datos al teléfono de los cuidadores haciendo uso de Internet. El sistema de alerta se activará y comenzará a alarmar si hay lecturas problemáticas, ayudando a los progenitores a entender la situación y tomar acciones inmediatas sobre el mismo. [10]

El el articulo "A Smart Wearable System for Sudden Infant Death Syndrome Monitoring" desarrollado por André G. Ferreira, Duarte Fernandes, Sérgio Branco, João L. Monteiro, Jorge Cabral, André P. Catarino, Ana M. Rocha, menciona que el Dispositivo IoT Wearable recoge diferentes tipos de datos fisiológicos y envía esos parámetros al Gateway, que está dentro del rango de comunicación del dispositivo IoT portátil. El Dispositivo IoT Wearable adquiere los parámetros fisiológicos del lactante, los analiza y envía los datos procesados al Gateway, teniendo un registro de todos los signos vitales del lactante y para hacerlo más cómodo para el paciente se usó el módulo GB2530-L de GBAN (CC2530 con factor de forma pequeño) y un PCB con dimensiones reducidas (4.3x3.8x0.9cm). [11]

El paper "Infant Monitoring System Using Multiple Sensors", desarrollado por Sakshi Gupta, Zarnain M. Khan, Rupali Srivastava, P.A. Chougule, menciona que su dispositivo funciona para el monitoreo continuo de signos vitales los cuales tiene un enfoque que es ampliamente utilizado para la comunicación de larga distancia, alta velocidad y confiabilidad. Este tipo de comunicación se puede utilizar en el hogar como una red WBAN, así como en los hospitales para los sistemas de monitoreo

central. Además de su función de monitoreo, consta de una dispositivo zumbador y se le ha establecido límites de umbral superior e inferior para los que se activa el zumbador y poder alertar de las alteraciones. [12]

En el review desarrollado por Zhihua Zhu, Tao Liu, Guangyi Li, Tong Li y Yoshio Inoue sobre "Wearable Sensor Systems for Infants", señala que como una herramienta significativa en la práctica clínica y la investigación de la conducta humana, los sistemas de sensores portátiles han atraído cada vez más la atención de diversas maneras, tales como la integración exitosa en la ropa infantil y los dispositivos textiles electrónicos, los sistemas de sensores portátiles se están volviendo discretos y no invasivos para los bebés. Los métodos y técnicas de monitoreo, como los de un solo parámetro Monitoreo, monitoreo multiparamétrico y tecnología de electrodos textiles en sistemas de sensores portátiles, fueron revisados de acuerdo a algunas investigaciones recientes y aplicaciones en el campo. Se pueden esperar y explotar perspectivas de aplicación significativas para el desarrollo sanitario y físico de los lactantes. Además, cabe destacar que los sistemas de sensores usables intercambian la información adquirida de los niños con equipos auxiliares que se pueden usar, y los equipos auxiliares, tales como actuadores o dispositivos de protección, pueden comenzar a trabajar para ayudar a los bebés o padres a completar algunas acciones, incluyendo vestirse, bañarse o amamantar cuando es necesario, lo que puede reducir en gran medida la carga sobre los padres o cuidadores. [13]

2.2. Fundamentación Teórica

2.2.1. Lactantes

Se considera lactante al período comprendido los 29 días de vida hasta los 2 años de edad. Este periodo a su vez puede dividirse en dos sub-periodos. [14]

- Lactante Menor: desde los 29 días de nacido hasta los 12 meses de edad.
- Lactante Mayor: desde los 12 meses de edad hasta los 24 meses de edad.

2.2.2. Síndrome de Muerte Súbita en Lactantes (SMSL)

El SMSL es un evento de muy alto impacto social. Son muchas las causas desencadenantes, pero la muerte súbita es una muerte natural rápida e inesperada en menores de 1 año de edad. El abordaje de esta patología debe agrupar en una misma línea de trabajo a patólogos, forenses, cardiólogos y pediatras para establecer con certeza la causa de muerte. En la Figura 2.1 podemos observar la triada de propensión para presentarse el SMSL. [3]

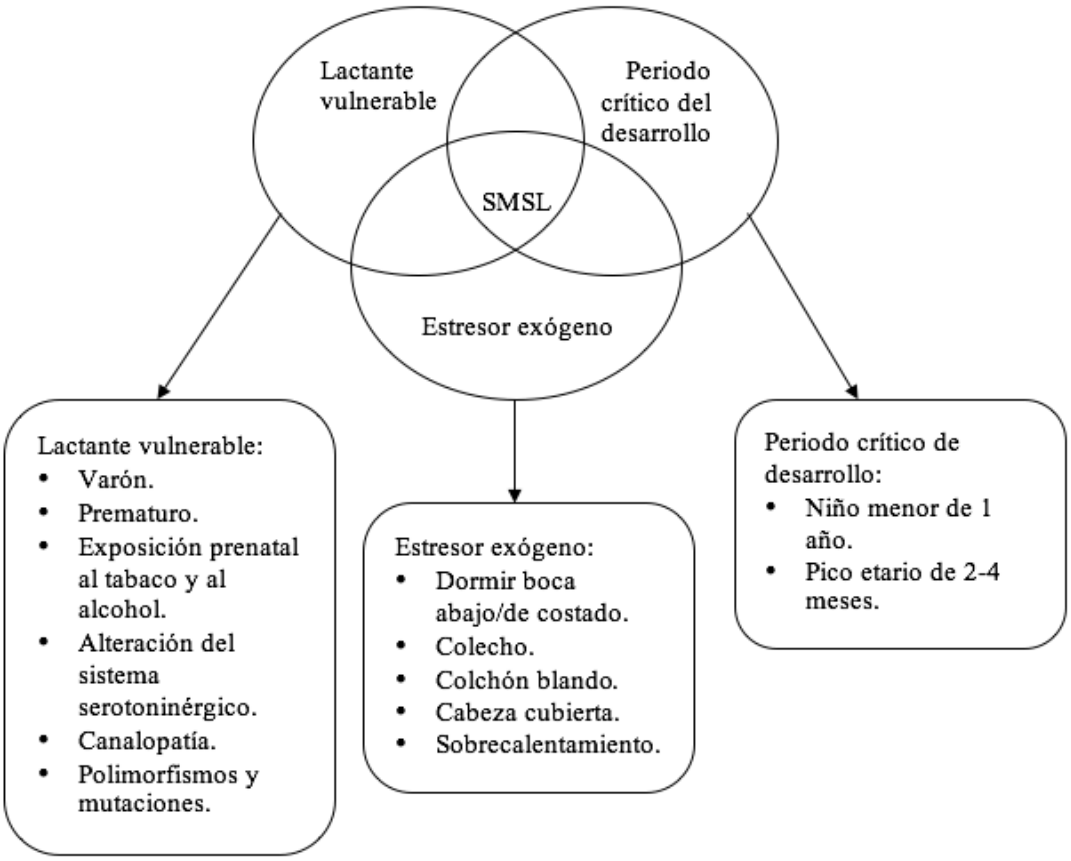


Figura 2.1. Hipótesis del triple riesgo [15]

Filiano y Kinney en 1994, asocian el concepto de 'triple riesgo' de SMSL que incluye, la vulnerabilidad intrínseca del lactante debida a factores biológicos, un período crítico del desarrollo comprendido entre los 2 y 6 meses de edad (período en el cual el control homeostático del cuerpo aún no ha madurado totalmente) y un factor estresante posnatal exógeno que desencadena el episodio. La convergencia de los tres grupos conduciría a la falla en la respuesta normal, produciendo asfixia, bradicardia e

hipotensión, llevando por último a la muerte. Es asi que los mecanismos madurativos anatómicos y fisiológicos condicionarían alteraciones en sus funciones vitales y en el control cardiorrespiratorio. [16].

Descripción del triple riesgo:

1. Lactante Vulnerable

Se considera lactante vulnerable al que presenta anomalías cerebrales, de predominio en el tronco cerebral o retraso madurativo relacionado con neurorregulación o control cardiorrespiratorio. Esto basado en lo siguiente:

- Factores de riesgo maternos y prenatales que indican un ambiente intrauterino menos óptimo. Teniendo como consecuencia la existencia de un trastorno cerebral, que puede tener su origen antes del nacimiento.

- Alteraciones sutiles en la regulación cardíaca, respiratoria y en los patrones de excitación del sueño, principalmente las alteraciones en la señalización de la serotonina (5HT) produciendo maduración tardía en las regiones cerebrales que participan en la respuesta ventilatoria y en la presión arterial frente a la hipoxia y la hipercapnia. Estas anomalías principalmente relacionadas al tabaquismo materno.

El papel que juegan los factores genéticos en la aparición del SMSL aún no está claro, aunque se han propuesto polimorfismos genéticos en la etiopatogenia de este síndrome; como las mutaciones específicas en la subunidad B del gen SCN5A, gen que codifica el canal de sodio cardíaco, alteración corroborada en el 5-10% de los infantes que han muerto con SMSL.

2. Período crítico del desarrollo

Se ha considerado que la frecuencia máxima se encuentra entre 2 y 4 meses de edad, debido a que es durante este periodo donde se culmina el desarrollo cardíaco y pulmonar, además de establecerse los patrones del sueño.

3. Estresores exógenos

Los factores exógenos, son potencialmente prevenibles y estan relacionados como desencadenantes ambientales del síndrome.

Como principal factor exógeno se encentra la posición en decúbito prono durante el sueño como se muestra en la Tabla 2.1:

Tabla 2.1. Mecanismos posibles relacionados con el riesgo de SIMS asociado a la posición en decúbito prono [15].

Causa	Efecto
Respiración del CO2 exhalado	Disminución de la pérdida de calor y peligro de sobrecalentamiento
Disminución de los microdespertares	Disminución de la oxigenación cerebral.
Alteración del control autonómico del sistema cardiovascular	Disminución de la presión arterial

En cuanto a la posición en decúbito lateral, se ha reportado que también aumenta el riesgo de presentar un SMSL, a la vez que se reconoce como posición inestable, ya que es probable que el infante ruede y quede en posición prona.

Otros desencadenantes ambientales incluyen el colecho, las superficies blandas durante el sueño, el sofocamiento, entre otros. [15]

2.2.3. Frecuencia Cardiaca

Frecuencia cardiaca u onda pulsátil de la sangre, es la que se produce por la contracción del ventrículo izquierdo del corazón, teniendo de esta manera que la extensión de las arterias se regulen mediante la expansión y contracción. También se menciona que es la cantidad de sangre que entra en las arterias con cada sístole ventricular y la adaptación de las arterias, mediante la contracción y dilatación. En la Tabla 2.2 se evidencia los rangos normales de la frecuencia cardiaca por grupo de edad en pacientes pediátricos. [17]

Tabla 2.2. Frecuencias cardíacas por edad. [18]

Edad	Frecuencia Despierto	Promedio	Frecuencia Dormido
Recién nacido hasta 3 meses	85 – 205	140	80 – 160
Niños de 3 mese hasta 2 años	100 – 190	130	75 – 160
Niños de 2 años hasta 10 años	60 – 140	80	60 – 90
Niños mayores a 10 años	60 – 100	75	50 – 90

2.2.4. Saturación de Oxígeno (Oximetría)

Saturación de oxígeno (Oximetría) es la cuantía de oxígeno que dispone el cuerpo humano en el torrente sanguíneo, es decir al respirar se introduce aire a nuestros pulmones para que así capten las moléculas de oxígeno y se unan a las células rojas de la sangre (eritrocitos), con el fin de ser transportados al resto del cuerpo [19]. Siendo esta la principal función del aparato respiratorio, debido a que si no se oxigena correctamente la totalidad del cuerpo, cada órgano o cada célula no cumplirían con su función adecuadamente y el cuerpo sufriría la escasez de oxigeno (hipoxia). [20]

Los valores normales de SaO2 oscilan entre 95% y 97%, con un rango de variación del 2%. Valores por debajo del 95% (en reposo) se asocian con situaciones patológicas y del 92-90% en pacientes con insuficiencia respiratoria crónica previa. [18]

2.2.5. Telemedicina

La telemedicina es un sistema tecnológico integrado que permite la prestación de servicios de salud sin contacto físico directo entre el profesional y el paciente, o entre profesionales entre sí, por medio de algún sistema telemático. En otras palabras, la telemedicina utiliza las tecnologías de la información y las telecomunicaciones para proporcionar o soportar la asistencia médica, independientemente de la distancia que

separa a los que ofrecen el servicio del paciente, este proceso se lleva acabo por medio de la movilización e intercambio digital de datos evitando en lo posible el desplazamiento de los pacientes y reduciendo los tiempos de respuesta y gastos asociados [21], orientados hacia una mayor equidad en la prestación de servicios, mayor preocupación por la efectividad y utilidad de las tecnologías para la salud. [22]

Por ello, la telemedicina hoy entra a desempeñar un papel fundamental en lo que concierne a la mejora sostenible de la salud de las comunidades. Es una herramienta más para el buen desempeño científico del personal de la salud, que no solucionará todos los problemas existentes en el sector sanitario, pero que con los avances generados durante los últimos quince años de las telecomunicaciones alámbricas e inalámbricas, entrará a desempeñar un rol de marcada importancia en todos los países del mundo.[23]

La telemedicina tiene como principales objetivos: [24]

- Prevenir, alertar, supervisar y controlar la expansión de enfermedades transmisibles y no transmisibles, mejorando la vigilancia epidemiológica.

- Contribuir a la integración del sistema de salud y la universalidad de los servicios de salud con calidad, eficiencia y equidad para beneficio prioritario de las poblaciones excluidas y dispersas.

- Promover la colaboración entre gobiernos, planificadores, profesionales de la salud, sociedad civil organizada y comunidades locales para crear un sistema de información y atención de salud fiable, y con calidad; fomentando así la capacitación, educación e investigación para la prevención y control de enfermedades.

- Agilizar la atención en salud, definiendo en tiempo real conductas a seguir (afinar los diagnósticos de los médicos en áreas rurales).

- Perfeccionar campañas preventivas de salud en la población.

- Justificar remisiones de pacientes o evitarlas si pueden ser de manejo del nivel del sitio de referencia a fin de no efectuar desplazamientos innecesarios.

- Facilitar diagnósticos más oportunos y tratamientos menos costosos por la oportunidad de una detección temprana de la enfermedad.

Estas actividades están en diversos niveles de desarrollo en cada uno de los países, pero, en todos estos campos, hoy se puede encontrar que la Telemedicina se usa, básicamente, en dos áreas de trabajo: la práctica médica y la educación.

En la práctica médica es posible resaltar las siguientes formas:

- Telediagnosis.
- Teleconsulta.
- Almacenamiento digital de datos o fichas electrónicas.

2.2.6. Monitoreo continuo de signos vitales

El monitoreo continuo es el proceso mediante el cual se registra de manera consecutiva las variaciones de los parámetros fisiológicos a investigarse, por medio de equipos diseñados para este fin. [25]

Los signos vitales son aquellos parámetros que indican el estado hemodinámico del paciente, van a estar controladas por los órganos principales que son: corazón, cerebro y pulmones, cuyo papel es la de dirigir las funciones del organismo ya que refleja los cambios que se producen en éste.

Los signos vitales son: [26]

- Temperatura corporal.
- Saturación de oxigeno
- Pulsos
- Respiración
- Tensión arterial.

2.2.7. Métodos de Adquisición de Signos Vitales

Los métodos de aquisición de signos vitales son específicos para cada uno, es asi que a continuación se detalla la adquisición de la frecuencia cardiaca y saturación de oxígeno que son las dos señales vitales que se monitorizaran en este proyecto.

Frecuencia cardiaca

Existen dos determinaciones de la frecuencia cardiaca, la frecuencia cardiaca periférica, la cual es equivalente al pulso y la frecuencia cardiaca central, la cual se define como las veces que late el corazón por unidad de tiempo. De estos métodos de adquisición se derivan: [27]

Método Empírico: El cual consiste en la palpación y detección de la dilatación arterial, de esta manera contabilizar la cantidad de latidos que produce las arterias en un minuto [28]

Electrocardiógrafo: método más fiable, el cual consiste en la toma y registro de la actividad de la corriente eléctrica que se genera en el corazón en un intervalo de tiempo, las cuales se obtienen mediante el uso de electrodos de plata con un gel conductor, los que se encuentran ubicados en diferentes secciones del tórax, herramienta que procesa dichas señales y presenta por medio tiras de papel milimetrados o mediante una pantalla. [29]

Saturación de Oxígeno (Oximetría)

El oxímetro de pulso tiene un método aplicado para la obtención de la saturación de oxígeno en la sangre (SpO2) mediante la emisión de rayos de luz que pasan a través de la sangre del dedo o del lóbulo de la oreja, mediante el proceso de recolección de la información los haces de luz son emitidos a través de los tejidos, de un lado del sensor hacia el otro, los cuales mediante la variación de color de la sangre es captado con facilidad por medio del sensor óptico, dicho color varía dependiendo de la cantidad de oxígeno saturado en la sangre, además de la aplicación de la onda de luz se genera una onda de pulso generada por la fluctuación de la sangre permitiendo obtener de una forma más efectiva los pulsos cardiacos. [30] [20]

2.2.8. Sistemas de adquisición y procesamiento de señales

La adquisición y tratamiento de las señales biológicas conllevan un proceso más complejo que cualquier otro tipo de señal, por lo que se establece el siguiente sistema de adquisición de señales vitales, iniciando primero con la detección de la señal fisiológica mediante un sensor como se muestra en la Figura 2.2, encargado de la transformación de la señal vital en señal eléctrica, así llegando a la etapa de acondicionamiento; fase en la cual se realiza un proceso de amplificación y filtrado analógico, seguidamente la señal es muestreada y convertida a señales digitales para su procesamiento mediante herramientas electrónicas. [31]

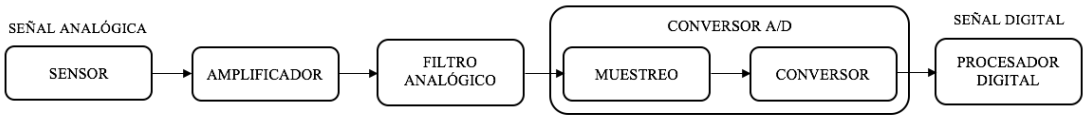


Figura 2.2. Sistema de adquisición de señales. [32]

En la etapa de amplificación se usa la transimpedancia de retroalimentación, con una resistencia y un capacitor como integrador debido que el voltaje de la salida que resulta es leído por un convertidor de analógico-digital y enviado para el microcontrolador, dado que el valor de la resistencia de retroalimentación es muy grande, el circuito es muy sensible a los cambios en intensidad, por lo que el diseño puede variar y ser de utilidad lograr una oscilación de la salida negativa como se muestran en la Figura 2.3. [33]

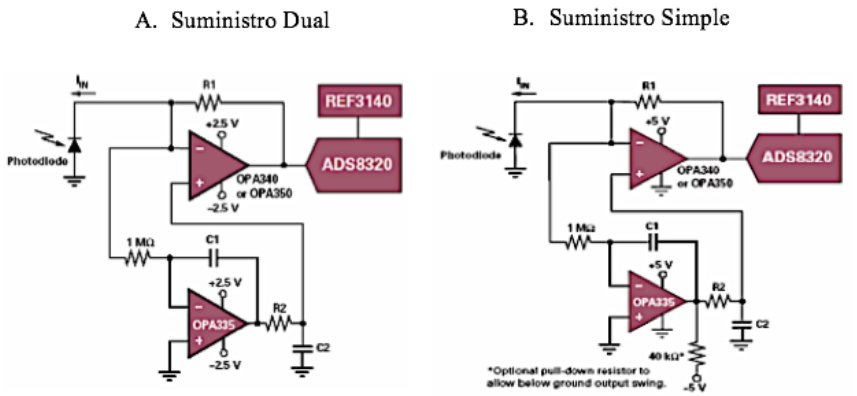


Figura 2.3. Configuraciones del amplificador de transimpedancia. [33]

En la etapa de procesamiento de la señal se puede utilizar microcontrolador monoplaca, de hardware libre, manejo sencillo y bajo coste, desarrollado para simplificar el uso de la electrónica, existen diferencias características en cuanto al número de entradas/salidas, pudiendo seleccionar el usuario según sus necesidades.

Los microcontroladores más habituales en Arduino son los de la familia AVR de ATMEL como se muestra en la Figura 2.4, aunque algunas plataformas utilizan otros microcontroladores, ejemplo Cortex M3 de ARM, de 32 bits.

Figura 2.4. Familia AVR de ATMEL [34]

Para facilitar su uso y programación se desarrolló simultáneamente y conjuntamente con Arduino un IDE (entorno de desarrollo integrado), en el que se usa un lenguaje de programación parecido a C++, basado en el lenguaje Wiring, el entorno de desarrollo está basado en Processing. El IDE permite editar, compilar y enviar el programa, así como comunicarse vía serial y mostrar los datos en una ventana terminal. Arduino se comunica con el IDE mediante un programa cargador o bootloader, el cual se encuentra precargado en el microcontrolador. EL IDE es de software libre y se puede descargar gratuitamente desde el sitio web oficial. [34]

2.2.9. Dispositivos Usables para la Salud

En el ámbito de la salud la tecnología wearable ha proporcionado innumerables ayudas para el monitoreo y la prevención de diferentes enfermedades, gradualmente se están integrando nuevas herramientas para el control de la salud permitiendo el ahorro de tiempo y dinero y a su vez aportando beneficios tanto para el paciente como para el médico. [35]

Además han ayudado a las redes de atención domiciliara para pacientes que no necesitan ser internados en el hospital, mediante los servicios de teleasistencia, teniendo una atención oportuna, mediante el uso de dispositivos wearable de control de parámetros biológicos como son; pulsímetro, pulsioxímetro, etc., los cuales están dotados de conectividad Wi-Fi y/o Bluetooth, siendo capaces de generar una alarma al servidor central en caso de detectar alguna anomalía.[36]

2.2.10. Dispositivos de Fuente abierta

Un dispositivo de fuente abierta es aquel que tenga su código disponible al público en general [37], eso quiere decir que tras su adquisición, puede ser utilizado, copiado, analizado, modificado y redistribuido por los usuarios con total libertad, gracias al código abierto u open source.

Aunque sea open source no hay que confundir "libre" con "gratuito", ya que también se puede distribuir comercialmente. Tampoco se debe confundir con el software de dominio público, el que no requiere de ningún tipo de licencias para su aplicación e implementación ya que pertenecen a todos, mientras que el open source, respetando sus principios fundamentales, se ejecuta con diferentes licencias para explotar los programas: GNU GPL, AGPL, de estilo BDS o MPL y derivadas.[38]

2.2.11. Red de Área Corporal

Tomando la definición planteada por la revista Sistemas telemática, las redes de área corporal son sistemas de comunicación a pequeña escala y se encuentran englobadas dentro del estándar de comunicaciones de los 80. Dentro de las comunicaciones de redes de área corporal son las que se encuentran dentro, cerca y alrededor de una persona, encontrando algunas tecnologías existentes que se adaptan perfectamente a dicha necesidad, como son: Bluetooth, ZigBee, millimeter-wabe y Ultra Wide Band.

Una de sus características es su área de cobertura ya que se encuentra confinada a distancias que no superan los 2 o 3 metros y la emisión de esta tecnología es muy baja lo que contribuye a la durabilidad de las baterías en los dispositivos.

Dentro de las principales aplicaciones de las redes de área corporal se encuentran englobados la medicina y entretenimiento. Las aplicaciones médicas se centran en el

monitoreo continuo de signos vitales del cuerpo humano ayudando de una mejor manera al control, diagnóstico y tratamiento de enfermedades. También se encuentran aplicaciones de entretenimiento y de intercambio de información teniendo como principal objetivo el intercambio de información sobre el estado del cuerpo al realizar dicha actividad física en un determinado lapso de tiempo. [39]

Una WBAN se encuentran constituida por nodos detalladas en la Figura 2.5, aquellos que están constituidos por sensores y actuadores los cuales están destinados a realizar una tarea específica en un determinado tiempo, los cuales están conectados a un dispositivo de transmisión. [40]

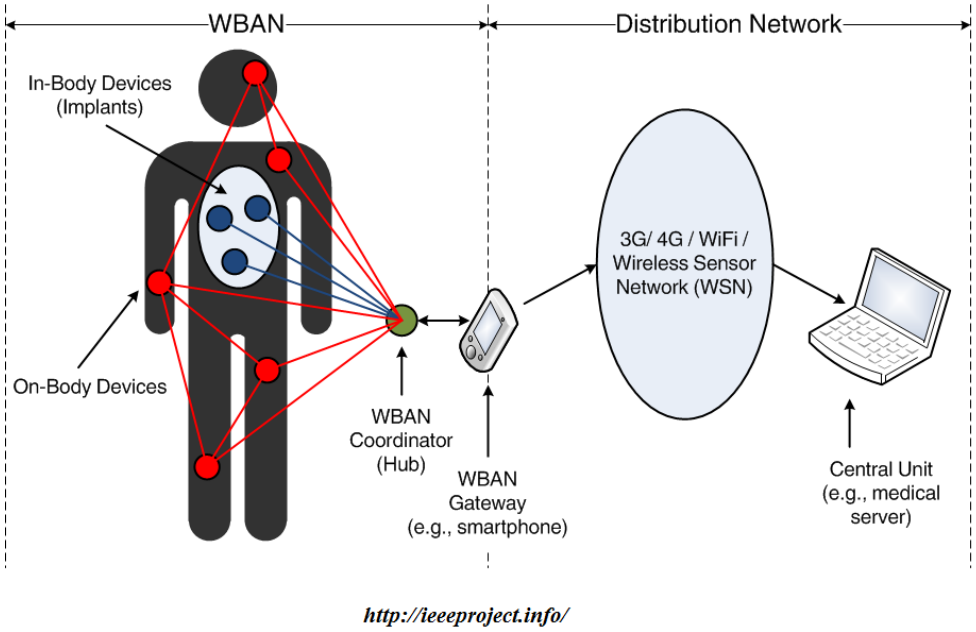


Figura 2.5. Red de área corporal WBAN [41]

2.2.12. Tecnologías de Transmisión de WBAN

Las tecnologías de redes de área personal en su totalidad, utilizan métodos de transmisión de datos inalámbricas, teniendo en claro que las WBAN están contenidas dentro de las redes PAN. A continuación la Tabla 2.3 describe las tecnologías de transmisión.

Tabla 2.3. Características técnicas de los dispositivos de transmisión de datos WBAN.

	Tecnología			
Parámetro	**Bluetooth**	**ZIGBEE**	**Infrarrojo**	**UWB**
Estándar	IEEE 802.15.1	IEEE 802.15.4	IRDA	IEEE 802.15.3
Frecuencia	2.4 Ghz	2.4 Ghz / 915 Mhz / 868 Mhz	850 nm	7.5Ghz
Modulación	GFSK	BPSK - QPSK	RZI - PPM	BPSK - QPSK
Alcance Máximo	10 metros	75 metros	Menor a 10 metros	50 metros
Capacidad Máxima	720 kbps	250 kbps / 40 kbps / 20 kbps	4Mbps	Entre 110 y 480 Mbps
Número de Nodos	8	65000	1	8
Penetración en el mercado	Alta	Media	En desuso	Baja
Consumo energético Tx	57 mA	24.7 mA	55mA aprox.	227.3 mA aprox.
Consumo Energético RX	46 mA	27 mA	55mA aprox.	227.3 mA aprox.

2.2.13. Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) de utilidad para la elaboración de aplicaciones Android, el cual está basado en IntelliJ IDEA; con sus herramientas y el editor de códigos, Android Studio tiene más funcionalidades que otros desarrolladores, lo que aumenta su productividad durante la compilación de apps, como las siguientes:

- Un sistema de compilación basado en Gradle flexible
- Un emulador rápido con varias funciones
- Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos Android

- Instant Run para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo APK
- Integración de plantillas de código y GitHub para ayudarte a compilar funciones comunes de las apps e importar ejemplos de código
- Gran cantidad de herramientas y frameworks de prueba
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK
- Soporte incorporado para Google Cloud Platform, lo que facilita la integración de Google Cloud Messaging y App Engine

Dentro de la estructura del proyecto se encuentra uno o más módulos con archivos de código fuente y archivos de recursos, como los siguientes:

- Módulos de apps para Android.
- Módulos de bibliotecas.
- Módulos de Google App Engine.

Cada módulo muestra los diferentes archivos del proyecto, organizandolos de una manera adecuada proporcionando un rápido acceso a los archivos de origen clave como se muestra en la Figura 2.6, teniendo visibles los archivos de compilación en el nivel superior de secuencias de comando de gradle, contienido en las siguientes carpetas: [42], [43]

- **Manifests**: contiene el archivo AndroidManifest.xml.
- **Java**: contiene los archivos de código fuente de Java, incluido el código de prueba JUnit.
- **Res**: Contiene todos los recursos, como diseños XML, cadenas de IU e imágenes de mapa de bits.

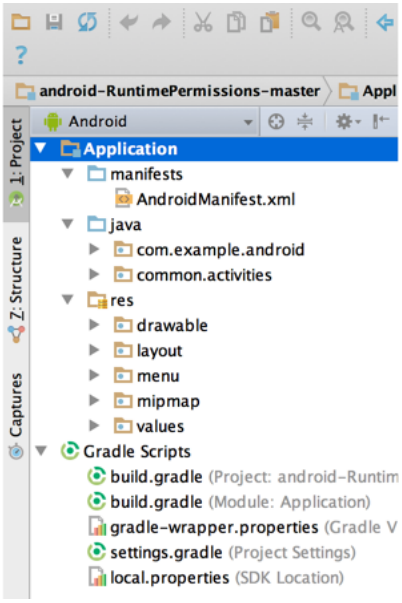


Figura 2.6. Archivos del proyecto en la vista de Android. [42]

La estructura del proyecto para Android puede diferir de esta representación plana. Además se puede personalizar la vista de los archivos del proyecto, teniendo un orden y control de todos los aspectos específicos del desarrollo de la app. Teniendo así la vista **Problems** como se muestra en la Figura 2.7, en donde aparecerán enlaces a los archivos de origen que contengan errores conocidos de codificación y sintaxis, como una etiqueta de cierre faltante para un elemento XML en un archivo de diseño. [44]

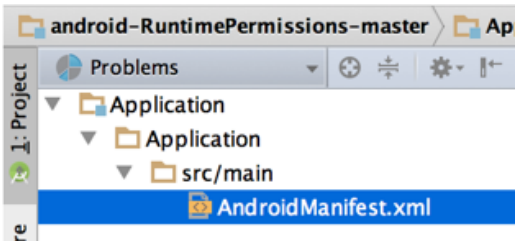


Figura 2.7. Archivos del proyecto en la vista Problems, en la que se muestra un archivo de diseño con un problema. [44]

La interfaz de usuario consta de una ventana principal y varias áreas lógicas que se identifican en la Figura 2.8 y se describen a continuación. [42], [45]

1. La **barra de herramientas** te permite realizar una gran variedad de acciones, como la ejecución de tu app y el inicio de herramientas de Android.

2. La **barra de navegación** te ayuda a explorar tu proyecto y abrir archivos para editar. Proporciona una vista más compacta de la estructura visible en la ventana **Project**.

3. La **ventana del editor** es el área donde puedes crear y modificar código. Según el tipo de archivo actual, el editor puede cambiar. Por ejemplo, cuando se visualiza un archivo de diseño, el editor muestra el editor de diseño.

4. La **barra de la ventana de herramientas** se extiende alrededor de la parte externa de la ventana del IDE y contiene los botones que te permiten expandir o contraer ventanas de herramientas individuales.

5. Las **ventanas de herramientas** te permiten acceder a tareas específicas, como la administración de proyectos, las búsquedas, los controles de versión, etc. Puedes expandirlas y contraerlas.

6. En la **barra de estado**, se muestra el estado de tu proyecto y del IDE en sí, como también cualquier advertencia o mensaje.

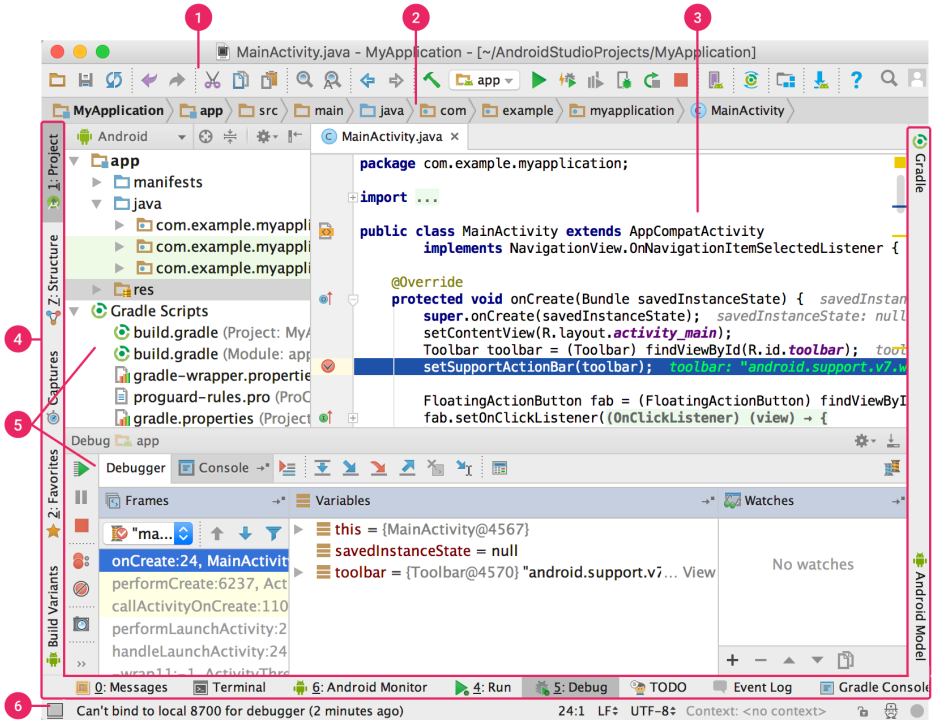


Figura 2.8. Ventana principal de Android Studio. [45]

2.3. Propuesta de Solución

Mediante la implementación del sistema de telemedicina para monitoreo continuo de constantes vitales, se pretende reducir la mortalidad por el síndrome de muerte súbita en lactantes menores mediante la detección de alteraciones en la saturación de oxígeno en la sangre o de la frecuencia cardíaca, emitiendo una estimulación lo suficientemente molesta, aunque inofensiva al lactante y al mismo tiempo transmite de forma inalámbrica una notificación audible y vibratoria a un dispositivo inteligente ya sea de los padres o cuidadores, y generar un registro de estos signos vitales con las cuales el médico especialista podrá evaluar y considerar el manejo adecuado del lactante.

Capítulo III Metodología

3.1. Modalidad de la Investigación

El presente proyecto se desarrolló utilizando los siguientes tipos de investigación:

Investigación aplicada, porque el objetivo principal fue resolver un problema práctico mediante el diseño e implementación, obteniendo datos comparativos y resultados reales.

Investigación bibliográfica, porque la explicación científica de las variables del proyecto de investigación se realizó basándose en libros, publicaciones y artículos científicos para lo cual se requirió una conexión a internet y consultas en bibliotecas públicas y privadas.

Investigación experimental, porque el desarrollo del proyecto requirió múltiples pruebas operacionales de los dispositivos electrónicos a emplear, recolectando información para corregir errores y pulir defectos.

3.2. Población y Muestra

Se realizó con una muestra de 10 lactantes, teniendo en cuenta que es el número que tiene la institución Sociedad Protectora del niño huérfano y abandonado Hogar Santa Marianita y de esta manera poder obtener la información pertinente.

3.3. Recolección de Información

La información recabada, se recopiló y obtuvo de fuentes y referencias bibliográficas realizadas en revistas de investigación científica, publicaciones, libros y artículos las cuales fueron provenientes del internet.

3.4. Procesamiento y Análisis de Datos

La información recabada se organizó de manera metodológica, contribuyendo de manera óptima para el desarrollo de la investigación, permitiendo la obtención de nuevos conocimientos con la ayuda de toda la información recolectada de diferentes fuentes fiables.

3.5. Desarrollo del Proyecto

Para el desarrollo de la investigación será necesario efectuar los siguientes pasos, los cuales se describen a continuación:

- Estudio del Síndrome de Muerte Súbita en Lactantes.
- Determinación de las características del Síndrome de Muerte Súbita en Lactantes menores.
- Estudio de métodos de adquisición de señales vitales.
- Testeo de dispositivos electrónicos de libre acceso, orientados a la adquisición de señales vitales.
- Acondicionamiento de señales eléctricas producidas por sensores de señales vitales.
- Diseño de dispositivo para la adquisición, procesamiento y envío de señales vitales.
- Análisis para localización óptima de sensores de señales vitales.
- Desarrollo de Interfaz entre sensores y sistema de adquisición de datos basado en hardware libre.
- Calibración de sistema de adquisición de señales.
- Desarrollo de aplicación para dispositivos móviles de media y alta gama para la presentación continúa de datos obtenidos.

- Enlace mediante radiofrecuencia (Bluetooth), entre el sistema de adquisición de datos y dispositivo de monitoreo.
- Pruebas de confiabilidad de sistema de monitoreo personal.
- Análisis de resultados obtenidos de pruebas del prototipo.

Capítulo IV Desarrollo de la Propuesta

El SMSL siempre ha representado un miedo constante y latente en los padres, principalmente los primerizos a los cuales se les hace difícil identificar signos y síntomas de peligro en los lactantes, es así que en diferentes países como la India y China basados en esta preocupación se decide realizar un prototipo que permita la monitorización de constantes vitales utilizando como fuente de adquisición de datos diferentes zonas corporales. En nuestro país no existe un antecedente de equipos electrónicos elaborados con este fin en nuestra población de estudio, por lo tanto se propone un sistema de telemedicina para el monitoreo continuo de signos vitales del lactante menor, mediante un equipo electrónico que cumpla con los requerimientos funcionalidad, es decir, además de brindar comodidad al momento del uso y brinde confiabilidad en la obtención de datos y generación de alertas.

4.1. Análisis de Factibilidad

El desarrollo del proyecto tuvo una factibilidad técnica, económica y bibliográfica de forma detallada a continuación:

4.1.1. Factibilidad Técnica

La realización del presente proyecto tiene factibilidad técnica dado que los equipos y elementos electrónicos necesarios para el desarrollo del prototipo del sistema de activación de alertas, se encuentran en el país.

4.1.2. Factibilidad Económica

El presente proyecto es económicamente factible debido a que el financiamiento de la investigación es solventada con los recursos económicos del investigador.

4.1.3. Factibilidad Bibliográfica

La presente investigación tiene factibilidad bibliográfica debido a que la información requerida se encuentra en libros, documentos científicos, revistas y documentos web.

4.2. Desarrollo

Con el propósito de orientar el desarrollo de este proyecto se a realizado un esquema general del prototipo basándose en la topología de una red de área corporal inalámbrica que permita la adquisición de las señales biológicas y se detalla a continuación (Figura 4.1).

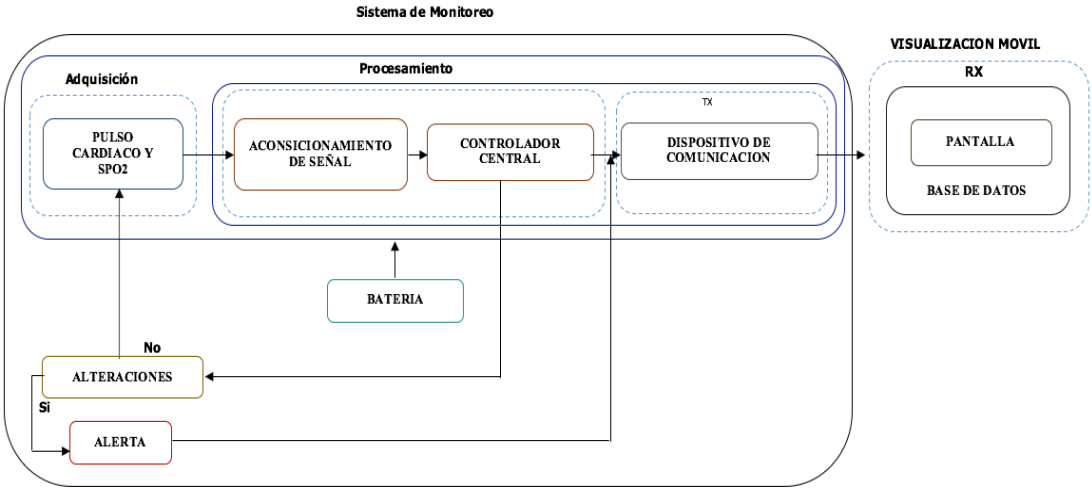


Figura 4.1. Esquema General del Sistema de Monitoreo.
Elaborado por: Investigador

Gracias al esquema general se obtiene un concepto más claro de las etapas del sistema de adquisición, procesamiento, envío y recepción de las constantes vitales, tomando en cuenta la ubicación de los sensores en puntos importantes y de fácil acceso.

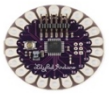
4.3. Análisis de los dispositivos

Para el proceso de selección de los equipos se realizó un análisis técnico de los dispositivos que cumplan los requerimientos preestablecidos por el investigador.

4.3.1. Controlador

En el campo de las placas destinadas para el desarrollo de prendas electrónicas o para el censado de señales vitales se presenta una comparativa a través de la Tabla 4.1 tomando en cuenta la accesibilidad, parámetros técnicos y de soporte que brinden a las mejores prestaciones.

Tabla 4.1. Análisis comparativo de placas electrónicas [46, 47, 48].

Controlador			
Parámetros Técnicos	Arduino Pro Micro	Arduino Lilypad	Flora Adafruit
			
Controlador	Atmega32u4	Atmega328/P	Atmega32u4
Voltaje de operación	5 – 12V	2.7 – 5.5 V	3.5 – 16V
Pines de entrada o salida digital	12	14	8
Pines de entradas analógicas	4	6	4
Pines PWM	5	6	4
Corriente de pines entrada y salida	40mA	40mA	40mA
EEPROM	1 KB	512 bytes	1 KB
SRAM	2.5 KB	1 KB	2.5 KB
Memoria Flash	32 KB	16 KB	23 KB
Velocidad de reloj	16 MHz	8 MHz	8MHz
Comunicaciones	SPI /I2C/UART	SPI /I2C/UART	SPI /I2C/UART
Dimenciones	3.31 x 1.78 cm	4.9 x 0.3 cm	4.5 x 0.7 cm
Peso	2 g	5 g	4.7 g
Interfaz de programación	Micro USB	FTDI	Micro USB
Sócalo de batería	No	No	Si
Lavable	No	Si	Si
Precio	11.25 USD	19,95 USD	14.95 USD

Elaborado por: Investigador

Teniendo en cuenta todas las características técnicas y la funcionalidad de cada una de las placas para la operatividad del sistema, se ha seleccionado al arduino pro micro

como controlador central debido a que cumple con los requerimientos; por el voltaje y corriente de operación, número de entradas y salidas, tamaño, fácil acceso geográfico y su costo.

4.3.2. Sensor de Oximetría

Para la medición de oximetría se requiere un dispositivo que permita el monitoreo en tiempo real y a su vez que sea compatible con el controlador central y encontrarse dentro del rango de trabajo energético del sistema, por lo que se realiza una tabla comparativa (Tabla 4.2) entre los posibles componentes a utlizar.

Tabla 4.2. Análisis comparativo de sensores oximetría [49, 50, 51].

Sensor de Pulsioximetría			
Parámetros técnicos	MAX30102	MAX30100	Keyestudio FR-4
			
Accesibilidad	Determinadas zonas geográficas	Determinadas zonas geográficas	Determinadas zonas geográficas
Métodos de adquisición	Oximetría de Pulso	Oximetría de pulso	Frecuencia de pulso
Dispositivo interno	Maxim Integrate MAX30102	Maxim Integrate MAX30100	FR-4
Dimenciones	12.7 x 12.7 mm	23.5 x 19 mm	33 x 25 mm
Voltaje de Operación	3.1 – 5.25 V	5 V	5 V
Consumo energético	600 µA	600 µA	20 mA
Temperatura de Operación	-40ºC a 85ºC	-40ºC a 85ºC	-30ºC a 70ºC
Aditamentos necesarios	No	No	No
Wearable	Si	Si	Adaptable
Lavable	Adaptable	No	No
Precio	10 USD	30 USD	2.12 USD

Elaborado por: Investigador

Se ha optado por el uso del sensor MAX30102, siendo su principal característica su voltaje, temperatura de funcionamiento, comunicación la cuál es a través de una

interfaz estándar compatible con I2C. El módulo se puede apagar a través de un software con cero corriente de espera, lo que permite que los rieles de alimentación permanezcan conectados en todo momento. Además no causa ninguna molestia al usuario por lo que representa una alternativa rentable.

4.3.3. Dispositivos de Comunicación

En cuanto a la comunicación se requiere de un dispositivo estable y compatible con el hardware y software a utilizarse y represente una herramienta ideal para las comunicaciones entre el sistema y la interfaz de visualización. Dichos parámetros se muestran en la Tabla 4.3.

Tabla 4.3. Análisis comparativo de dispositivos de comunicación [52, 53, 54].

Módulo de Comunicación Bluettoth			
Parámetros técnicos	Módulo Flora Bluefruit LE	Módulo Bluetooth HC-05	Módulo BluetoothHM-10
			
Dispositivo Interno	Módulo MDBT40	CSR-BC417143	CC2541F256
Protocolo	Bluetooth V4 BLE	Bluetooth 2.0 + EDR	Bluetooth 4.0
Compatibilidad con dispositivos móviles	Android 4.0 o superiores	Todas las distribuciones de android	Todas las distribuciones de android
Tasa de Baudios por defecto	9600 baud	9600 baud	9600 baud
Protocolos de comunicación	Beacon/UART	UART	UART
Velocidad Comunicación serial	9600 bps	1200 – 1382400 bps	1200 – 1382400 bps
Voltaje de operación	2.7 a 3.6 V	3.1 a 4.7 V	3.3 a 5 V
Consumo Corriente	13.4 mA	30-40 mA	8.5 mA
Frecuencias de trabajo	2.4 – 2.483 GHz	2.4 – 2.48 GHz	2.4 GHz
Temperatura de operación	-25ºC a 75ºC	-25ºC a 75ºC	-25ºC a 75ºC
Antena	Interna	Interna	Interna
Precio	17.50 USD	7 USD	8 USD

Elaborado por: Investigador

Para la comunicación se optó por el módulo de bluetooth HC-05, debido a que ofrece una mejor relación de precio y características, ya que es un modulo Maestro-Esclavo, quiere decir que además de recibir conexiones desde una PC o tablet, también es capaz de generar conexiones hacia otros dispositivos bluetooth. Esto nos permite por ejemplo, conectar dos módulos de bluetooth y formar una conexión punto a punto para transmitir datos entre dos microcontroladores o dispositivos.

4.3.4. Batería

Para la selección de la fuente de alimentación para abastecer a los dispositivos de sistema de telemedicina deberán cumplir varios parámetros los cuales son; el tamaño reducido, ser recargable, y mayor durabilidad, que brinde seguridad y el mayor tiempo de conexión entre la indumentaria y el dispositivo de monitoreo; descrita en la Tabla 4.4.

Tabla 4.4. Análisis comparativo de Baterías [55, 56, 57].

Baterias			
Parametros técnicos	Ni-MH	LIPO	LI-ION
			
Composición	Níquel-Hidruro Metálico	Polímero de Litio	Iones de Litio
Voltaje	3.7 V	3.7V	3.7V
Peso	567 g	27.5 g	40 g
Tamaño	9.5mm x 3.8mm x 7.6mm	9mm x 30mm x 54mm	66mm x 46mm x 3mm
Pin de monitoreo de carga	No	No	Si
Pin de monitoreo de temperatura	No	No	Si
Riesgo de inflamación	Casi Nula	Si	Casi Nula

Elaborado por: Investigador

Se ha optado por utilizar las baterías de LI-ION gracias a su posibilidad de carga, accesibilidad y monitoreo frente a las de LI-PO ya que se vuelven inestables al pasar el tiempo y si exceden sus valores de carga tienden a explotar, en cambio las Ni-MH tienen mayor ventaja sobre las LI-ION por su facilidad de acceso, estabilidad, durabilidad y resistencia ante la descarga en cuanto a su trabajo, pero su mayor debilidad es su tamaño y peso.

4.4. Ubicación del sensor

Uno de los aspectos importantes a considerar es la ubicación de los sensores dentro del sistema de telemedicina, ya que de esto dependerá la calidad de los datos obtenidos.

Existen diferentes zonas corporales que permiten una adecuada medición de la saturación de oxígeno y los pulsos cardiacos, puesto que son relativamente translúcidas y tienen un buen flujo sanguíneo, por ejemplo los dedos de la mano, del pie y el lóbulo de la oreja; para una mejor toma de la información y comodidad para el usuario se optó por ubicarla en el dedo de la mano, presentada en la Figura 4.2.

Figura 4.2. Localización del sensor de pulsioximetría.

Elaborado por: Investigador

4.5. Requerimientos técnicos

Según lo descrito en esquema general del sistema de monitoreo Figura 4.1, se deriva el esquema de la indumentaria para el prototipo como lo muestra la Figura 4.3.

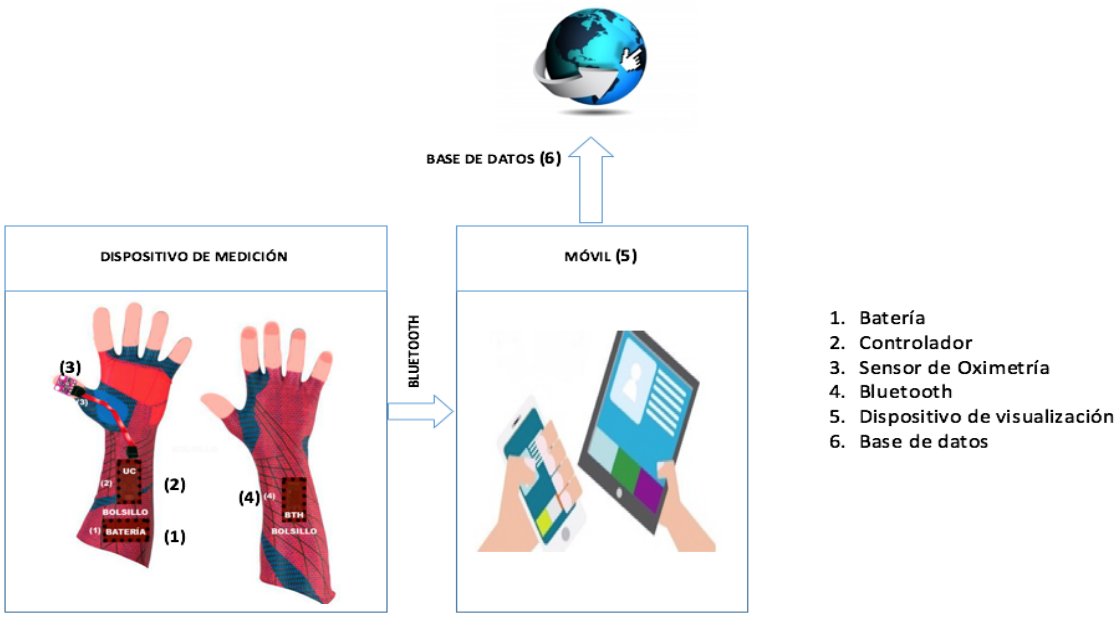


Figura 4.3. Esquema de la indumentaria para el monitoreo continuo de constantes vitales.

Elaborado por: Investigador

Descripción del esquema de la indumentaria:

1. Fuente de energía: desempeña un papel fundamental puesto que es el proveedor de energía necesaria para poner en marcha todos los dispositivos electrónicos durante un tiempo considerable.

2. Controlador: es el dispositivo principal, el cual es el responsable del procesamiento, toma de decisiones y envío de las constantes vitales hacia el dispositivo móvil.

3. Sensor de oximetría: dicho sensor deberá ser capaz de tomar la lectura de las constantes vitales, con un mínimo margen de error, de tamaño adecuado y poder usarse de manera indefinida sin que las lecturas aumente su error.

4. Dispositivo de comunicación: dispositivo principal después del controlador central, el cual permite la comunicación del guante con el dispositivo de visualización, el cual deberá ser de bajo consumo energético, compatible con el controlador central y de fácil administración

5. Dispositivo de visualización: dispositivo que soporte la comunicación inalámbrica y que permita el envío de señales de emergencia en caso de ser necesario

6. Base de datos: lugar donde se van a alojar la información de las constantes vitales y van a proveer un registro al dispositivo de visualización.

4.6. Adquisición y acondicionamiento de señales

El sistema principal del dispositivo de monitoreo depende la confiabilidad de las señales, que deben ser tratadas con cuidado y responsabilidad, para que el sistemas sea una herramienta de control y prevención del SMSL.

Seguidamente se describe las señales y el tipo de acondicionamiento de los componentes tanto de hardware como de software.

- **Señal de Pulsioximetría**

Dado el índice relativamente bajo de cambio del flujo sanguíneo, un diseño de oxímetro de pulso típico puede operar a tasas de adquisición correspondientemente bajas. Un diseño típico impulsará los LED rojo e IR (Infrarrojo) alternadamente a una frecuencia baja y un ciclo de trabajo mínimo para minimizar el consumo de energía. La operación en un ciclo de trabajo bajo también permite la medición de la luz de fondo de la luz ambiental mientras ambos LED están apagados. Sin embargo, al manejar los LED, debe minimizar el ruido al tiempo que entrega pulsos de corriente a niveles precisos apropiados para cada tipo de LED. Además, el tiempo entre los pulsos de corriente LED rojo e IR debe controlarse cuidadosamente para mantener la integridad de la medición. Los diseños capaces de cumplir estos requisitos pueden ser relativamente complejos.

La medición de la frecuencia cardíaca aprovecha los cambios en la absorción de la luz asociados con el flujo sanguíneo y la desoxigenación de la hemoglobina. El enfoque

más simple, la fotopletismografía, aprovecha los cambios en la absorción de la luz en la piel debido al incremento en el volumen de sangre local causado por la breve expansión de los vasos sanguíneos locales a medida que cada latido del corazón empuja la sangre a través de ellos. Un enfoque más confiable, la oximetría de pulso, aprovecha la absorción diferencial de luz que se encuentra en la hemoglobina en sus estados oxigenado y desoxigenado.

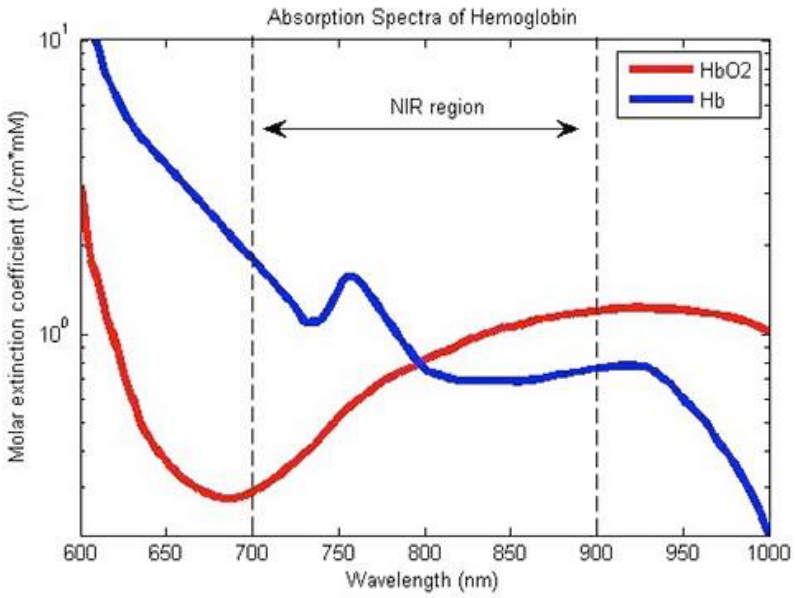


Figura 4.4. Absorción de energía diferencial de la hemoglobina desoxigenada (Hb) y la hemoglobina oxigenada (HbO2) en las longitudes de onda roja e infrarroja [58]

Como se muestra en la Figura 4.4, la hemoglobina desoxigenada (Hb) absorbe más luz roja que la luz infrarroja, mientras que la hemoglobina oxigenada (HbO2) absorbe más luz infrarroja que la luz roja. Los oxímetros de pulso comparan las lecturas de absorción de la desoxihemoglobina a alrededor de 660 nm y la oxihemoglobina a alrededor de 880 nm para calcular la SpO2. Este enfoque de medición diferencial produce datos de frecuencia cardíaca más limpios que la fotopletismografía porque es menos sensible al movimiento, ruido u otros artefactos de medición. Aunque la pulsioximetría es simple en concepto, la implementación presenta desafíos múltiples e interesantes.

Para cada LED se utilizan sub-circuitos de controladores de corriente separados, cada uno compuesto por un convertidor digital a analógico (DAC) para control de corriente,

un escenario de filtro para reducción de ruido y un amplificador operacional en la configuración de sentido de fuerza Figura 4.5.

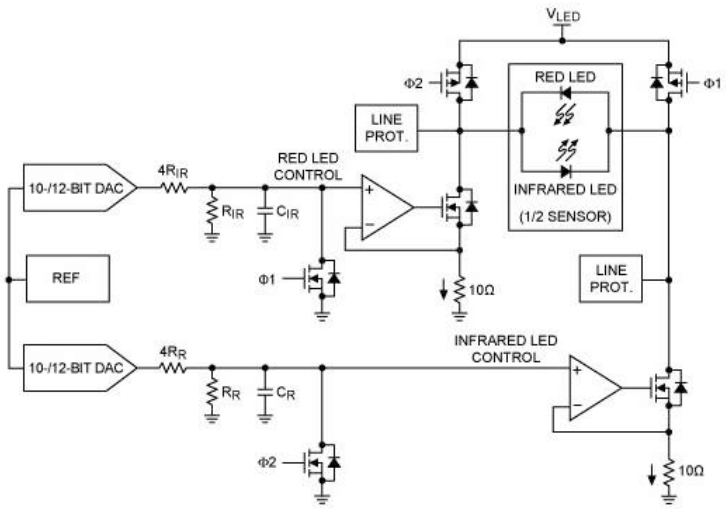


Figura 4.5. Controladores de corriente de bajo nivel de ruido para administrar los LED rojos e IR. [58]

Se utilizó un enfoque el cual realiza mediciones principalmente en el dominio analógico, utilizando un convertidor analógico digital (ADC) dedicado para cada longitud de onda, o un único ADC de alta resolución sincronizado con los controladores LED para medir los resultados en cada longitud de onda. Alternativamente, puede realizar mediciones principalmente en el dominio digital.

Este enfoque simplifica el diseño del hardware, y el intercambio es un aumento modesto en la complejidad del software. En el núcleo de cualquiera de los enfoques, un amplificador de transimpedancia (TIA) convierte la corriente de salida del fotodiodo en voltaje para la medición mediante el ADC como se muestra en la Figura 4.6.

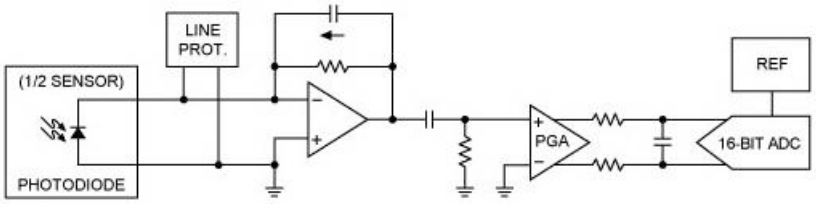


Figura 4.6. Amplificador de Transimpedancia. [58]

En un oxímetro de pulso, las señales de interés generarán una salida de corriente de fotodiodo relativamente pequeña, particularmente en comparación con la corriente generada por las fuentes de luz ambiental. Para ayudar a maximizar la SNR en este entorno, el propio TIA necesita mostrar una corriente y ruido de entrada muy bajos. Un filtro de paso alto se usa típicamente para eliminar el componente fuente ambiental de la señal. Finalmente, un amplificador de ganancia programable está configurado para usar el rango dinámico completo del ADC para una resolución de conversión de señal óptima.

Junto con el diseño cuidadoso de los circuitos de entrada de salida de fotodiodos y LED, los diseños de oxímetros de pulso requieren una colocación mecánica adecuada de los dispositivos LED en relación con el fotodiodo. De hecho, los oxímetros de pulso miden la salida de luz usando dos configuraciones físicas diferentes. Los dispositivos diseñados para conectarse a un dedo o lóbulo de la oreja miden la transmisión de luz roja e IR. En estos dispositivos, los LED y el fotodiodo se colocan en lados opuestos, típicamente contenidos en los brazos opuestos de un clip. Los LED ubicados en un brazo del clip hacen brillar la luz a través de la parte del cuerpo, mientras que un fotodiodo ubicado en el brazo opuesto lee los niveles de luz transmitida.

Por el contrario, los dispositivos diseñados para usarse en la muñeca o colocarse contra la frente del paciente se basan en la reflectancia de la luz roja e IR. En estos dispositivos, los LED y el fotodiodo están en el mismo lado del dispositivo para entrar en contacto con la piel. Los HRM de reflectancia ofrecen más flexibilidad para el usuario porque pueden colocarse contra cualquier área de piel suficientemente plana. Para el diseñador, sin embargo, este enfoque requiere una colocación cuidadosa de los LED y el fotodiodo uno con respecto al otro para optimizar la recepción de la luz roja e IR ya que cada longitud de onda se refleja en el flujo sanguíneo que pasa por debajo de la superficie de la piel.

El MAX30102 es un módulo autónomo que cumple con los requisitos electrónicos y mecánicos descritos anteriormente. Junto con la electrónica para la administración de corriente de LED de bajo ruido y la adquisición de señal de fotodiodo, incluye LED rojos e IR y un fotodiodo, todos posicionados de manera óptima para realizar mediciones de absorción de luz. Es necesario agregar solo algunos componentes

externos para implementar un subsistema de oxímetro de pulso capaz de proporcionar datos de frecuencia cardíaca a un procesador host. Figura 4.7

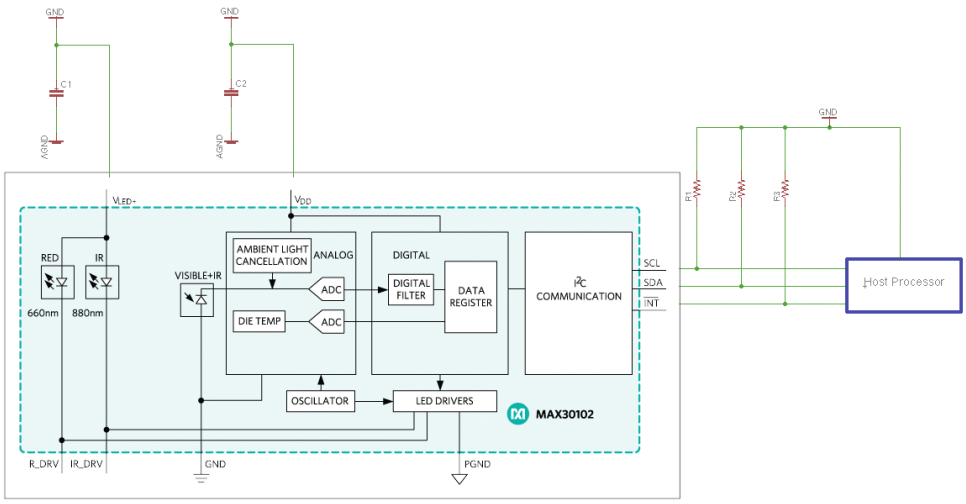


Figura 4.7. Módulo Maxim Integrado MAX30102. [58]

Gracias al ancho de pulso se optimiza la compensación entre la precisión de la medición y el consumo de energía, como se observa en la Tabla 4.5. Una función de proximidad integrada ayuda a reducir los requisitos de potencia al desactivar la emisión de luz cuando los sensores del oxímetro de pulso se retiran de la superficie de la piel del usuario.

Tabla 4.5. Caracteristicas técnicas Sensor MAX30102. [58]

Corriente controlador LED Rojo	0 – 50 mA
Corriente controlador LED IR	0 – 50 mA
Ancho de pulso	69 µs – 411 µs
Subsistema de medición de SpO2	Cancelación luz ambiente
Filtro	Tiempo discreto patentado y un ADC sigma-delta
Rango de entrada ADC	18 bits de 2 µA a 16 µA
Velocidad de muestreo	50 muestras por segundo a 3.200
Sensor de temperatura	Permite compensación de error
Modo de operación	fotopletismograma convencional (PPG) LED rojo
Modo de operación	SpO2 LED rojo como el IR

Para admitir secuencias alternas de iluminación roja e IR, el MAX30102 proporciona dos canales LED Figura 4.8. Cada canal puede emitir un pulso de 69, 118, 215 o 411 µs de duración con un tiempo preestablecido entre pulsos de 358, 407, 505 o 696 µs, respectivamente, como se muestra en la Tabla 4.5. Además, la frecuencia de muestreo establece implícitamente un límite superior en el ancho del impulso. Por ejemplo, a una velocidad de muestreo máxima de 3.200 sps, el ancho del pulso se limita a un máximo de 69 µs, lo que da como resultado el retardo entre pulsos rojos e IR de 358 µs. El ancho de pulso seleccionado también establece la resolución de ADC en 15 bits (ancho de pulso de 69 µs), 16 bits (118 µs), 17 bits (215 µs) o 19 bits (411 µs).

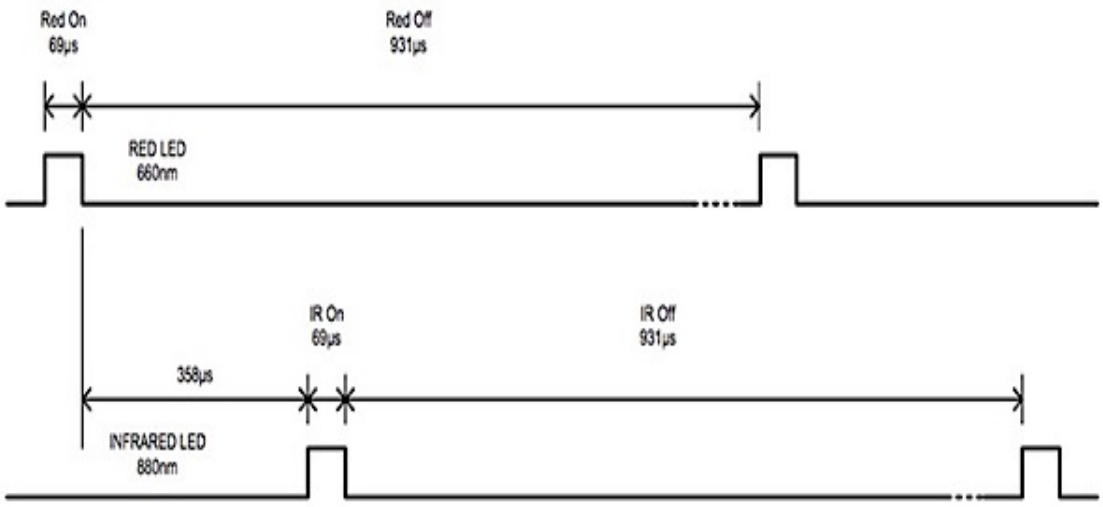


Figura 4.8. Canales separados para controlar los LED rojos e IR con un ancho de pulso. [58]

En el lado de recepción, el MAX30102 integra un búfer primero en entrar, primero en salir (FIFO) que almacena hasta 32 muestras. En consecuencia, el procesador host no necesita leer los datos de salida del sensor después de cada muestra. En cambio, el host puede adquirir periódicamente los datos de temperatura necesarios para la compensación, confiando en la FIFO mientras tanto, preservar los datos de la frecuencia cardíaca mientras el MAX30102 continúa muestreando a la velocidad preestablecida.

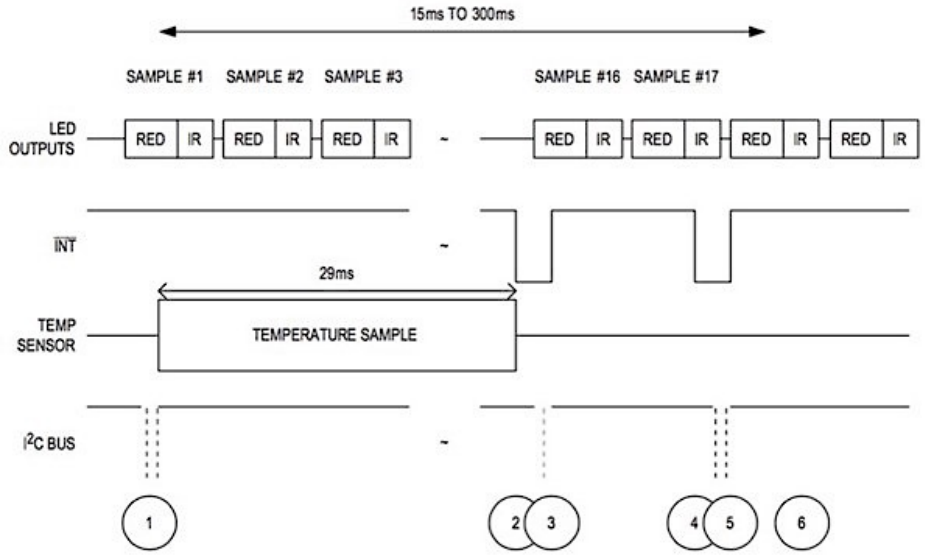


Figura 4.9. Búfer de datos MAX30102 de Maxim Integrated. [58]

Como se muestra en la Figura 4.9, una secuencia de muestreo de pulso-oximetría completa comienza con el ajuste del dispositivo en modo SpO2 escribiendo 0x03 en los bits 2: 0 en el registro de control de modo del dispositivo. A medida que avanza el muestreo, el software de muestreo puede establecer el bit TEMP_EN en el registro de configuración de temperatura del dispositivo para iniciar una medición de temperatura (evento "1"). Cuando el MAX30102 completa la medición de temperatura, activa la interrupción TEMP_RDY para alertar al host ("2"). A su vez, la interrupción se borra cuando el host lee los datos de temperatura ("3"). Cuando el búfer de muestra FIFO alcanza un umbral "casi total" (un valor establecido en el registro de configuración del dispositivo), el dispositivo emite una interrupción ("4"), indicando al host que lea el FIFO, lo que borra automáticamente la interrupción ("5 ") Y ajusta el puntero de lectura FIFO a la ubicación de la siguiente muestra nueva (" 6 "). Debido a que los punteros FIFO se ajustan con cada lectura FIFO, este proceso puede continuar indefinidamente hasta que la aplicación termine explícitamente la medición de SpO2 como se encuentra en la Figura 4.10. y como están ligados a los Anexos C y Anexo D a los mismos.

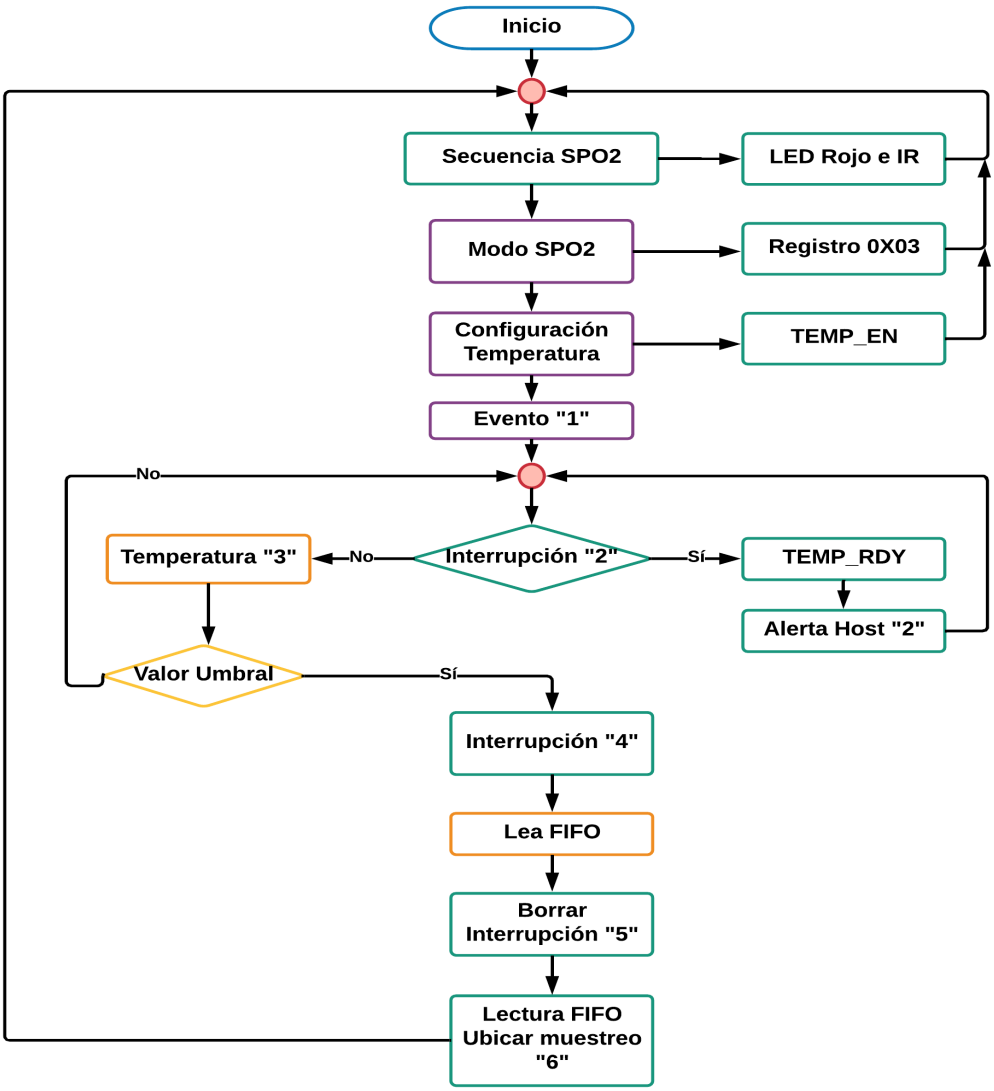


Figura 4.10. Secuencia de muestreo de Saturación de Oxígeno “SOP2”

Elaborado por: Investigador

La placa de diseño de referencia se puede usar con cualquier MCU que proporcione una interfaz I2C. Para los diseñadores que buscan un inicio rápido en el desarrollo de aplicaciones, Maxim Integrated proporciona ejemplos de bibliotecas de software para las plataformas ARM® mbed o Arduino. Al utilizar el software de muestra, el tablero de diseño de referencia consume menos de 5,5 mW y alcanza un nivel de precisión comparable al de un producto HRM con correa para el pecho como se muestra en la Figura 4.11. [45]

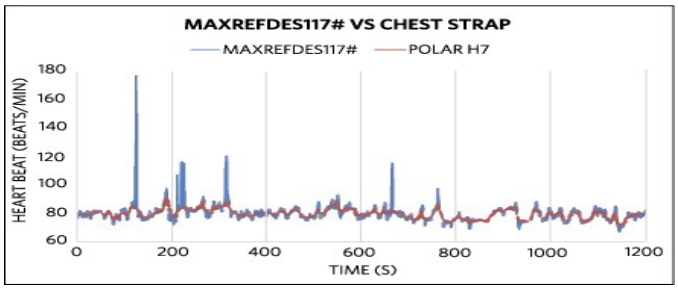


Figura 4.11. MAXREFDES117# vs correa de pecho Polar H7. [58]

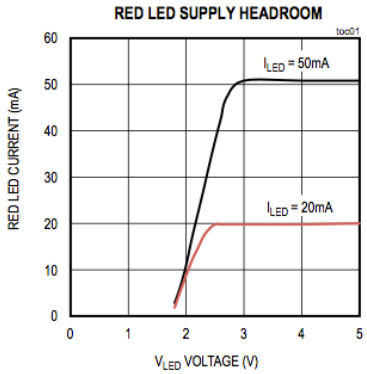


Figura 4.12. Fuente Led Rojo en Espacio libre. [58]

Como se muestra en la Figura 4.12. y Figura 4.13. observamos al LED rojo y al infrarrojo que iluminan alternativamente durante cierto tiempo (ancho de pulso) la zona expuesta (un dedo). La luz reflejada se detecta con un fotodiodo, que queda en la parte de abajo de las siguientes imágenes de detalle. La corriente de estos LED se puede configurar a un valor entre cero y 50 mA.

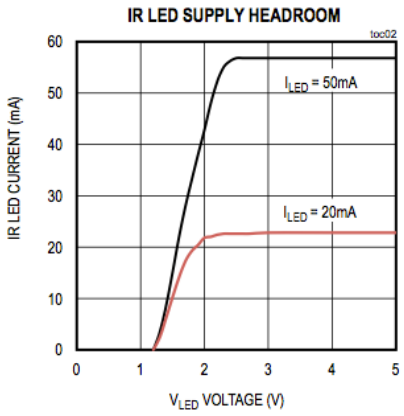


Figura 4.13. Fuente IR en Espacio libre. [58]

La nomenclatura usada al hablar de dispositivos como el MAX30102, la propia hoja de datos es un buen ejemplo, pueden inducir a error y hacer pensar que la información que entrega es el valor de la frecuencia cardíaca o del ritmo cardíaco pero la medida que ofrece MAX30102 es la intensidad de luz infrarroja y/o roja reflejada.

Utilizando el valor de esas intensidades a lo largo del tiempo es posible estimar tanto el pulso (la frecuencia cardíaca) como en nivel de oxigenación de la sangre (la relación entre la cantidad de hemoglobina y la cantidad de oxihemoglobina)

Ya se ha dicho que el MAX30102 almacena varias lecturas en una FIFO lo que permite leer varios valores ya preparados en lugar de ir cargando cada uno que se integra. Si el MCU no está muy ocupado con otras tareas también es posible leer cada valor cuando el MAX30102 informe, generando una interrupción, de que el proceso de la ADC ha terminado.

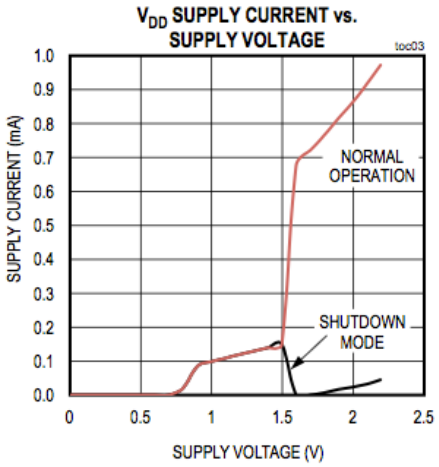


Figura 4.14. Corriente de suministro de VDD vs. Voltaje de suministro [58]

De esta manera también se puede tener una forma de modo de reposo según el voltaje y la corriente suministrada el cual nos ayuda a a ser un dispositivo de bajo consumo como se indica en la Figura 4.14.

Para el acondicionamiento de señales se utilizo la paca de arduino micro pro y la ultima versión de arduino, de esta manera se pudo obtener los valores de las señales que nos suministra el dispositivo mediante los LED rojo e IR, obteniendo un diagrama electrónico mostrado en la Figura 4.15 y un diseño PCB en la Figura 4.16.

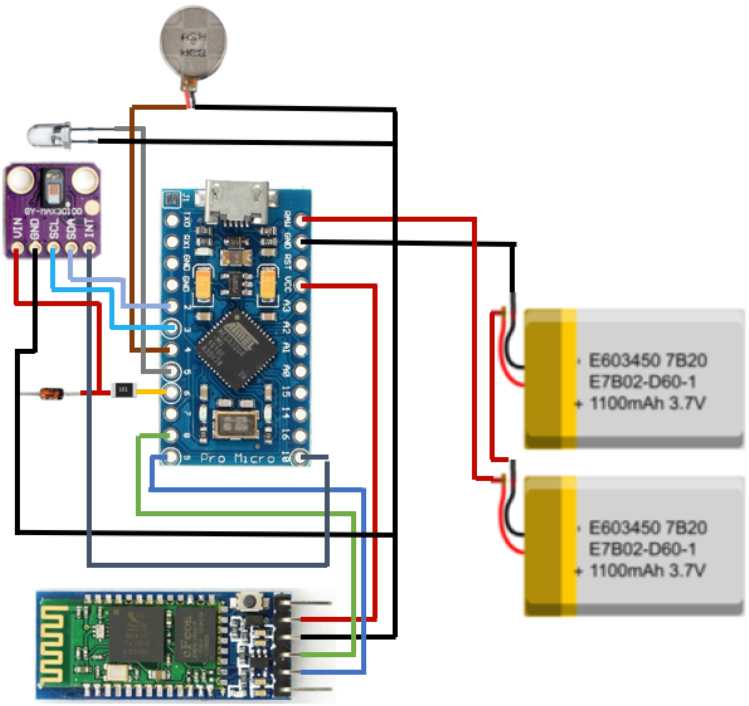


Figura 4.15. Diagrama Electrónico del prototipo

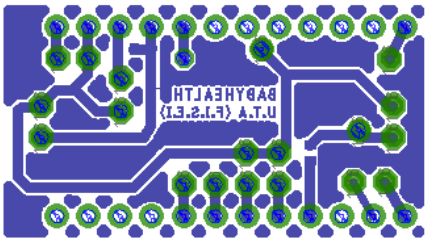


Figura 4.16. Diseño PCB para el Prototipo del Pulsioxímetro.

Elaborado por: Investigador

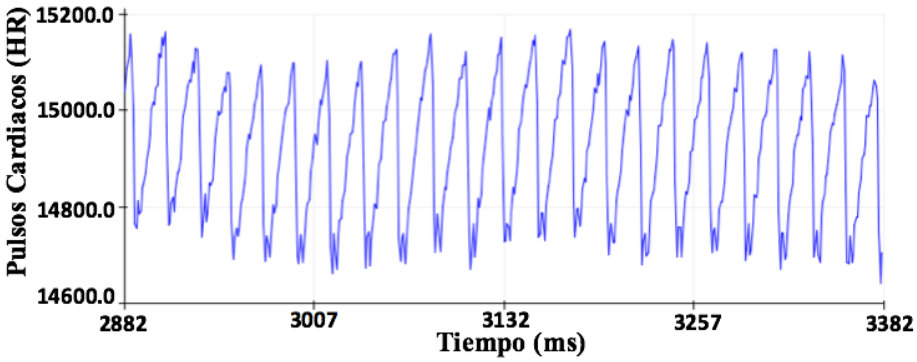


Figura 4.17. Adquisición de dados de la frecuencia cardiaca del Prototipo.

Elaborado por: Investigador

Analizando la gráfica resultante del código de ejemplo es posible, además de acotar el rango de valores útiles, buscar un método de cálculo del pulso basado en la medida del oxímetro de pulso MAX30102 de la luz infrarroja reflejada. A simple vista, lo más sencillo parece medir el tiempo entre el inicio de las crestas o los valles del gráfico, por desgracia, hay algunos inconvenientes que impiden simplemente detectar cuándo se cambia la dirección de la curva.

En primer lugar, la banda de valores máximos y mínimos es relativamente estrecha comparada con el rango de posibles valores. En segundo lugar, esta franja puede ir variando a lo largo del tiempo (lo que, en parte, puede que se minimice cuando la lectura se estabilice) dependiendo, por ejemplo, de lo cerca que se sitúe en el sensor el objeto medido (un dedo, normalmente). Por último, los valores no siguen una distribución uniforme (del valle a la cima) sino que existe, en el mejor caso, la inflexión que en un ECG correspondería con el complejo QRS y en el peor, más o menos ruido en la lectura, dependiendo también de la resolución y la frecuencia de muestreo.

La Figura 4.17 muestra la salida del monitor serie de Arduino con una frecuencia de muestreo más alta. La distribución de valores puede llegar a ser tan ruidosa que oculte los quiebros de la curva (en este caso en concreto, por la influencia de luz ambiental). Por otro lado, es interesante apreciar que siguen claramente destacados los máximos y los mínimos que permitirían calcular la frecuencia cardíaca (pulso).

Dos de las formas más económicas en recursos del MCU que pueden utilizarse consisten en: promediar los valores obtenidos hasta que solamente resulten relevantes los máximos y los mínimos y tomar unos u otros y determinar el tiempo transcurrido entre ellos y estimar cuándo los valores dejan de subir o de bajar, admitiendo una tolerancia (máximo de duración de valores en una dirección cuando se atiende a la contraria) y midiendo el tiempo entre ambos eventos. Si el rango de datos está muy bien acotado el primer método parece más sencillo pero se intuye que el segundo será más flexible si el rango de datos es variable (normalmente cuando el contexto físico es más susceptible de cambiar) por lo que es la opción que se toma como referencia.

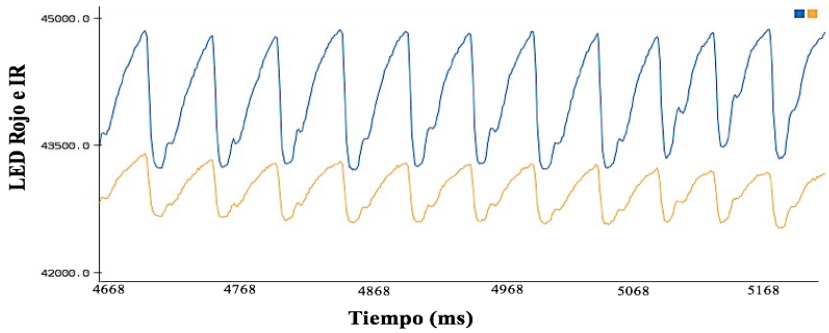


Figura 4.18. Adquisición de datos de la Saturación de Oxígeno "SPO2" vs Frecuencia Cardiaca "HR" del Prototipo.

Elaborado por: Investigador

La saturación de oxígeno en sangre se suele expresar como un porcentaje que relaciona la presencia de oxihemoglobina con la de hemoglobina. La SpO2 se puede calcular con la ecuación 1:

$$SpO_2 = {}^{100\times HbO_2}/_{(HbO_2 + Hb)} \quad (1)$$

Las medidas que el MAX30102 entrega corresponden a la luz roja e infrarroja reflejada en la zona expuesta al sensor y emitida por los correspondientes LED. Como la hemoglobina absorbe más radiación en la longitud de onda del rojo (~650 nm) y la oxihemoglobina del infrarrojo (~950 nm), se puede inferir la presencia de Hb y HbO_2 por la luz de cada tipo medida.

Supuesto que la longitud de onda de la luz emitida por cada LED sea la correcta, un inconveniente al calcular la SpO_2 podría consistir en la intensidad diferencial emitida por cada uno de los LED. Para compensarla, además de un desplazamiento y un coeficiente estimados por ensayo, sería posible suministrar diferente corriente a cada uno de los LED. Para ilustrar esta posibilidad y para poder visualizar la distribución de ambas medidas, en el siguiente código se utilizan 30600 µA para el rojo y 50000 µA para el infrarrojo de forma que en la Figura 4.18 que se obtiene con el trazador serie de Arduino puede superponer los valores medidos para ambos LED. Con pruebas de este tipo se pueden conocer mejor los márgenes en los que funciona un de sensor, el MAX30100, en este caso. Por ejemplo, se observó que hasta no llegar a los 7600 µA

no se detecta de manera estable la luz reflejada y que la luz que emite el LED rojo a más de 40200 µA satura el fotodiodo.

Calibración de la relación entre la luz roja e infrarroja.

Algoritmo:

Leer: intensidad_rojo, intensidad_infrarrojo

Calcular: $SPO2 = 100 * \frac{Intensidad\_rojo}{Intensidad\_rojo + Intensidad\_infrarrojo}$

Graficar: SPO2.

Se obtiene una lectura como se muestra en la Figura 4.19, en la que puede verse que el porcentaje está siempre muy cerca del 100 %, como cabría esperar. De hecho, al tratarse de un dispositivo usado como wearable de salud (es decir, un equipo no apto para su uso clínico), el límite inferior de lectura debería estar siempre por encima del 95 % de saturación.

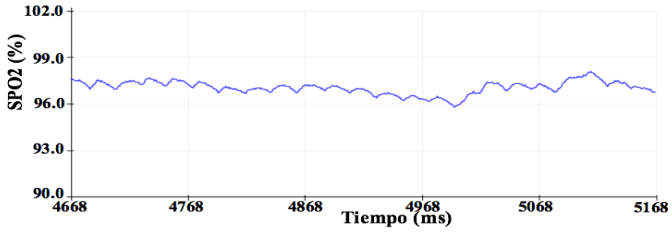


Figura 4.19. Saturación de Oxígeno "SPO2" del prototipo

Elaborado por: Investigador

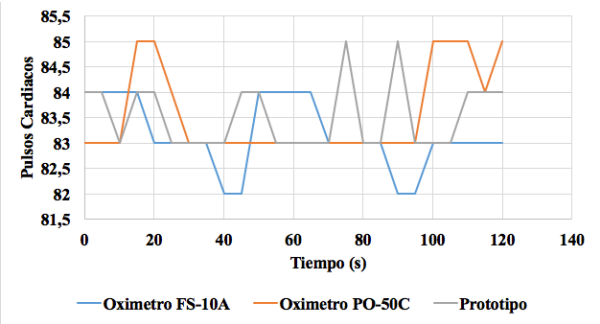


Figura 4.20. Medición de datos del prototipo de frecuencia cardíaca Vs Oxímetro de pulso PO-50 C y Oxímetro de pulso FS-10A

Elaborado por: Investigador

De lo anterior se desprende que el prototipo es una herramienta efectiva para la prevención del SMSL, ya que con el Oxímetro de Pulso PO-50C y el Oxímetro de Pulso FS-10A comparten una similitud en los valores emitidos con un error por el como se muestra en Figura 4.20 y Figura. 4.21, de esta manera emitirá alertas de una manera más eficiente si los parámetros monitoreados disminuyen los rangos establecidos, emitiendo una alerta al dispositivo móvil a través del dispositivo Bluetooth, con la ayuda del puente móvil y el la nube forma un sistema de monitoreo escalable.

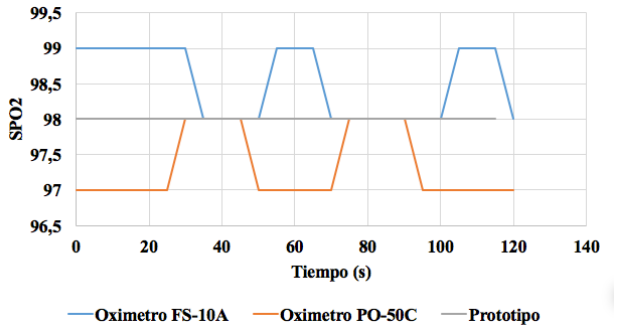


Figura 4.21. Medición de datos del prototipo de Saturación de Oxígeno “SPO2” Vs Oxímetro de pulso PO-50 C y Oxímetro de pulso FS-10A

Elaborado por: Investigador

4.7. Procesamiento y envío de señales

La etapa de procesamiento de las señales eléctricas ya acondicionadas se encuentra alojada en el controlador Arduino ProMicro, el cual es el encargado de recibir las señales de pulsioximetría variantes en el tiempo y convertirlas en valores indicadores de señales vitales. Consecutivamente se envían dichas señales en formato de cadena concatenada hacia la etapa de visualización en el cual interviene el módulo de comunicaciones bluetooth. Dicho proceso sistematizado del procesamiento y la conversión desde señales de pulsioximetría emitida por los LED Rojo e IR a valores reales de señales vitales se describe en el siguiente diagrama de flujo de la Figura 4.22 y además a través de los Anexos A, B, C y D se presentan los algoritmos ligados al mismo:

El controlador se encuentra configurado de tal manera que sea capaz de determinar entre dos subrutinas para evitar la saturación y acceder a cada una de aquellas dependiendo de comandos enviados desde el dispositivo de monitoreo.

La trama de envío representada en la Figura 4.17 y Figura 4.19 se compone de los datos de pulsos cardíacos, SPO2. Trama que es enviada de una manera muy sistemática, lo cual permite que las señales sean enviadas de manera sincronizada con los latidos del corazón del paciente.

Dicha metodología ha sido utilizada con el fin de reducir los retardos en el envío de la información y el consumo energético generado por el uso continuo del controlador central y el módulo de comunicaciones.

Subsiguientemente y utilizando el enlace Bluetooth entre la indumentaria y el visualizador, ha de desentramarse la información en la etapa de presentación de las señales. Teniendo claro que significa cada variable por la siguiente Tabla 4.6 y vinculada al Anexo A.

Tabla 4.6. Variables ocupadas para el procesamiento y envio de datos de los sensores al microcontrolador.

Serial.aviable()	Estado del sensor
ch_hr_valid	Estado pulsos cardiacos
ch_spo2_valid	Estado SPO2
cntN	Contador ubicación del dedo
n_heart_rate	Pulsos cardiacos en pulsos por minutos
n_spo2	SPO2 en porcentaje

Elaborado por: Investigador

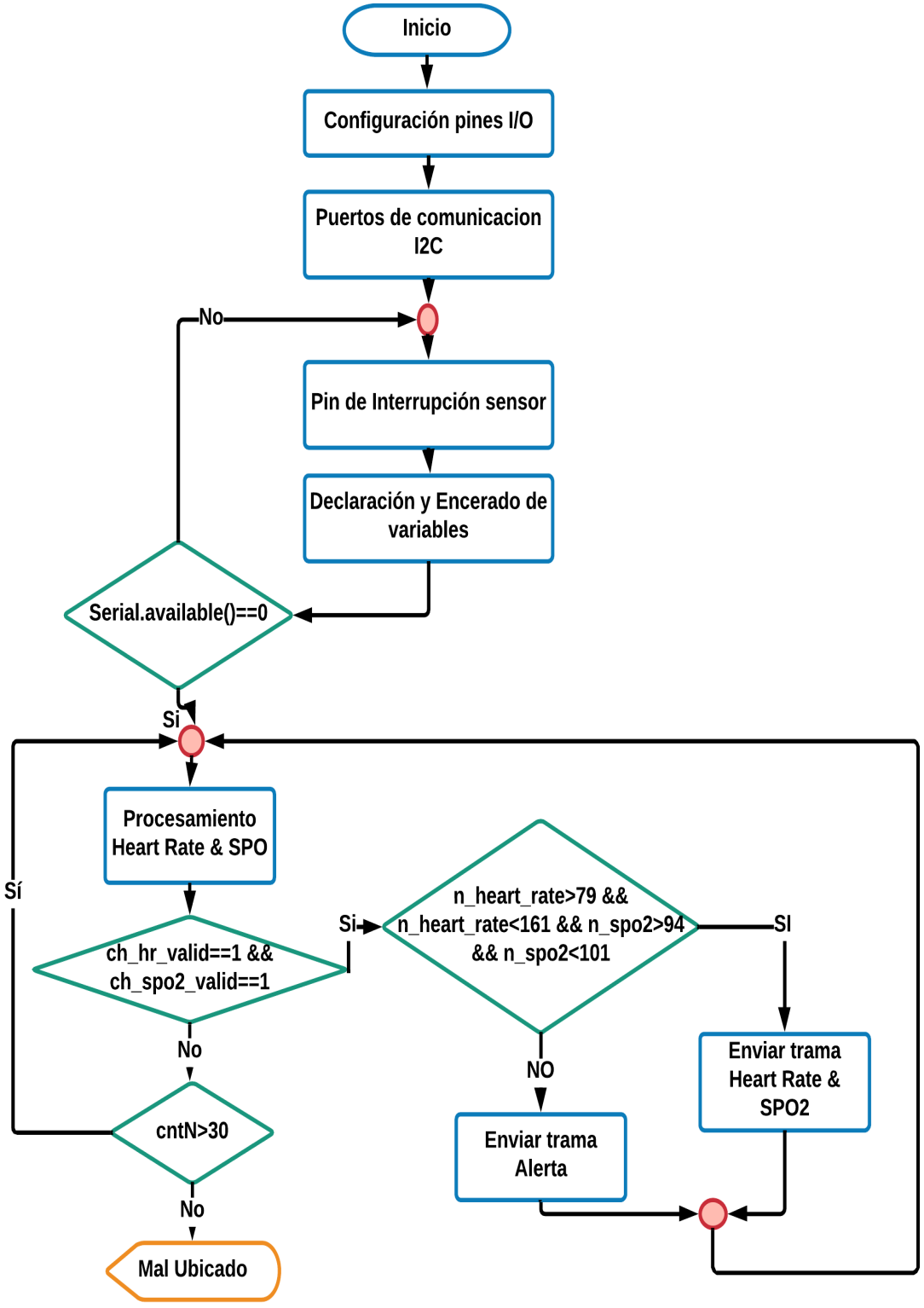


Figura 4.22. Diagrama de flujo procesamiento y envío de datos del sensor al microcontrolador

Elaborado por: Investigador

4.8. Visualización de las Señales Dispositivo Móvil

Para la exposición de los datos se utilizó un dispositivo inteligente de uso personal. Mediante el cual, la trama de datos recibidos son desempaquetadas y presentadas en una interfaz gráfica de usuario de manera amigable y simplificada que brinde un panorama mucho más amplio al usuario sobre su estado de salud en tiempo real.

Como se ha puntualizado inicialmente, la API ha de desarrollarse en lenguaje de programación JAVA Android Studio versión 2.3.3, con lo que se debe tomar en cuenta los términos y condiciones descritas en el Anexo W y ***Ministerio de Modernización*** *Dirección Nacional de Servicios Digitales* para la interfaz de ususario descrita en el Anexo X. Principalmente a la Compilación rápida, ejecución de la app en tiempo real gracias al emulador, ejecución de la app directamente desde el móvil, no soporta el desarrollo para NDK, pero intellij con el plugin Android sí, tiene renderizado en el tiempo real, layouts y puede hacer uso de parámetros tools. funciona bien (sobre todo si usas versiones estables), contiene todo lo necesario para desarrollar cualquier IDE, es capad de asociar automáticamente carpetas y archivos con su papel en la aplicación, la creación de nuevas carpetas, borrado de archivos en values y debido a que permitirá la comprensión del código implementado y su modificación en caso de ser necesario. Las operaciones de la interfaz inician una vez se haya establecido un canal de comunicación entre el dispositivo móvil y el módulo de comunicaciones; la información receptada es desempaquetada en pequeñas cadenas que contienen la información de las señales vitales para su visualización y almacenamiento en la base de datos destinada a registrar los cambios de los signos vitales del cuerpo humano y además permita obtener un historial completo de las señales en caso de presentarse una alteración de los mismos. El comportamiento de la misma esta descrito por el diagrama de la Figura 4.23, y se detalla las variables a utilizarse en la Tabla 4.7 y Anexos de la E a la N. Gracias a la interfaz de visualización amigable al usuario, el paciente se puede logear o crear una cuenta de una manera fácil para poder tener un registro mas adecuado, en la pantalla secundaria le permitirá seleccionar el dispositivo bluetooth previamente emparejado con su dispositivo móvil el cual permita acceder hacia la etapa de monitoreo del pulsos cardiacos y SPO2 o simplemente acceder hacia la tabla de la base de datos que almacena los valores de las señales vitales adquiridas

con simplemente presionar un botón en la interfaz de usuario como se puede observar en la Figura 4.24.

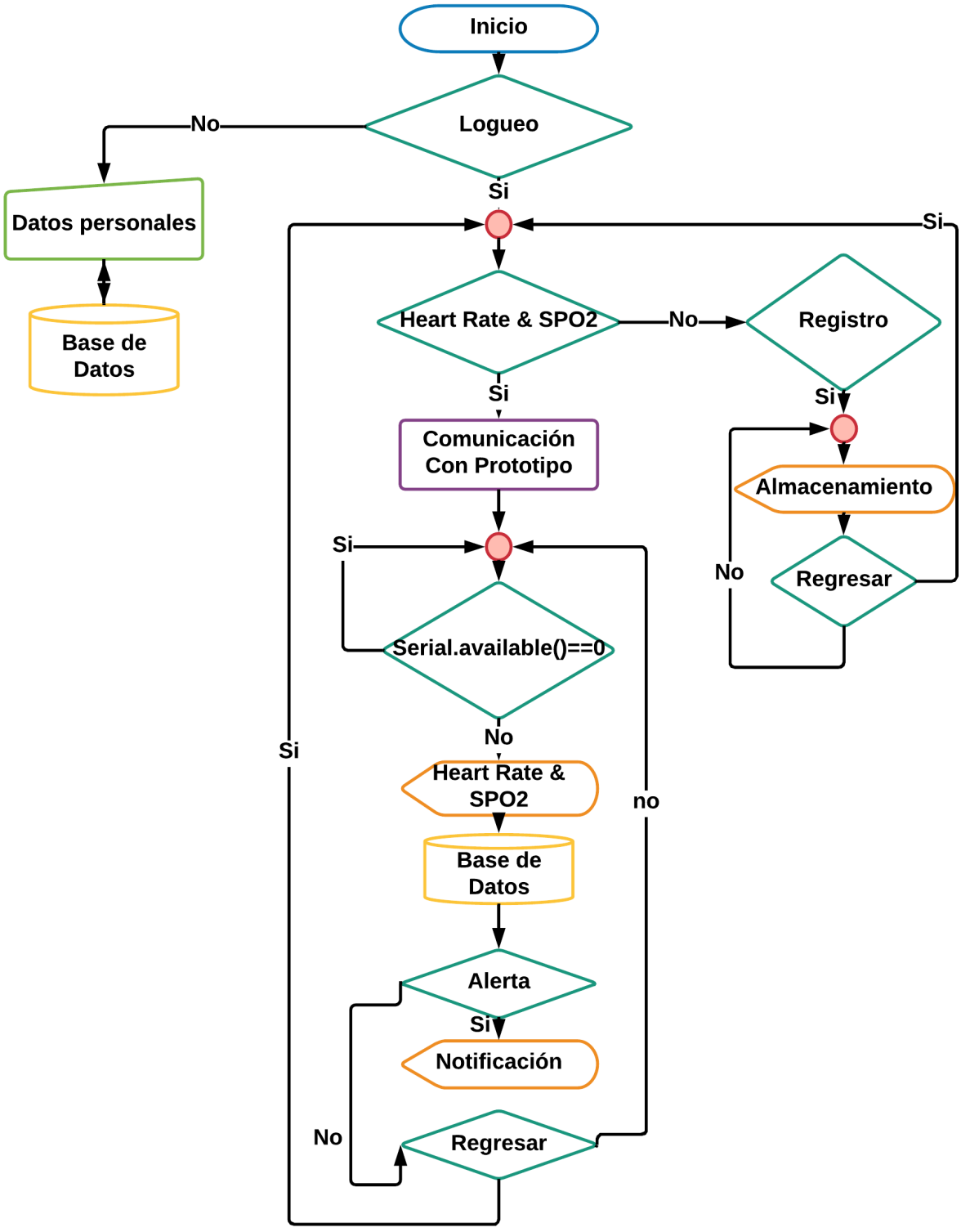


Figura 4.23. Diagrama de flujo de la etapa de visualización de datos.

Elaborado por: Investigador

Tabla 4.7. Variables ocupadas en la etapa de visualización en el dispositivo móvil.

Serial.aviable()	Estado del sensor
Heart Rate	Estado pulsos cardiacos
SPO2	Estado SPO2
Registro	Valores SPO2 y HR adquiridas
Alerta	Notificación sonora de alteraciones

Elaborado por: Investigador

La presentación de alertas resultan de las alteraciones de en las constantes vitales normales, haciendo de esta manera que el sistema salte de sus funciones normales dando de esta manera una alerta oportuna al usuario del sistema y generando un registro en la base de datos descritos desde el Anexo R hasta el Anexo U .

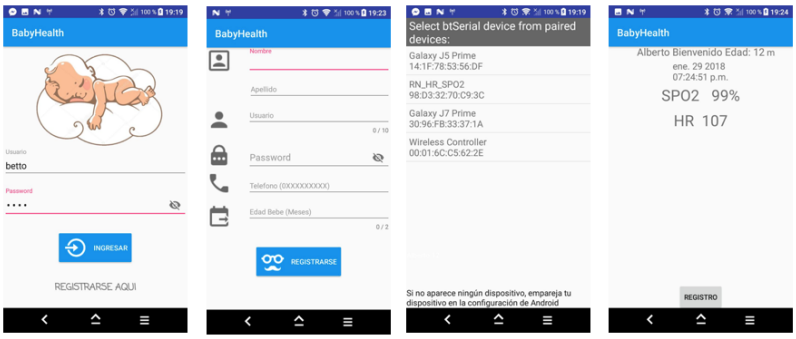


Figura 4.24. Diseño de la aplicación Móvil para la visualización de datos

Elaborado por: Investigador

4.9. Diseño del Prototipo

Para el diseño del prototipo se utilizo cables de bus de datos soldados en la placa la cual contiene al microcontrolador arduino micropro, mismos que están de una manera aislada del usuario, por medio de un empaquetamiento, siendo contenidos en capas de tela esponjada con el fin de que no sea rustico y de descomplicar la manipulación entre conexiones requeridas, como se muestra en las Figuras 4.25 y 4.26, vinculado al Anexo Q.

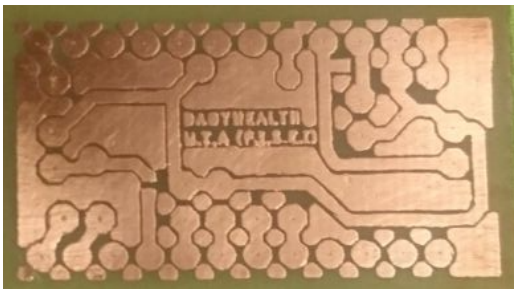


Figura 4.25. Placa del prototipo

Elaborado por: Investigador

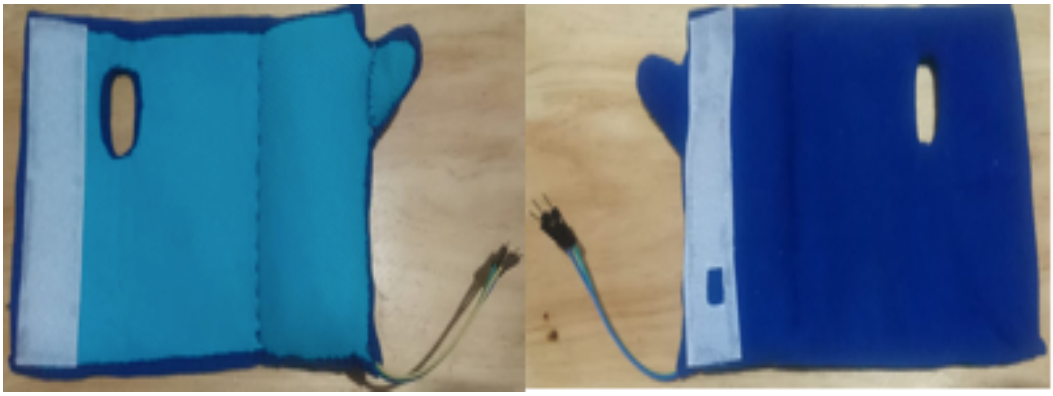


Figura 4.26. Prototipo del guante vista frontal y posterior.

Elaborado por: Investigador

Teniendo de esta manera una prenda semejante a un guante el que impida el acceso directo hacia la parte electrónica que a su vez incomodaría al usuario al contacto directo con la piel.

4.10. Pruebas de funcionamiento

Para la prueba de funcionamiento del dispositivo se ha puso a prueba la aplicación móvil y el guante a análisis de trabajo. Tomando en cuenta la aplicación como se muestra en la Figura 4.27 se comprobó la ausencia errores de conectividad o problemas con la resolución de pantalla; a su vez se ha comparado las señales obtenidas con valores proporcionados por dispositivos médicos como se presentan a en las Figura 4.28, en la Figura 4.29 y en los Anexos O y P se aprecia la forma de notificar una alteración del dispositivo al móvil.

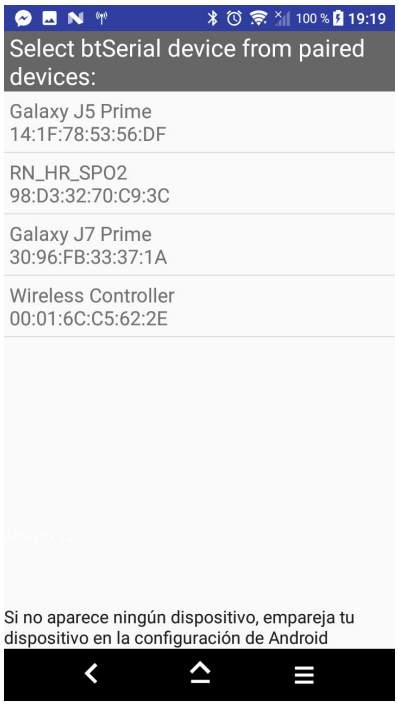


Figura 4.27. Prueba de funcionamiento de la conectividad del prototipo con el dispositivo móvil.

Elaborado por: Investigador

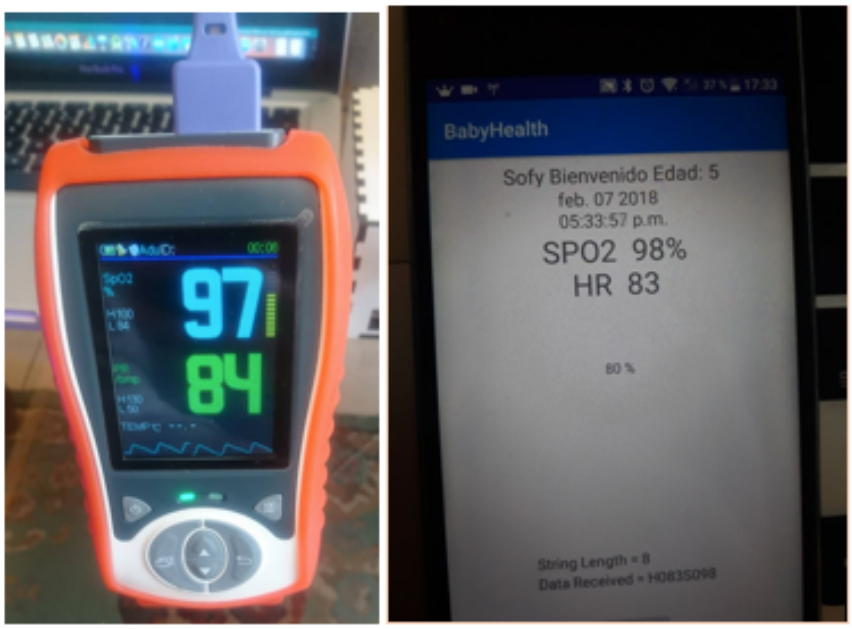


Figura 4.28. Adquisición de datos de la Saturación de Oxígeno “SPO2” y Pulsos cardiacos del Oxímetro de pulso PO-50 C vs Prototipo

Elaborado por: Investigador

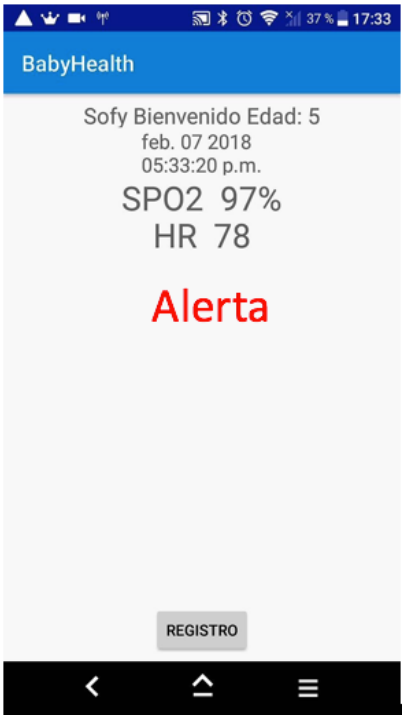


Figura 4.29. Visualización de notificaciones de alteraciones del Prototipo en el dispositivo móvil.

Elaborado por: Investigador

4.11. Análisis de resultados

Para este proceso se ha utilizado las formulas de la teoría del cálculo de errores tomado de Conceptos Básicos Cálculo de Errores de Edicta Arriagada y Victor Peralta, para determinar la funcionabilidad y el beneficio del prototipo en comparación a productos del mercado, estableciendo su error absoluto y relativo aplicando las ecuaciones (2) y (3) respectivamente.

$$Error\ Absoluto\ (Ea) = |Valor\ Real - Valor\ Indumentaria| \quad (2)$$

$$Error\ Relativo\ (Er) = \left|\frac{Error\ Absoluto\ (Ea)}{Valor\ Real}\right| * 100 \quad (3)$$

Dicho análisis puede observarse a partir de la Tabla 4.8 y Tabla 4.9

Tabla 4.8. Análisis del Error Absoluto y Error Relativo de los valores de Saturación de Oxígeno "SPO2".

SPO2 "%"		Error	
Valor Saturador	Valor Prototipo	Error Absoluto (Ea)	Error Relativo (Er)
96	96	0	0,000
97	96	1	1,031
96	96	0	0,000
98	97	1	1,020
99	99	0	0,000
96	96	0	0,000
97	97	0	0,000
98	97	1	1,020
98	98	0	0,000
97	96	1	1,031
94	94	0	0,000
96	96	0	0,000
97	98	1	1,031
97	96	1	1,031
96	98	2	2,083
95	97	2	2,105
97	97	0	0,000
98	98	0	0,000
99	99	0	0,000
97	98	1	1,031
		0,55	0,57

Elaborado por: Investigador

Para la variable de SPO2, después del análisis de los valores obtenidos en varias tomas y en diferentes pacientes, logramos determinar que los datos son similares a los resultados de equipos comerciales ya probados por lo que el prototipo tiene una alta confiabilidad teniendo que tomar en cuenta que la ubicación del sensor y las condiciones de humedad de la piel pueden provocar una variabilidad del 3%. Se pudo establecer un error absoluto primedio de 0,55 en % y una confiabilidad de 99,43 %.

Tabla 4.9. Análisis del Error Absoluto y Error Relativo de los valores de los Pulsos Cardiacos.

Pulsos Cardiacos "ppm"		Error	
Valor Saturador	Valor Prototipo	Error Absoluto (Ea)	Error Relativo (Er)
80	81	1	1,250
82	82	0	0,000
84	84	0	0,000
79	80	1	1,266
88	88	0	0,000
90	89	1	1,111
88	87	1	1,136
87	86	1	1,149
90	91	1	1,111
87	85	2	2,299
78	78	0	0,000
81	81	0	0,000
84	84	0	0,000
85	85	0	0,000
88	88	0	0,000
87	87	0	0,000
86	86	0	0,000
79	80	1	1,266
89	88	1	1,124
88	88	0	0,000
		0,8	0,932

Elaborado por: Investigador

Para la variable de pulsos cardiacos, después del análisis de los valores obtenidos en varias tomas y en diferentes pacientes, logramos determinar los datos son similares a los resultados de equipos comerciales ya probados por lo que el prototipo tiene una alta confiabilidad 99,1 % y un error absoluto promedio de 0,8 pulsos por minuto.

En la parte de la alimentación electrica del dispositivo se realizó un análisis en cuanto a la durabilidad. Teniendo en cuenta que la fuente de energía se encuentra compuesta de dos baterías o celdas de 3.7 voltios y 1900mAh y que el módulo bluetooth tiene un consumo energético máximo de 40mA, mediante la ecuación (4) de Digikey Electronics.

$$Vida\ Útil\ Batería = \frac{CapacidadBateria\ (mA)}{Icarga\ (mA)} * 0{,}7\ (4)$$

Siendo el valor de 0.70 un factor de tolerancia ante factores ajenos al dispositivo que puedan afectar la vida útil de la fuente energética y con una corriente de consumo de 90 mA como se muestra en la figura 4.30, la vida útil de las baterías esta dada por la el resultado de la ecuación (5):

$$Vida\ Útil\ Batería = \frac{1900\ (mA)}{90\ (mA)} * 0{,}7\ (5)$$

$$Vida\ Útil\ Batería = 14{,}77\ horas$$

Se obtiene como resultado que la fuente de energía del prototipo tendría una vida útil de alrededor de 14 horas. Considerando que el usuario mantenga ambas celdas cargadas, el monitoreo de signos vitales podrá extenderse al 60% del día aproximadamente.

Figura 4.30. Consumo de Corriente del prototipo en funcionamiento.

Elaborado por: Investigador

4.12. Análisis Económico del Proyecto

Dentro del análisis económico se considera el costo de hardware y el tiempo de diseño del prototipo alcanzando un precio aproximado de $285.81, detallado a continuación.

Cabe recalcar que el software empleado en el diseño es de acceso libre y gratuito por lo que no genera costes.

4.12.1. Costo Hardware

En la Tabla 4.10 se detalla el costo de los dispositivos y componentes que se usaron en el sistema electrónico.

Tabla 4.10. Costo de Hardware.

Elemento	Cantidad	Valor Unitario	Valor total
Arduino Pro Micro	1	$ 11,50	$ 11,50
Max30102	1	$ 10	$ 10
Bluetooth HC-05	1	$ 11,50	$ 11,50
Kit Batería LI-ION	1	$ 25	$ 25
Diodo Led	2	$ 0,10	$ 0,20
Diodo 1N4728A	2	$ 0,10	$ 0,20
Resistencia smd	2	$ 0,10	$ 0,20
Motor vibrador Nokia	1	$ 5,00	$ 5,00
Cable jumper Hembra – Macho	1	$ 0,50	$ 0,50
Cable bus de datos	1 m	$ 1,00	$ 1,00
Baquelita	1	$ 4,00	$ 4,00
Tela Diamante	1m	$ 2,00	$ 2,00
Tela licra	1m	$ 4,00	$ 4,00
		TOTAL	$ 75,10

Elaborado por: Investigador

4.12.2. Costo de diseño

Para la elaboración del diseño se tomó en cuenta el total de número de horas trabajadas y el valor del salario básico de un Ingeniero Electrónico y Comunicaciones, establecido por el Ministerio del Trabajo, mostrado en la Tabla 4.11.

Tabla 4.11. Sueldo básico y mensual de un Ingeniero Electrónico y Comunicaciones

Mensual	**$ 885**
Diario	$ 42,14
Hora	$ 5,26

Elaborado por: Investigador

Se estimo un total de 40 horas de investigación distribuidas en el diseño, programación y pruebas de funcionamiento.

$$Costo\ inicial\ del\ diseño = Total\ horas\ trabajadas * valor\ de\ hora \quad (6)$$

$$Costo\ inicial\ del\ diseño = 40 * \$\ 5{,}26 \quad (7)$$

$$Costo\ inicial\ del\ diseño = \$\ 210{,}71$$

4.12.3. Costo Total del Sistema

Para la representación del costo total del sistema de monitoreo, se puede observar en la Tabla 4.12

Tabla 4.12. Costo total del prototipo

Descripción	**Valor**
Costo de Hardware	$ 75,10
Costo de diseño	$ 210,71
Total	$ 285,81

Elaborado por: Investigador

El valor inicial para la implementación del prototipo es de $ 285,81, dicho valor es bajo en comparación a los dispositivos comerciales como el Pulsioxímetro de dedo Nonin con Bluetooth 4.0 el cual tiene un valor de 480,99 €, que en el mercado internacional es el que mas se acerca a las características propuestas en este proyecto sin cumplirlas en su totalidad lo que refleja que este prototipo puede ser altamente comercializable.

Capítulo V Conclusiones y Recomendaciones

Conclusiones

- Se desarrollo un prototipo de monitoreo continuo de constantes vitales para la prevención de muerte súbita de lactantes menores de 0 a 6 meses, obteniéndose adecuados resultados en la aplicación de campo en comparación con equipos médicos certificados, con una confiabilidad del 99,43 % y absoluto 0,55 en %SPO2 y en frecuencia cardiaca una confiabilidad 99,1 % y un error absoluto del 0,8 pulsos por minutos lo que se traduce en una alta efectividad y eficiencia en el desarrollo del mismo, lo que permitirá reducir los casos de muerte súbita y en parte identificar las causas del mismo.
- Conjuntamente con el prototipo se elaboró una aplicación móvil mediante Android Studio, que permite la visualización del monitoreo en tiempo real de la HR y SPO2, además de generar notificaciones de alteraciones de las constantes vitales de acuerdo a rangos preestablecidos catalogados dentro de la normalidad; se emplea notificaciones auditivas y vibratorias en el dispositivo móvil las cuales son generadas por el prototipo y enviadas vía dispositivo bluetooth, conjuntamente de la activación del motor vibrador ubicado estratégicamente dentro del prototipo al nivel de la palma de la mano, cuyo propósito es despertar al lactante.
- El prototipo demostró tener un adecuado desempeño en la transmisión bluetooth, puesto que la comunicación del dispositivo móvil y el prototipo fue estable, garantizando el monitoreo continuo.
- Se ideo un equipo ergonómicamente adaptable a la anatomía del lactante menor así como electrónicamente segura puesto que el material electrónico nunca tiene contacto directo con la superficie corporal del mismo, lo que lo convierte en un dispositivo de fácil utilización domiciliaria.

Recomendaciones

- Se recomienda el uso de este dispositivo de telemedicina en todos los pacientes pediátricos con factores de riesgo y aquellos con historial familiar de muerte súbita.
- En el uso de dispositivos ópticos en el caso de los pulsos cardiacos y la saturación de oxigeno se recomienda localizar el sensor en lugares que tenga un buen flujo sanguíneo, por ejemplo los dedos de la mano o del pie o el lóbulo de la oreja, en este caso es ha especificado la ubicación de este dispositivo en el dedo pulgar del lactante, ya que de ello dependerá la posibilidad de adquisición de las señales.
- Al acoplar el sistema de telemedicina a un paciente pediátrico de riesgo es fundamental que los cuidadores entiendan el funcionamiento del dispositivo, sus alcances y limitaciones para conseguir funcionabilidad hasta de un 100%.
- Es recomendable ocupar material hipo-alergénico para que los lactantes no generen rechazo al mismo y sea un equipo utilizable para el monitoreo de constantes vitales propósito con el cual fue elaborado.

Bibliografía

[1] Emely Licet Morales Rúa, MD, Carolina Alexandra Zambrano Pérez, María Luisa Latorre Castro, MD, PhD en Salud Pública. Síndrome infantil de muerte súbita: revisión y ampliación de recomendaciones. CCAP, vol. 14, pp. 6-20, 2013.

[2] Dr. Pablo Riofrío Pediatra. Eventos de aparente amenaza a la vida. Universidad Internacional del Ecuador. UIDE, pp. 70-73, Ene. 2017.

[3] Dra. M. Isabel Izquierdo Macián, Dra. Beatriz Aguilera Tapia, Cristina Cáceres Marzal. Libro Blanco de la Muerte Súbita Infantil. 3ª edición. Madrid. 2003

[4] F. J. Sánchez Ruiz-Cabello, L. C. Ortiz González y Grupo PrevInfad/PAPPS Infancia y Adolescencia. "Síndrome de la muerte súbita del lactante (parte 1). Factores de riesgo". Scielo. Rev Pediatr Aten Primaria vol.15 no.60 Madrid oct./dic. 2013 [Online]. Disponible en: http://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S1139-76322013000500017

[5] Indexmundi. (2014) "Tasa de mortalidad infantil (muertes/1000 nacimientos normales)". Accedido may 2017. [Online]. Disponible en: http://www.indexmundi.com/g/g.aspx?c=ec&v=29&l=es

[6] OMS. (2012) Salud en las Américas. [Online]. Disponible en: http://www.paho.org/salud-en-las-americas-2012/index.php?id=40:ecuador&option=com_content

[7] Jhon Usiña, Darwin Céspedes y Julio Yunga. INEC: Anuario de Estadísticas Vitales - Nacimientos y Defunciones 2014. Ecuador, 2014.

[8] Jacob Colvin. "SIDS Prevention". AAMI, SPRING, pp. 6-7. 2013.

[9] Vijay K. Varadan and Linfeng Chen. “Mobile Wearable Nano-Bio Health Monitoring Systems with Smartphones as Base Stations”. ASME, 3 Park Avenue, New York, NY 10016, USA, pp. 1-7. 2012.

[10] Mrudula Borkar, Neha Kenkre, Harshada Patke Ankita Gupta . “An Innovative Approach for Infant Monitoring System using Pulse Rate and Oxygen Level”. International Journal of Computer Applications (0975 – 8887), vol. 160 – No 5,feb. 2017

[11] André G. Ferreira, Duarte Fernandes, Sérgio Branco, João L. Monteiro, Jorge Cabral, André P. Catarino, Ana M. Rocha. “A Smart Wearable System for Sudden Infant Death Syndrome Monitoring”. IEEE, pp. 1920-1925. 2016.

[12] Sakshi Gupta, Zarnain M. Khan, Rupali Srivastava, P.A. Chougule. “Infant Monitoring System Using Multiple Sensors”. IJRET: International Journal of Research in Engineering and Technology eISSN: 2319-1163 | pISSN: 2321-7308, Volume: 05 Issue: 05 may-2016.

[13] Zhihua Zhu, Tao Liu, Guangyi Li, Tong Li y Yoshio Inoue. “Wearable Sensor Systems for Infants”. SENSOR, ISSN 1424-8220, pp. 3721-3749, feb 2015.

[14] Medico Pedia, “Diccionario Médico Pediátrico”. Accedido 28 de Abril 2017 [Online]. Disponible en: http://www.portalesmedicos.com/diccionario_medico/index.php/Lactante.

[15] Emely Licet Morales Rúa, Carolina Alexandra Zambrano Pérez, María Luisa Latorre Castro. “Síndrome Infantil De Muerte Súbita: Revisión Y Ampliación De Recomendaciones”. CCAP vol. 14 Número 3.

[16] Remigio R. Gorrita Pérez, Joaquín Román Lafont. “Síndrome De Muerte Súbita Del Lactante: Un Tema Para La Polémica”. Revista de Ciencias Médicas La Habana 2013.

[17] E.U. Angela Aguayo P, E.U. Ana Paulina Lagos T. “Guia Clinica De Control De Signos Vitales”. Accedido 30 Abril 2017 [Online]. Disponible en: http://academico.upv.cl/doctos/KINE-4068/%7B328B1B37-2C2A-4747-8B38-

169806A27753%7D/2012/S1/GUIA%20TECNICA%20DE%20CONTROL%20DE%20SIGNOS%20VITALES%20KINE.pdf.

[18] Darío Cobo, Paola Daza. “Signos Vitales En Pediatría”. Revista Gastrohnup Vol. 13 Número 1 Suplemento 1: S58-S70, pp. 58-70, 2011.

[19] Guillermo Pérez. “Saturación de Oxígeno en Sangre”. Accedido 01 Mayo 2017 [Online]. Disponible en: http://www.gasometria.com/saturacion_de_oxigeno_en_sangre.

[20] Organización Mundial de la Salud. “Manual de Oximetría de Pulso Global”. Accedido 01 de Mayo 2017 [Online]. Disponible en: http://www.lifebox.org/wp-content/uploads/2012/11/WHO-Pulse-Oximetry-Training-Manual-Final-Spanish.pdf.

[21] Freddy Pomares Herrera, Francisco Fernández Periche, “SISTEMA DE TELEMEDICINA UdC: Un nuevo paradigma en la atención médica colombiana para el sur de Bolívar”. Informática y Sistemas. UNIVERSIDAD TÉCNICA DE MANABÍ Vol.1 N.1 Año (2017).

[22] Pedro Galván, Miguel Velazquez, Gualberto Benitez, Antonio Barrios, Enrique Hilario. “Perspectivas de un Sistema de Telemedicina en la Salud Pública del Paraguay. Estudio Piloto”. Rev. Salud Pública Parag. 2014; Vol. 4 N° 2; Julio – Diciembre 2014.

[23] Siemens Healtineers, 2011. "Telemedicina". Accedido 25 Abril 2017 [Online]. Disponible en http://www.elhospital.com/temas/Que-es-la-telemedicina+8082249.

[24] SCIELO, 2010. "Telemedicina: ¿futuro o presente?". Accedido 25 Abril 2017 [Online]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1729-519X2010000100017.

[25] RAE, "Diccionario de la rae", 2017. Accedido 25 de Abril 2017 [Online]. Diesponible en: http://dle.rae.es/?id=PehHKV2.

[26] Nuria Puentes Sallago, Sara Farrouh Alés, Cornelia Iibañez Vidal. Las Constantes Vitales Procedimientos Básicos de Enfermería. España: ISBN.

[27] HealthStudio. “Signos Vitales”. Accedido 01 de Mayo 2017 [Online]. Disponible en: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0ahUKEwjzhe2thdLTAhXKZiYKHUObAyMQFggxMAI&url=http%3A%2F%2Fkwaas.org%2Fhealthstudio%2Fpluginfile.php%2F201%2Fmod_glossary%2Fattachment%2F24%2FSignos%2520Vitales%2520-%2520Semiolog%25C3%25ADa.pdf&usg=AFQjCNF9zlc6rByrMVVJTrhtT4sa-u9x_Q&sig2=I1P6S6-1N90SSDyQQgWYNg&cad=rja

[28] Edgar Lopategui Corsino. “Procedimientos A Seguir Para Determinar La Frecuencia Cardiaca En Reposo”. Accedido 01 de Mayo 2017 [Online]. Disponible en: http://www.saludmed.com/labsfisiologiaejercicio/reposocardiovascular/LAB_D5-Frecuencia_Cardiaca.pdf.

[29] Luis Azcona. “El Electrocardiográma”. Accedido 01 de Mayo 2017 [Online]. Disponible en: http://www.fbbva.es/TLFU/microsites/salud_cardio/mult/fbbva_libroCorazon_cap4.pdf.

[30] Cristian Aguilar. “El futuro de los dispositivos wearables, ingeribles e incrustables para nuestra salud”. Accedido 01 Mayo 2017 [Online]. Disponible en: http://www.sabien.upv.es/futuro-los-dispositivos-wearables-ingeribles-e-incrustables-nuestra-salud/

[31] J. M. Rodriguez, “Adquisicion de senales biologicas.” Grupo de Sistemas Inteligentes Universidad de Santiago, 2005. Available: http://www.usc.es/catedras/telemedicina/2005/materialAsignatura/AdquisicionSenalesBiologicas.pdf. [Ultimo acceso: 24 03 2018]

[32] Margarita Álvarez Cervera, Lanz Euán Oscar Armando, Pech Dzul José Joaquín. “SISTEMA MONITOR DE SEÑALES DE USO MÉDICO”. Instituto Tecnológico de Mérida.

[33] Alexis Meneses Arévalo Daissy Carola Toloza Cano. “PROYECTO DISEÑO Y CONSTRUCCIÓN DE UN MONITOR DE SIGNOS VITALES BASADO EN UN COMPUTADOR PORTÁTIL”. DALCAME.

[34] José Carlos Herrero Herranz, Jesús Sánchez Allende. “UNA MIRADA AL MUNDO ARDUINO”. UNIVERSIDAD ALFONSO X EL SABIO Escuela Politécnica Superior Villanueva de la Cañada (Madrid). Mayo, 2015.

[35] American Thoracic Society. “Oximetría de pulso”. Am J Respir Crit Care Med Vol. 184, pp. 1-2, 2013.

[36] Victoria Ramos, Pilar García-Santesmases, Silvia de Miguel-Bilbao, Jorge García, José Roldán, Miguel García-Santesmases, Alejandro Úbeda, M. Antonia Martínez, M. Antonia Cid, Lucía Chacón, M. Ángeles Trillo, Francisco Falcone, Leire Azpilicueta, M. Dolores Marcos y José L. Bardasano. “Innovación Tecnológica Para La Salud Y La Seguridad Electromagnética Personal Tecnological Innovation For Health And Personal Electromagnetic Safety”. Madrid: Victoria Ramos (ISCII), 2013.

[37] FIME. “Definición Open Source”. Accedido 02 Mayo 2017 [Online]. Disponible en: http://www.fime.uanl.mx/jcedillo/Definic%EDondeOpenSourceySoftwareLibre.pdf.

[38] BBVAOPEN4U. “Open Source”. Accedido: 02 Mayo 2017 [Online]. Disponible en: https://bbvaopen4u.com/sites/default/files/bbva-open4u-ebook-sofware_v2.pdf.

[39] Leonardo Betancur, Ph.D. “Redes de área corporal. Una perspectiva al futuro desde la investigación”. Accedido 02 Mayo 2017 [Online]. Disponible en: https://www.icesi.edu.co/revistas/index.php/sistemas_telematica/article/viewFile/1027/1052.

[40] Daniel Remiro Aramendia. “Dispositivo Hrm Y Red Ad-Hoc Para Compartir Y Geolocalizar A Un Grupo De Corredores De Running”. Ing. Tesis. Escuela Técnica Superior De Ingenieros Industriales Y De Telecomunicación. Pamplona. Julio 2014

[41] IEEE Project. “Multi-hop WBAN Construction for Healthcare IoT Systems”. Accedido 02 Mayo 2017 [Online]. Disponible en: http://ieeeproject.info/blog/multi-hop-wban-construction-for-healthcare-iot-systems/

[42] Developer Android. “Conoce Android Studio”, Accedido 03 Junio 2017 [Online]. Disponible en: https://developer.android.com/studio/intro/index.html?hl=es-419#estructura_del_proyecto.

[43] ARCORE, “Getting Started with Java”, Accedido 02 Mayo 2017 [Online]. Disponible: https://developers.google.com/ar/develop/java/getting-started.

[44] TECHMAKE. “Android Studio tutorial for beginners”. Accedido 03 Junio 2017 [Online]. Disponible en: https://www.androidauthority.com/android-studio-tutorial-beginners-637572/.

[45] Developer Android. “Conoce Android Studio”, Accedido 03 Junio 2017 [Online]. Disponible en: https://developer.android.com/studio/intro/index.html?hl=es-419#estructura_del_proyecto.

[46] SPARKFUN. “Pro Micro & Fio V3 Hookup Guide”. learn.sparkfun. Accedido 03 Junio 2017 [Online]. Disponible en: https://learn.sparkfun.com/tutorials/pro-micro--fio-v3-hookup-guide/all.pdf.

[47] LilyPad Development Worksheets. “LilyPad Development”. Accedido 03 Junio 2017 [Online]. Disponible en: https://cdn.sparkfun.com/assets/resources/5/lilyPad_dev_handout.pdf.

[48] Becky Stern. “Getting Started with FLORA”. Accedido 03 Junio 2017 [Online]. Disponible en: https://cdn-learn.adafruit.com/downloads/pdf/getting-started-with-flora.pdf.

[49] DIGI-KEY. “*MAX3000E/MAX3001E/MAX3002–MAX3012*”. Accedido 03 Junio 2017 [Online]. Disponible en: https://datasheets.maximintegrated.com/en/ds/MAX3000E-MAX3012.pdf.

[50] MAXINTEGRATE. “MAX30100” ”. Accedido 03 Junio 2017 [Online]. Disponible en: https://datasheets.maximintegrated.com/en/ds/MAX30100.pdf.

[51] Gearbest. “arjeta de Pulso con Monitor para Arduino Keyestudio”. Accedido 03 Junio 2017 [Online]. Disponible en: https://es.gearbest.com/boards-shields/pp_351337.html

[52] Adafruit. “Adafruit Flora Bluefruit LE”. Accedido 03 Junio 2017 [Online]. Disponible en: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-flora-bluefruit-le.pdf.

[53] Makezine. “HC Serial Bluetooth Products User Instructional Manual”. Accedido 03 Junio 2017 [Online]. Disponible en: https://cdn.makezine.com/uploads/2014/03/hc_hc-05-user-instructions-bluetooth.pdf.

[54] Guangzhou HC Information Technology Co., Ltd. “Product Data Sheet”. Accedido 03 Junio 2017 [Online]. Disponible en: https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf.

[55] INDUSTRIAL BATTERIES. “NI-MH HaNdbook”. Accedido 03 Junio 2017 [Online]. Disponible en: https://www.mouser.com/pdfdocs/PanasonicBatteries_NI-MH_Handbook.pdf.

[56] HARDTOFIND. “Batería de Tecnología Li-Po 11.1V/5200mAh/3S2P/10C”. Accedido 03 Junio 2017 [Online]. Disponible en: http://hardtofind.com.mx/pdfs/91270016.PDF.

[57] EEMB. “Lithium-ion Battery DATA SHEET”. Accedido 03 Junio 2017 [Online]. Disponible en: https://www.ineltro.ch/media/downloads/SAAItem/45/45958/36e3e7f3-2049-4adb-a2a7-79c654d92915.pdf.

[58] DIGI-KEY. “Adding Heart-Rate Monitoring Functionality to Fitness Gear”. Accedido: 28 Enero 2018 [Online]. Disponible en: https://www.digikey.com/en/articles/techzone/2016/nov/adding-heart-rate-monitoring-functionality-to-fitness-gear.

ANEXOS

ANEXO A

Código de programación Arduino Pro_Micro

```
#include <SoftwareSerial.h>
#include <Arduino.h>
#include "algorithm.h"
#include "max30102.h"
#define MAX_BRIGHTNESS 255
#if defined(ARDUINO_AVR_MICRO)
// Arduino Pro Micro no tiene suficiente SRAM para almacenar 100 muestras de datos led IR y datos led rojos en formato de 32 bits
// Para resolver este problema, se truncarán los MSB de 16 bits de los datos muestreados. Las muestras se convierten en datos de 16 bits.
uint16_t aun_ir_buffer[100]; //datos del sensor LED infrarrojo
uint16_t aun_red_buffer[100]; //datos del sensor LED rojo
#else
uint32_t aun_ir_buffer[100]; //datos del sensor LED infrarrojo
uint32_t aun_red_buffer[100]; //datos del sensor LED rojo
#endif
int32_t n_ir_buffer_length; //longitud del dato
int32_t n_spo2; //valor SPO2
int8_t ch_spo2_valid; //indicador para mostrar si el cálculo de SPO2 es válido
int32_t n_heart_rate; //valor de frecuencia cardíaca
int8_t ch_hr_valid; //indicador para mostrar si el cálculo de la frecuencia cardíaca es válido
uint8_t uch_dummy;
// Variables para medir los pulsos cardiacos
int pulso_anterior=0;
int contador;
int restahr;
int cntN;
int contnd;
String cadena;
String CadBatnd;
#define vcc 6
#define alerta 5
#define dedo 4
#define analogobateria 3
int analogValor = 0;
float voltaje = 0;
int porcent;
//SoftwareSerial SerialGSM(RxD, TxD); // RX, TX
SoftwareSerial SerialGSM(8, 9); // RX, TX
// la rutina de instalación se ejecuta una vez cuando presionas restablecer:
void setup(){
  maxim_max30102_reset(); //restablece el MAX30102
  // inicializar la comunicación serial a 115200 bits por segundo:
  Serial.begin(115200);
  SerialGSM.begin(57600);
  pinMode(10, INPUT); //el pin D10 se conecta al pin de salida de interrupción del MAX30102
  pinMode(vcc, OUTPUT);
  pinMode(alerta, OUTPUT);
  pinMode(dedo, OUTPUT);
  digitalWrite(vcc, HIGH);
  digitalWrite(alerta, LOW);
  digitalWrite(dedo, LOW);
  delay(1000);
maxim_max30102_read_reg(REG_INTR_STATUS_1,&uch_dummy); //Lee / borra el registro de estado de interrupción
uch_dummy=Serial.read();
  maxim_max30102_init(); //inicializar el MAX30102
}// la rutina de bucle se repite una y otra vez para siempre:
void loop() {
  uint32_t un_min, un_max, un_prev_data, un_brightness; //variables para calcular el brillo del LED a bordo que refleja los latidos del corazón
  int32_t i;
  float f_temp;
  un_brightness=0;
  un_min=0x3FFFF;
  un_max=0;
  n_ir_buffer_length=100; //longitud del buffer de 100 datos almacenados en 4 segundos de muestras ejecutándose a 25sps
  //lea las primeras 100 muestras y determine el rango de señal
  for(i=0;i<n_ir_buffer_length;i++) {
    while(digitalRead(10)==1); //espere hasta que el pin de interrupción afirme
maxim_max30102_read_fifo((aun_red_buffer+i), (aun_ir_buffer+i)); //leer de MAX30102 FIFO
```

```
   if(un_min>aun_red_buffer[i])
     un_min=aun_red_buffer[i];  //actualizar la señal min
   if(un_max<aun_red_buffer[i])
     un_max=aun_red_buffer[i];  //actualizar la señal max
   Serial.print(F("red="));
   Serial.print(aun_red_buffer[i], DEC);
   Serial.print(F(", ir="));
   Serial.println(aun_ir_buffer[i], DEC);
  }
  un_prev_data=aun_red_buffer[i];
  //calcular la frecuencia cardíaca y la SpO2 después de las primeras 100 muestras (los primeros 4 segundos de las muestras)
maxim_heart_rate_and_oxygen_saturation(aun_ir_buffer, n_ir_buffer_length, aun_red_buffer, &n_spo2, &ch_spo2_valid, &n_heart_rate, &ch_hr_valid);
  //Continuamente tomando muestras de MAX30102. La frecuencia cardíaca y la SpO2 se calculan cada 1 segundo
  while(1)  {
   i=0;
   un_min=0x3FFFF;
   un_max=0;
   //descargar los primeros 25 conjuntos de muestras en la memoria y cambiar los últimos 75 conjuntos de muestras a la parte superior
   for(i=25;i<100;i++)    {
     aun_red_buffer[i25]=aun_red_buffer[i];
     aun_ir_buffer[i-25]=aun_ir_buffer[i];
     //actualizar la señal min y max
     if(un_min>aun_red_buffer[i])
       un_min=aun_red_buffer[i];
     if(un_max<aun_red_buffer[i])
       un_max=aun_red_buffer[i];
   }
   //tome 25 conjuntos de muestras antes de calcular la frecuencia cardíaca.
   for(i=75;i<100;i++)
   {
     un_prev_data=aun_red_buffer[i-1];
     while(digitalRead(10)==1);
     digitalWrite(9, !digitalRead(9));
maxim_max30102_read_fifo((aun_red_buffer+i), (aun_ir_buffer+i));
     //calcule el brillo del LED
     if(aun_red_buffer[i]>un_prev_data)      {
       f_temp=aun_red_buffer[i]-un_prev_data;
       f_temp/=(un_max-un_min);
       f_temp*=MAX_BRIGHTNESS;
       f_temp=un_brightness-f_temp;
       if(f_temp<0)
         un_brightness=0;
       else
         un_brightness=(int)f_temp;
     }
     else
     {
       f_temp=un_prev_data-aun_red_buffer[i];
       f_temp/=(un_max-un_min);
       f_temp*=MAX_BRIGHTNESS;
       un_brightness+=(int)f_temp;
if(un_brightness>MAX_BRIGHTNESS)
un_brightness=MAX_BRIGHTNESS;
     }
     if(ch_hr_valid==1 && ch_spo2_valid==1)
     {
       restahr=n_heart_rate-pulso_anterior;
       if(restahr > -5 && restahr < 5 && restahr !=0 && contador ==1 )
       {
         if(n_heart_rate>79  &&  n_heart_rate<161  &&  n_spo2>94 && n_spo2<101)
         {
           visualizacion();
         }else{
           visualizacion_alerta();
         }
       }else if(restahr == 0 && contador ==1){ if(cntN>30)
         {
           Serial.print(F("RED: "));
           Serial.print(F("HR="));
           Serial.print(n_heart_rate, DEC);
           Serial.print(F(", SPO2="));
           Serial.println(n_spo2, DEC);
           if(n_heart_rate>79  &&  n_heart_rate<161  &&  n_spo2>94 && n_spo2<101)
           {
             visualizacion();
           }else{
             visualizacion_alerta();
           }
           cntN=0;
           delay(500);
         }
         cntN ++;
       }else if(restahr < -5 || restahr > 5){
         contador ++;
       }else if(contador > 2){
         reseteo:
         delay(250);
         maxim_max30102_reset();       //restablece el MAX30102
         delay(250);
maxim_max30102_read_reg(REG_INTR_STATUS_1,&uch_dummy);  //Lee / borra el registro de estado de interrupción
         delay(250);
         maxim_max30102_init();        //inicializar el MAX30102
         delay(250);
         contador=0;
       } else{
//         Serial.println("no considera opciones");
       }
       pulso_anterior=n_heart_rate;
     }
     else
```

```
{
 Bateria();
 Serial.println("No dedo");
 Serial.println(porcent);
 CadBatnd = "NDB"+(String)porcent+"F";
 SerialGSM.println(CadBatnd);
 contnd ++;
 delay(250);
 if(contnd > 15){
  Bateria();
  CadBatnd = "NDB"+(String)porcent+"F";
  SerialGSM.println(CadBatnd);
  Serial.println(porcent);
  digitalWrite(alerta, LOW);
  goto reseteo;
  }
 digitalWrite(alerta, LOW);
 digitalWrite(dedo, HIGH);
 }
}
maxim_heart_rate_and_oxygen_saturation(aun_ir_buffer, n_ir_buffer_length, aun_red_buffer, &n_spo2, &ch_spo2_valid, &n_heart_rate, &ch_hr_valid);
```

ANEXO B

Código de programación Comunicación I2C

```
#ifndef SOFTI2CMASTER_H_
#define SOFTI2DMASTER_H_
#include <avr/io.h>
#include <Arduino.h>
#ifdef ARDUINO_AVR_MICRO
// para arduino basado en atmega32u4 (MICRO, MICRO PRO, LEONARDO).
// busca el pinout de MCU en los archivos de esquema y agrega otro #IF_CASE
// con pinout donde se encuentran los pines del bus i2c. IPC17EC
#define SDA_PORT PORTD
#define SDA_PIN 1
#define SCL_PORT PORTD
#define SCL_PIN 0
#endif
#ifdef ARDUINO_AVR_MEGA2560
#define SDA_PORT PORTD
#define SDA_PIN 1
#define SCL_PORT PORTD
#define SCL_PIN 0
#endif
#define I2C_TIMEOUT 100
#define I2C_NOINTERRUPT 0
#define I2C_SLOWMODE 1
#define FAC 1
#define I2C_CPUFREQ (F_CPU/FAC)
#ifndef _SOFTI2C_H
#define _SOFTI2C_H   1
// Función de inicio. Necesita ser llamado una vez al principio.
// Devuelve falso si SDA o SCL son bajos, lo que probablemente significa
// un bloqueo de bus I2C o que las líneas no se detienen (IPC17 mdf).
boolean __attribute__ ((noinline)) i2c_init(void) __attribute__ ((used));
// Función de inicio de transferencia: <addr> es la dirección I2C de 8 bits (incluido el R / W
// poco).
// Devuelve: true si el esclavo responde con un "reconocimiento", false en caso contrario
bool __attribute__ ((noinline)) i2c_start(uint8_t addr) __attribute__ ((used));
// Similar a la función de inicio, pero espera un ACK! Tenga cuidado, esto puede
// ¡da como resultado un ciclo infinito!
void __attribute__ ((noinline)) i2c_start_wait(uint8_t addr) __attribute__ ((used));
// Función de inicio repetido: después de haber reclamado el bus con una condición de inicio,
// puedes direccionar otro o el mismo chip nuevamente sin una intervención
// detener la condición.
// Devuelve: true si el esclavo responde con un "reconocimiento", false en caso contrario
bool __attribute__ ((noinline)) i2c_rep_start(uint8_t addr) __attribute__ ((used));
// Emita una condición de detención, liberando el bus.
void __attribute__ ((noinline)) i2c_stop(void) asm("ass_i2c_stop") __attribute__ ((used));
// Escribe un byte en el chip esclavo que se ha tratado
// por la llamada de inicio anterior. <valor> es el byte que se enviará.
// Devuelve: true si el esclavo responde con un "reconocimiento", false en caso contrario
bool __attribute__ ((noinline)) i2c_write(uint8_t value) asm("ass_i2c_write") __attribute__ ((used));
// Leer un byte. Si <last> es verdadero, enviamos un NAK después de haber recibido
// el byte para terminar la secuencia de lectura (IPC17mdf).
uint8_t __attribute__ ((noinline)) i2c_read(bool last) __attribute__ ((used));
// Puedes configurar I2C_CPUFREQ independientemente de F_CPU si
// cambia la frecuencia de la CPU sobre la marcha. Si no lo defines,
// usará el valor de F_CPU
#ifndef I2C_CPUFREQ
#define I2C_CPUFREQ F_CPU
#endif
// CPU clock:        1MHz  2MHz  4MHz  8MHz  16MHz  20MHz
// Fast I2C mode      40    80    150   300   400    400
// Standard I2C mode  40    80    100   100   100    100
// Slow I2C mode      25    25    25    25    25     25
```

```
#define I2C_READ    1
#define I2C_WRITE   0
#define                                                 SDA_DDR
            (_SFR_IO_ADDR(SDA_PORT) - 1)
#define                                                 SCL_DDR
            (_SFR_IO_ADDR(SCL_PORT) - 1)
#define                                                 SDA_OUT
            _SFR_IO_ADDR(SDA_PORT)
#define                                                 SCL_OUT
            _SFR_IO_ADDR(SCL_PORT)
#define SDA_IN
            (_SFR_IO_ADDR(SDA_PORT) - 2)
#define SCL_IN
            (_SFR_IO_ADDR(SCL_PORT) - 2)
#ifndef __tmp_reg__
#define __tmp_reg__ 0
#endif
void    __attribute__    ((noinline))    i2c_delay_half(void)
asm("ass_i2c_delay_half") __attribute__ ((used));
void    __attribute__    ((noinline))    i2c_wait_scl_high(void)
asm("ass_i2c_wait_scl_high") __attribute__ ((used));
void  i2c_delay_half(void)
{ // function call 3 cycles => 3C
#if I2C_NOINTERRUPT
    " cli \n\t"
#endif
    " sbi     %[SCLDDR],%[SCLPIN]          ;force      SCL
low \n\t"
    " rcall   ass_i2c_delay_half    ;delay  T/2 \n\t"
    " cbi     %[SDADDR],%[SDAPIN]          ;release    SDA
\n\t"
    " rcall   ass_i2c_delay_half    ;delay T/2 \n\t"
    " cbi     %[SCLDDR],%[SCLPIN]          ;release    SCL
\n\t"
    " rcall   ass_i2c_delay_half    ;delay  T/2 \n\t"
    " sbis      %[SCLIN],%[SCLPIN]       ;check for clock
stretching slave\n\t"
    " rcall    ass_i2c_wait_scl_high   ;wait until SCL=H\n\t"
    " sbi     %[SDADDR],%[SDAPIN]          ;force      SDA
low \n\t"
    " rcall   ass_i2c_delay_half    ;delay       T/2 \n\t"
    " rcall    ass_i2c_write       \n\t"
    " ret"
    : : [SCLDDR] "I"    (SCL_DDR), [SCLPIN] "I"
(SCL_PIN),[SCLIN] "I" (SCL_IN),
       [SDADDR] "I"     (SDA_DDR), [SDAPIN] "I"
(SDA_PIN));
  return true; // just to fool the compiler
}
void  i2c_start_wait(uint8_t addr)
{
 __asm__ __volatile__
  (
   " push    r24                  ;save original parameter \n\t"
   "_Li2c_start_wait1: \n\t"
   " pop      r24                 ;restore original parameter\n\t"
   " push     r24                  ;and save again \n\t"
#if I2C_NOINTERRUPT
    " sei                          ;enable interrupts again!\n\t"
#endif
    : : [SCLDDR] "I"    (SCL_DDR), [SCLPIN] "I"
(SCL_PIN), [SCLIN] "I" (SCL_IN),
       [SDADDR] "I"     (SDA_DDR), [SDAPIN] "I"
(SDA_PIN));
}
bool i2c_write(uint8_t value)
{
 __asm__ __volatile__
  (
    " sec                          ;set carry flag \n\t"
   " rol      r24                 ;shift in carry and shift out MSB
\n\t"
    " rjmp _Li2c_write_first \n\t"
    "_Li2c_write_bit:\n\t"
    " lsl      r24                 ;left shift into carry ;; 1C\n\t"
    "_Li2c_write_first:\n\t"
    " breq     _Li2c_get_ack       ;jump if TXreg is empty;;
+1 = 2C \n\t"
    " sbi      %[SCLDDR],%[SCLPIN]     ;force SCL low
;; +2 = 4C \n\t"
    " nop \n\t"
    " nop \n\t"
    " nop \n\t"
    "          brcc                                _Li2c_write_low
;;+1/+2=5/6C\n\t"
    " nop                                          ;; +1 = 7C \n\t"
    " cbi %[SDADDR],%[SDAPIN]                  ;release
SDA          ;; +2 = 9C \n\t"
    " rjmp      _Li2c_write_high                       ;; +2 =
11C \n\t"
    "_Li2c_write_low: \n\t"
    " sbi      %[SDADDR],%[SDAPIN]         ;force      SDA
low         ;; +2 = 9C \n\t"
    " rjmp   _Li2c_write_high                      ;;+2 = 11C
\n\t"
    "_Li2c_write_high: \n\t"
```

ANEXO C

Código de programación Algoritmo

```
#include "algorithm.h"
#include <Arduino.h>
#if defined(ARDUINO_AVR_MICRO)
// Arduino Pro Micro no tiene suficiente SRAM para almacenar 100 muestras de datos led IR y datos led rojos en formato de 32 bits
// Para resolver este problema, se truncarán los MSB de 16 bits de los datos muestreados. Las muestras se convierten en datos de 16 bits.
void maxim_heart_rate_and_oxygen_saturation(uint16_t *pun_ir_buffer, int32_t n_ir_buffer_length, uint16_t *pun_red_buffer, int32_t *pn_spo2, int8_t *pch_spo2_valid,
        int32_t *pn_heart_rate, int8_t *pch_hr_valid)
#else
void maxim_heart_rate_and_oxygen_saturation(uint32_t *pun_ir_buffer, int32_t n_ir_buffer_length, uint32_t *pun_red_buffer, int32_t *pn_spo2, int8_t *pch_spo2_valid,
        int32_t *pn_heart_rate, int8_t *pch_hr_valid)
#endif
/**
* \brief       Calcule la frecuencia cardíaca y el nivel de SpO2
* \par         Detalles
*              Al detectar los picos del ciclo de PPG y la correspondiente señal de AC / DC de rojo / infrarrojo, se calcula la an_ratio para el SPO2.
*              Dado que este algoritmo apunta a Arm M0 / M3. formaula para SPO2 no alcanzó la precisión debido al desbordamiento de registro.
*              Por lo tanto, SPO2 preciso es precalculado y ahorra longo uch_spo2_table [] por cada an_ratio.
* \ param [in] * pun_ir_buffer - buffer de datos del sensor IR
* \ param [in] n_ir_buffer_length - longitud del buffer de datos del sensor IR
* \ param [in] * pun_red_buffer - Memoria intermedia de datos del sensor rojo
* \ param [out] * pn_spo2 - valor de SpO2 calculado
* \ param [out] * pch_spo2_valid - 1 si el valor de SpO2 calculado es válido
* \ param [out] * pn_heart_rate - Valor de frecuencia cardíaca calculado
* \ param [out] * pch_hr_valid - 1 si el valor de frecuencia cardíaca calculado es válido
*
* \ retval     Ninguno
*/
{
  uint32_t un_ir_mean,un_only_once ;
  int32_t k, n_i_ratio_count;
  int32_t i, s, m, n_exact_ir_valley_locs_count, n_middle_idx;
  int32_t n_th1, n_npks, n_c_min;
  int32_t an_ir_valley_locs[15] ;
  int32_t n_peak_interval_sum;
  int32_t n_y_ac, n_x_ac;
  int32_t n_spo2_calc;
  int32_t n_y_dc_max, n_x_dc_max;
  int32_t n_y_dc_max_idx, n_x_dc_max_idx;
  int32_t an_ratio[5], n_ratio_average;
  int32_t n_nume, n_denom ;
  // calcula DC media y resta DC de ir
  un_ir_mean =0;
  for (k=0 ; k<n_ir_buffer_length ; k++ ) un_ir_mean += pun_ir_buffer[k] ;
  un_ir_mean =un_ir_mean/n_ir_buffer_length ;
// eliminar DC e invertir la señal para que podamos usar el detector de pico como detector de valle
  for (k=0 ; k<n_ir_buffer_length ; k++ )
    an_x[k] = -1*(pun_ir_buffer[k] - un_ir_mean) ;
// Media móvil de 4 puntos
  for(k=0; k< BUFFER_SIZE-MA4_SIZE; k++){
    an_x[k]=(  an_x[k]+an_x[k+1]+  an_x[k+2]+ an_x[k+3])/(int)4;
  }
  // calcular el umbral
  n_th1=0;
  for ( k=0 ; k<BUFFER_SIZE ;k++){
    n_th1 +=  an_x[k];
  }
  n_th1=  n_th1/ ( BUFFER_SIZE);
  if( n_th1<30) n_th1=30; // min allowed
  if( n_th1>60) n_th1=60; // max allowed

  for ( k=0 ; k<15;k++) an_ir_valley_locs[k]=0;
```

```
 // desde que volteamos la señal, utilizamos detector de pico como detector de valle
 maxim_find_peaks( an_ir_valley_locs, &n_npks, an_x, BUFFER_SIZE, n_th1, 4, 15 );//altura del pico, distancia máxima, max_num_pages
 n_peak_interval_sum =0;
 if (n_npks>=2){
  for (k=1; k<n_npks; k++) n_peak_interval_sum += (an_ir_valley_locs[k] -an_ir_valley_locs[k -1] ) ;
  n_peak_interval_sum =n_peak_interval_sum/(n_npks-1);
  *pn_heart_rate =(int32_t)( (FS*60)/ n_peak_interval_sum );
  *pch_hr_valid  = 1;
 }
 else  {
  *pn_heart_rate = -999; // no se puede calcular porque # de picos son demasiado pequeños
  *pch_hr_valid  = 0;
 }
 //  carga el valor bruto nuevamente para el cálculo de SPO2: ROJO (= y) e IR (= X)
 for (k=0 ; k<n_ir_buffer_length ; k++ )  {
    an_x[k] =  pun_ir_buffer[k] ;
    an_y[k] =  pun_red_buffer[k] ;
 }
 // encuentra el min preciso cerca de an_ir_valley_locs
 n_exact_ir_valley_locs_count =n_npks;
 //utilizando exact_ir_valley_locs, busque ir-red DC andir-red AC para SPO2 calibración an_ratio
 //encontrar AC / DC máximo de primera
 n_ratio_average =0;
 n_i_ratio_count = 0;
 for(k=0; k< 5; k++) an_ratio[k]=0;
 for (k=0; k< n_exact_ir_valley_locs_count; k++){
  if (an_ir_valley_locs[k] > BUFFER_SIZE ){
   *pn_spo2 =  -999 ; // no use SPO2 ya que el loc valle está fuera de rango
    *pch_spo2_valid  = 0;
    return;
  }
 }
 // encontrar el máximo entre dos ubicaciones de valle
 // y utilice una relación entre el componente de CA de Ir & Red y el componente DC de Ir & Red para SPO2
 for (k=0; k< n_exact_ir_valley_locs_count-1; k++){
  n_y_dc_max= -16777216 ;
  n_x_dc_max= -16777216;
  if (an_ir_valley_locs[k+1]-an_ir_valley_locs[k] >3){
      for (i=an_ir_valley_locs[k]; i< an_ir_valley_locs[k+1]; i++){
       if (an_x[i]> n_x_dc_max) {n_x_dc_max =an_x[i]; n_x_dc_max_idx=i;}
       if (an_y[i]> n_y_dc_max) {n_y_dc_max =an_y[i]; n_y_dc_max_idx=i;}
    }
    n_y_ac= (an_y[an_ir_valley_locs[k+1]] - an_y[an_ir_valley_locs[k] ] )*(n_y_dc_max_idx - an_ir_valley_locs[k]); //red
    n_y_ac=  an_y[an_ir_valley_locs[k]] + n_y_ac/ (an_ir_valley_locs[k+1] - an_ir_valley_locs[k]) ;
    n_y_ac=  an_y[n_y_dc_max_idx] - n_y_ac;  // restar componentes lineales de DC de principio
    n_x_ac=  (an_x[an_ir_valley_locs[k+1]] - an_x[an_ir_valley_locs[k] ] )*(n_x_dc_max_idx - an_ir_valley_locs[k]); // ir
    n_x_ac=  an_x[an_ir_valley_locs[k]] + n_x_ac/ (an_ir_valley_locs[k+1] - an_ir_valley_locs[k]);
    n_x_ac=  an_x[n_y_dc_max_idx] - n_x_ac;  // restar componentes lineales de DC de principio
    n_nume=( n_y_ac *n_x_dc_max)>>7 ; //prepare X100 para preservar el valor flotante
    n_denom= ( n_x_ac *n_y_dc_max)>>7;
    if (n_denom>0  && n_i_ratio_count <5 &&  n_nume != 0)
    {
     an_ratio[n_i_ratio_count]= (n_nume*100)/n_denom ; //Fórmula es ( n_y_ac *n_x_dc_max) / ( n_x_ac *n_y_dc_max) ;
      n_i_ratio_count++;
    }
   }
 }
 // elija el valor mediano ya que la señal PPG puede variar de latido a latido
 maxim_sort_ascend(an_ratio, n_i_ratio_count);
 n_middle_idx= n_i_ratio_count/2;
 if (n_middle_idx >1)
  n_ratio_average =( an_ratio[n_middle_idx-1] +an_ratio[n_middle_idx])/2; // use median
 else
  n_ratio_average = an_ratio[n_middle_idx ];
 if( n_ratio_average>2 && n_ratio_average <184){
  n_spo2_calc= uch_spo2_table[n_ratio_average] ;
  *pn_spo2 = n_spo2_calc ;
  *pch_spo2_valid  = 1;//  float_SPO2 = -45.060 * n_ratio_average * n_ratio_average / 10000 + 30.354 * n_ratio_average / 100 + 94.845; // para comparar con la tabla
 }
 else{
  *pn_spo2 =  -999 ; // no use SPO2 ya que la señal an_ratio está fuera de rango
  *pch_spo2_valid  = 0;
 }
}
void maxim_find_peaks( int32_t *pn_locs, int32_t *n_npks, int32_t *pn_x, int32_t n_size, int32_t n_min_height, int32_t n_min_distance, int32_t n_max_num )
/**
* \brief       Find peaks
* \par         Details
*                  Find at most MAX_NUM peaks above MIN_HEIGHT separated by at least MIN_DISTANCE
*
* \retval      None
*/
{
```

```
 maxim_peaks_above_min_height( pn_locs, n_npks, pn_x, n_size, n_min_height );
 maxim_remove_close_peaks( pn_locs, n_npks, pn_x, n_min_distance );
 *n_npks = min( *n_npks, n_max_num );
}
void maxim_peaks_above_min_height( int32_t *pn_locs, int32_t *n_npks, int32_t *pn_x, int32_t n_size, int32_t n_min_height )
/**
* \brief       Find peaks above n_min_height
* \par         Details
*              Find all peaks above MIN_HEIGHT
*
* \retval      None
*/{
 int32_t i = 1, n_width;
 *n_npks = 0;
 while (i < n_size-1){
   if (pn_x[i] > n_min_height && pn_x[i] > pn_x[i-1]){
// find left edge of potential peaks
     n_width = 1;
    while (i+n_width < n_size && pn_x[i] == pn_x[i+n_width])  // find flat peaks
       n_width++;
    if (pn_x[i] > pn_x[i+n_width] && (*n_npks) < 15 ){
// find right edge of peaks
       pn_locs[(*n_npks)++] = i;
       // for flat peaks, peak location is left edge
       i += n_width+1;
    }else
       i += n_width;
   }else
     i++;
 }
}
void maxim_remove_close_peaks(int32_t *pn_locs, int32_t *pn_npks, int32_t *pn_x, int32_t n_min_distance)
/**
* \brief       Remove peaks
* \par         Details
*                    Remove peaks separated by less than MIN_DISTANCE
*
* \retval      None
*/
{
 int32_t i, j, n_old_npks, n_dist;
 /* Order peaks from large to small */
 maxim_sort_indices_descend( pn_x, pn_locs, *pn_npks );
 for ( i = -1; i < *pn_npks; i++ ){
   n_old_npks = *pn_npks;
   *pn_npks = i+1;
   for ( j = i+1; j < n_old_npks; j++ ){
     n_dist =  pn_locs[j] - ( i == -1 ? -1 : pn_locs[i] ); // lag-zero peak of autocorr is at index -1
     if ( n_dist > n_min_distance || n_dist < -n_min_distance )
       pn_locs[(*pn_npks)++] = pn_locs[j];
   }
 }
 // Resort indices int32_to ascending order
 maxim_sort_ascend( pn_locs, *pn_npks );
}void maxim_sort_ascend(int32_t  *pn_x, int32_t n_size)
/**
* \brief       Sort array
* \par         Details
*                Sort array in ascending order (insertion sort algorithm)
*
* \retval      None
*/
{
 int32_t i, j, n_temp;
 for (i = 1; i < n_size; i++) {
   n_temp = pn_x[i];
   for (j = i; j > 0 && n_temp < pn_x[j-1]; j--)
      pn_x[j] = pn_x[j-1];
   pn_x[j] = n_temp;
 }
}void maxim_sort_indices_descend( int32_t *pn_x, int32_t *pn_indx, int32_t n_size)
/**
* \brief       Sort indices
* \par         Details
*                Sort indices according to descending order (insertion sort algorithm)
*
* \retval      None
*/
{ int32_t i, j, n_temp;
 for (i = 1; i < n_size; i++) {
   n_temp = pn_indx[i];
   for (j = i; j > 0 && pn_x[n_temp] > pn_x[pn_indx[j-1]]; j--)
     pn_indx[j] = pn_indx[j-1];
   pn_indx[j] = n_temp;
```

ANEXO D

Código de programación MAX3012 “B”

```
#ifndef MAX30102_H_
#define MAX30102_H_
#include <Arduino.h>
#define I2C_WRITE_ADDR 0xAE
#define I2C_READ_ADDR 0xAF
//register addresses
#define REG_INTR_STATUS_1 0x00
#define REG_INTR_STATUS_2 0x01
#define REG_INTR_ENABLE_1 0x02
#define REG_INTR_ENABLE_2 0x03
#define REG_FIFO_WR_PTR 0x04
#define REG_OVF_COUNTER 0x05
#define REG_FIFO_RD_PTR 0x06
#define REG_FIFO_DATA 0x07
#define REG_FIFO_CONFIG 0x08
#define REG_MODE_CONFIG 0x09
#define REG_SPO2_CONFIG 0x0A
#define REG_LED1_PA 0x0C
#define REG_LED2_PA 0x0D
#define REG_PILOT_PA 0x10
#define REG_MULTI_LED_CTRL1 0x11
#define REG_MULTI_LED_CTRL2 0x12
#define REG_TEMP_INTR 0x1F
#define REG_TEMP_FRAC 0x20
#define REG_TEMP_CONFIG 0x21
#define REG_PROX_INT_THRESH 0x30
#define REG_REV_ID 0xFE
#define REG_PART_ID 0xFF

bool maxim_max30102_init();
#if defined(ARDUINO_AVR_MICRO)
// Arduino Pro Micro no tiene suficiente SRAM para almacenar 100 muestras de datos led IR y datos led rojos en formato de 32 bits
// Para resolver este problema, se truncarán los MSB de 16 bits de los datos muestreados. Las muestras se convierten en datos de 16 bits.
bool maxim_max30102_read_fifo(uint16_t *pun_red_led, uint16_t *pun_ir_led);
#else
bool maxim_max30102_read_fifo(uint32_t *pun_red_led, uint32_t *pun_ir_led);
#endif
bool maxim_max30102_write_reg(uint8_t uch_addr, uint8_t uch_data);
bool maxim_max30102_read_reg(uint8_t uch_addr, uint8_t *puch_data);
bool maxim_max30102_reset(void);
#endif /*  MAX30102_H_ */
```

ANEXO E

Código de programación XML Visualización pantalla Android Studio

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.albertoruiz.babyhealth.MainActivity">

    <android.support.design.widget.TextInputLayout
        android:id="@+id/usuario"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_above="@+id/password"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginBottom="20dp">

        <EditText
            android:id="@+id/etusuario"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/usuario"
            android:inputType="text" />
    </android.support.design.widget.TextInputLayout>

    <android.support.design.widget.TextInputLayout
        android:id="@+id/password"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_above="@+id/btn_Ingresar"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginBottom="25dp"
        app:passwordToggleEnabled="true">

        <android.support.design.widget.TextInputEditText
            android:id="@+id/campo_password"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/password"
            android:inputType="textPassword" />
    </android.support.design.widget.TextInputLayout>
```

```
    <Button
        android:id="@+id/btn_Ingresar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="30dp"
        android:drawableLeft="@drawable/login"
        android:drawablePadding="12dp"
        android:text="Ingresar"
        android:textColor="@android:color/white"
        app:backgroundTint="#0091ea"
        android:layout_above="@+id/TvRegiser"
        android:layout_centerHorizontal="true" />

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/usuario"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        app:srcCompat="@mipmap/bebe" />

    <TextView
        android:id="@+id/TvRegiser"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="26dp"
        android:fontFamily="casual"
        android:text="Registrarse Aqui"
        android:textAlignment="center"
        android:textAllCaps="true"
        android:textColor="?android:attr/textColorSecondary"
        android:textSize="18sp"
        android:textStyle="bold" />

</RelativeLayout>
```

ANEXO F

Código de programación XML Visualización registro Android Studio

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.albertoruiz.babyhealth.Registro">

    <LinearLayout
        android:id="@+id/area_nombre"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:layout_alignParentTop="true">

        <ImageView
            android:id="@+id/img_nombre"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_vertical"
            android:src="@drawable/nombre" />

        <android.support.design.widget.TextInputLayout
            android:id="@+id/til_nombre"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="32dp">

            <EditText
                android:id="@+id/tvnombre"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:ems="10"
                android:hint="@string/nombre2"
                android:inputType="text"
                android:singleLine="true"
                android:textSize="14sp" />
        </android.support.design.widget.TextInputLayout>

    </LinearLayout>

    <LinearLayout
```

```
        android:id="@+id/area_apellido"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/area_nombre"
        android:orientation="horizontal">

        <ImageView
            android:id="@+id/img_apellido"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_vertical"
             />

        <android.support.design.widget.TextInputLayout
            android:id="@+id/til_apellido"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="82dp">

            <EditText
                android:id="@+id/tvapellido"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:ems="10"
                android:hint="@string/apellido"
                android:inputType="text"
                android:singleLine="true"
                android:textSize="14sp" />
        </android.support.design.widget.TextInputLayout>

    </LinearLayout>

    <LinearLayout
        android:id="@+id/area_usuario"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/area_apellido"
        android:orientation="horizontal">

        <ImageView
            android:id="@+id/img_usuario"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_vertical"
            android:src="@drawable/user" />

        <android.support.design.widget.TextInputLayout
            android:id="@+id/til_usuario"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:counterEnabled="true"
            app:counterMaxLength="10"
            android:layout_marginLeft="32dp">

            <EditText
                android:id="@+id/tvusuario"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
```

```
            android:ems="10"
            android:hint="@string/usuario2"
            android:inputType="text"
            android:maxLength="10"
            android:singleLine="true"
            android:textSize="14sp" />
    </android.support.design.widget.TextInputLayout>

</LinearLayout>

<LinearLayout
    android:id="@+id/area_password"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/area_usuario"
    android:orientation="horizontal">

    <ImageView
        android:id="@+id/img_password"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical"
        android:src="@drawable/password" />

    <android.support.design.widget.TextInputLayout
        android:id="@+id/tvpassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="32dp"
        android:textSize="14sp"
        app:passwordToggleEnabled="true">

        <android.support.design.widget.TextInputEditText
            android:id="@+id/cam_password"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="@string/password2"
            android:inputType="textPassword"
            android:singleLine="true" />
    </android.support.design.widget.TextInputLayout>

</LinearLayout>

    <LinearLayout
    android:id="@+id/area_telefono"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/area_password"
    android:orientation="horizontal">

    <ImageView
        android:id="@+id/img_telefono"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical"
        android:src="@drawable/phone" />
```

```
    <android.support.design.widget.TextInputLayout
        android:id="@+id/til_telefono"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="32dp">

        <EditText
            android:id="@+id/tvtelefono"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Telefono (0XXXXXXXXX)"
            android:inputType="phone"
            android:maxLength="10"
            android:singleLine="true"
            android:textSize="14sp" />
    </android.support.design.widget.TextInputLayout>

</LinearLayout>

<LinearLayout
    android:id="@+id/area_fechan"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/area_telefono"
    android:orientation="horizontal">

    <ImageView
        android:id="@+id/img_fechan"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical"
        android:src="@drawable/date" />

    <android.support.design.widget.TextInputLayout
        android:id="@+id/til_fechan"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:counterEnabled="true"
        app:counterMaxLength="2"
        android:layout_marginLeft="32dp">

        <EditText
            android:id="@+id/tvfechan"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Edad Bebe (Meses)"
            android:inputType="date"
            android:maxLength="10"
            android:singleLine="true"
            android:textSize="14sp" />
    </android.support.design.widget.TextInputLayout>

</LinearLayout>


<Button
```

```
        android:id="@+id/btn_registro"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:drawableLeft="@drawable/registrarse_user"
        android:drawablePadding="12dp"
        android:text="@string/btn_registrarse"
        android:textColor="@android:color/white"
        app:backgroundTint="#0091ea"
        android:layout_below="@+id/area_fechan"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="15dp" />

    <TextView
        android:id="@+id/TvcamposVacios"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/btn_registro"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"
        android:fontFamily="sans-serif-smallcaps"
        android:textAlignment="center"
        android:textAllCaps="false"
        android:textColor="@android:color/holo_red_dark"
        android:textSize="18sp"
        android:textStyle="bold|italic"/>


</RelativeLayout>
```

ANEXO G

Código de programación XML Visualización de datos Android Studio

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
tools:context="com.example.albertoruiz.babyhealth.Visualizacion">

    <LinearLayout
        android:id="@+id/area_txtBienvenido"
        android:layout_centerInParent="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:orientation="vertical"
        android:weightSum="1"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true">

        <TextView
            android:id="@+id/tvWelcome"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:layout_weight="0.15"
            android:text="Bienvenido"
            android:textAlignment="center"
            android:textSize="20sp"
            tools:layout_editor_absoluteX="16dp"
            tools:layout_editor_absoluteY="16dp" />

        <TextView
            android:id="@+id/tvdate"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.15"
            android:text="Fecha"
            android:textSize="18sp" />
        <LinearLayout
```

```
    android:id="@+id/lnSPO2"
    android:layout_centerInParent="true"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="0.36"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="SPO2  "
        android:textAlignment="center"
        android:textSize="30sp" />

    <TextView
        android:id="@+id/tvspo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="%"
        android:textSize="30sp" />

</LinearLayout>

<LinearLayout
    android:id="@+id/lnHR"
    android:layout_centerInParent="true"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="0.15"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/textView6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="HR  "
        android:textSize="30sp" />

    <TextView
        android:id="@+id/tvhr"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="ppm"
        android:textSize="30sp" />
</LinearLayout>

<TextView
    android:id="@+id/tvAlerta"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="0.15"
    android:textAlignment="center"
    android:textSize="24sp"/>
```

```
        <TextView
            android:id="@+id/tvmubi"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.15"
            android:textSize="24sp"/>

        <TextView
            android:id="@+id/tvporcentBat"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.15"
            />

        <LinearLayout
            android:id="@+id/lnView"
            android:layout_centerInParent="true"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentEnd="true"
            android:layout_alignParentRight="true"
            android:layout_alignParentTop="true"
            android:layout_marginTop="175dp"
            android:layout_weight="0.18"
            android:orientation="vertical">

            <TextView
                android:id="@+id/testView1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_above="@+id/txtString"
                android:text="dtin"
                android:textSize="15sp" />

            <TextView
                android:id="@+id/txtString"
                android:layout_width="wrap_content"
                android:layout_height="50dp"
                android:layout_alignLeft="@+id/testView1"
                android:layout_alignParentBottom="true"
                android:text="dtin"
                android:textSize="15sp" />

        </LinearLayout>

        <Button
            android:id="@+id/btn_registro"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.15"
            android:text="Registro"/>

    </LinearLayout>


</RelativeLayout>
```

ANEXO H

Código de programación XML Visualización de dispositivos Bluetooth Android Studio

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    >
    <TextView android:id="@+id/title_paired_devices"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Select btSerial device from paired devices:"
        android:visibility="gone"
        android:background="#666"
        android:textColor="#fff"
        android:paddingLeft="5dp"
        />
    <ListView android:id="@+id/paired_devices"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:stackFromBottom="false"
        android:layout_weight="1"
        />

    <TextView
        android:id="@+id/tvnombre_edad"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="TextView"
        android:textColor="@android:color/background_light"/>

    <TextView
        android:id="@+id/connecting"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge" />

    <TextView
        android:id="@+id/infoText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:text="Si no aparece ningún dispositivo, empareja tu
dispositivo en la configuración de Android"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textSize="16sp" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center">

    </LinearLayout>

</LinearLayout>
```

ANEXO I

Código de programación Pantalla principal Android Studio

```
package com.example.albertoruiz.babyhealth;

import android.bluetooth.BluetoothAdapter;
import android.content.Intent;
import android.os.Bundle;
import android.support.design.widget.TextInputEditText;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.toolbox.Volley;

import org.json.JSONException;
import org.json.JSONObject;

public class MainActivity extends AppCompatActivity {

    // Depuración de LOGCAT
    private static final String TAG = "MainActivity";

    // String que se enviara a la actividad principal, mainactivity
    public static String EXTRA_DEVICE_ADDRESS = "device_address";

    // Declaracion de campos
    private BluetoothAdapter mBtAdapter;
    private ArrayAdapter mPairedDevicesArrayAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        final EditText etusuario = (EditText) findViewById(R.id.etusuario);
        final TextInputEditText etpassword = (TextInputEditText)
```

```
findViewById(R.id.campo_password);
        final Button btn_Ingresar = (Button)
findViewById(R.id.btn_Ingresar);
        final TextView TvRegister = (TextView) findViewById(R.id.TvRegiser);

        TvRegister.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent TvRegisterIntent = new Intent(MainActivity.this,
Registro.class);
                MainActivity.this.startActivity(TvRegisterIntent);
            }
        });

        btn_Ingresar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                final String usuario = etusuario.getText().toString();
                final String password = etpassword.getText().toString();

                Response.Listener<String> responseListener = new
Response.Listener<String>() {
                    @Override
                    public void onResponse(String response) {

                        try {

                            JSONObject jsonResponse = new
JSONObject(response);
                            boolean success =
jsonResponse.getBoolean("success");

                            if (success){

                                String nombre =
jsonResponse.getString("nombre");
                                int edad = jsonResponse.getInt("edad");

                                Intent intent = new
Intent(MainActivity.this, DeviceListActivity.class);
                                intent.putExtra("nombre", nombre);
                                intent.putExtra("usuario", usuario);
                                intent.putExtra("edad", edad);
                                MainActivity.this.startActivity(intent);

                            }else{

                                AlertDialog.Builder builder = new
AlertDialog.Builder(MainActivity.this);
                                builder.setMessage("Login Failed")
                                        .setNegativeButton("Retry", null)
                                        .create()
                                        .show();

                            }
                        } catch (JSONException e) {
                            e.printStackTrace();
```

```
                        }

                    }
                };

                LoginRequest loginRequest = new LoginRequest(usuario,
password, responseListener);
                RequestQueue queue =
Volley.newRequestQueue(MainActivity.this);
                queue.add(loginRequest);
            }
        });

    }

    @Override
    public void onResume() {
        super.onResume();
        //----------------------------------
        VerificarEstadoBT();
    }

    private void VerificarEstadoBT() {
        // Comprueba que el dispositivo tiene Bluetooth y que está
encendido.
        mBtAdapter= BluetoothAdapter.getDefaultAdapter();
        if(mBtAdapter==null) {
            Toast.makeText(getBaseContext(), "El dispositivo no soporta
Bluetooth", Toast.LENGTH_SHORT).show();
        } else {
            if (mBtAdapter.isEnabled()) {
                Log.d(TAG, "...Bluetooth Activado...");
            } else {
                //Solicita al usuario que active Bluetooth
                Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(enableBtIntent, 1);

            }
        }
    }
}
```

ANEXO J

Código de programación Pantalla Registro Android Studio

```
package com.example.albertoruiz.babyhealth;
import android.content.Intent;
import android.os.Bundle;
import android.support.design.widget.TextInputEditText;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.toolbox.Volley;
import org.json.JSONException;
import org.json.JSONObject;

public class Registro extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_registro);

        final EditText tvnombre = (EditText) findViewById(R.id.tvnombre);
        final EditText tvapellido = (EditText)
findViewById(R.id.tvapellido);
        final EditText tvusuario = (EditText) findViewById(R.id.tvusuario);
        final TextInputEditText tvpassword = (TextInputEditText)
findViewById(R.id.cam_password);
        final EditText tvtelefono = (EditText)
findViewById(R.id.tvtelefono);
        final EditText tvfechan = (EditText) findViewById(R.id.tvfechan);
        final Button btn_registro = (Button)
findViewById(R.id.btn_registro);
        final TextView campovacio = (TextView) findViewById
(R.id.TvcamposVacios);

                btn_registro.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                if (tvnombre.getText ().toString ().isEmpty
```

```
()||tvapellido.getText ().toString ().isEmpty ()||tvusuario.getText
().toString ().isEmpty ()||tvpassword.getText ().toString ().isEmpty
()||tvtelefono.getText ().toString ().isEmpty ()||tvfechan.getText
().toString ().isEmpty ()){
                    campovacio.setText ("Uno o varios datos vacios");
                } else {

                    final String nombre = tvnombre.getText().toString();
                    final String apellido = tvapellido.getText().toString();
                    final String usuario = tvusuario.getText().toString();
                    final String password = tvpassword.getText().toString();
                    final int telefono =
Integer.parseInt(tvtelefono.getText().toString());
                    final int edad =
Integer.parseInt(tvfechan.getText().toString());
                    Response.Listener<String> responseListener = new
Response.Listener<String>(){

                        @Override
                        public void onResponse(String response) {
                            try {
                                JSONObject jsonResponse = new
JSONObject(response);
                                boolean success =
jsonResponse.getBoolean("success");
                                if (success){
                                    Intent intent = new
Intent(Registro.this, MainActivity.class);
                                    Registro.this.startActivity(intent);
                                }else{
                                    AlertDialog.Builder builder = new
AlertDialog.Builder(Registro.this);
                                    builder.setMessage("Register Failed user
same")
                                            .setNegativeButton("Retry",
null)
                                            .create()
                                            .show();
                                }
                            } catch (JSONException e) {
                                e.printStackTrace();
                            }
                        }
                    };

                    RegisterRequest registerRequest = new
RegisterRequest(nombre, apellido, usuario, password, telefono, edad,
responseListener);
                    RequestQueue queue =
Volley.newRequestQueue(Registro.this);
                    queue.add(registerRequest);

                }
            }
        });
    }
}
```

ANEXO K

Código de programación Pantalla Visualización Android Studio

```
package com.example.albertoruiz.babyhealth;

import android.app.Notification;
import android.app.NotificationManager;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.icu.text.SimpleDateFormat;
import android.media.RingtoneManager;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.Vibrator;
import android.support.annotation.RequiresApi;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.app.NotificationCompat;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

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

import static com.example.albertoruiz.babyhealth.R.id.btn_registro;

public class Visualizacion extends AppCompatActivity {

    TextView txtString, txtStringLength, spo, hr, alerta, mubicado,
porcentbat;
    Handler bluetoothIn;

    final int handlerState = 0; //used to identify handler message

    private BluetoothAdapter btAdapter = null;
```

```
    private BluetoothSocket btSocket = null;
    private StringBuilder recDataString = new StringBuilder();

    private ConnectedThread mConnectedThread;

    // SPP UUID service - this should work for most devices
    private static final UUID BTMODULEUUID = UUID.fromString("00001101-0000-
1000-8000-00805F9B34FB");

    // String for MAC address
    private static String address;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_visualizacion);

        final Vibrator vibracion = (Vibrator) this.getSystemService
(Context.VIBRATOR_SERVICE);


        txtString = (TextView) findViewById(R.id.txtString);
        txtStringLength = (TextView) findViewById(R.id.testView1);
        spo = (TextView) findViewById(R.id.tvspo);
        hr = (TextView) findViewById(R.id.tvhr);
        alerta = (TextView) findViewById(R.id.tvAlerta);
        mubicado = (TextView) findViewById (R.id.tvmubi);
        porcentbat = (TextView) findViewById (R.id.tvporcentBat);

        bluetoothIn = new Handler() {
            public void handleMessage(android.os.Message msg) {
                if (msg.what == handlerState) {
//if message is what we want
                    String readMessage = (String) msg.obj;
// msg.arg1 = bytes from connect thread
                    recDataString.append(readMessage);
//keep appending to string until ~
                    int endOfLineIndex = recDataString.indexOf("F");
// determine the end-of-line
                    int endHearthRate = recDataString.indexOf("S");
                    if (endOfLineIndex > 0) {
// make sure there data before ~
                        String dataInPrint = recDataString.substring(0,
endOfLineIndex);    // extract string
                        txtString.setText("Data Received = " + dataInPrint);
                        int dataLength = dataInPrint.length();
//get length of data received
                        txtStringLength.setText("String Length = " +
String.valueOf(dataLength));
                        if (endHearthRate <= 'F'){

                            if (recDataString.charAt(0) == 'H' &&
recDataString.charAt(4) == 'S' && recDataString.charAt(1) != '0')
//if it starts with # we know it is what we are looking for
                            {
                                String HR = recDataString.substring(1,4);
                                String SpO = recDataString.substring(5,8);
                                //String Bateria =
```

```
recDataString.substring(10,13);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("");
                                mubicado.setText("");
                                porcentbat.setText("80 %");

                            }

                            if (recDataString.charAt(1) == '0' &&
recDataString.charAt(4) == 'S' && recDataString.charAt(5) != '0'){

                                String HR = recDataString.substring(2,4);
                                String SpO = recDataString.substring(5,8);
                                //String Bateria =
recDataString.substring(10,13);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("");
                                mubicado.setText("");
                                porcentbat.setText("80 %");
                            }

                            if (recDataString.charAt(1) == '0' &&
recDataString.charAt(5) == '0'){

                                String HR = recDataString.substring(2,4);
                                String SpO = recDataString.substring(6,8);
                                //String Bateria =
recDataString.substring(10,13);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("");
                                mubicado.setText("");
                                porcentbat.setText("80 %");
                            }

                            if (recDataString.charAt(0) == 'H' &&
recDataString.charAt(5) == '0' && recDataString.charAt(1) != '0'){

                                String HR = recDataString.substring(1,4);
                                String SpO = recDataString.substring(6,8);
                                //String Bateria =
recDataString.substring(10,13);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("");
                                mubicado.setText("");
                                porcentbat.setText("80 %");
                            }

                            if (recDataString.charAt(0) == 'H' &&
recDataString.charAt(3) == 'S'){
                                String HR = recDataString.substring(1,3);
```

```
                                String SpO = recDataString.substring(4,7);
                                //String Bateria =
recDataString.substring(9,13);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("");
                                mubicado.setText("");
                                porcentbat.setText("80 %");
                            }

                            if (recDataString.charAt(1) == '-' &&
recDataString.charAt(6) == '-'){
                                String HR = recDataString.substring(1,4);
                                String SpO = recDataString.substring(6,9);
                                //String Bateria =
recDataString.substring(11,14);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("");
                                mubicado.setText("");
                                porcentbat.setText("80 %");

                            }

                            //condiciones cuando llega alerta

                            if (recDataString.charAt(0) == '1' &&
recDataString.charAt(1) == 'H' && recDataString.charAt(2) != '0' &&
recDataString.charAt(5) == 'S' && recDataString.charAt(6) != '0'){
                                String HR = recDataString.substring(2,5);
                                String SpO = recDataString.substring(6,9);
                                //String Bateria =
recDataString.substring(11,14);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("Alerta");
                                mubicado.setText("");
                                porcentbat.setText("80 %");

                                NotificationCompat.Builder notificacion =
new NotificationCompat.Builder (Visualizacion.this);

                                notificacion.setSmallIcon
(R.drawable.alerta);
                                notificacion.setTicker ("Alerta Bebe");
                                notificacion.setWhen
(System.currentTimeMillis ());
                                notificacion.setContentTitle ("Alerta");
                                notificacion.setContentText ("Revisar
Constantes Vitales");
                                notificacion.setContentInfo ("FC - SPO2");

                                Uri sonido = RingtoneManager.getDefaultUri
(Notification.DEFAULT_SOUND);
                                notificacion.setSound (sonido);
```

```
                                vibracion.vibrate (3000);

                                Bitmap icono = BitmapFactory.decodeResource
(getResources (),R.drawable.bebealerta);
                                notificacion.setLargeIcon (icono);

                                Notification n = notificacion.build ();
                                NotificationManager nm=
(NotificationManager) getSystemService (NOTIFICATION_SERVICE);

                                nm.notify (1, n);

                            }

                            if (recDataString.charAt(0) == '1' &&
recDataString.charAt(1) == 'H' && recDataString.charAt(2) == '0' &&
recDataString.charAt(5) == 'S' && recDataString.charAt(6) != '0'){
                                String HR = recDataString.substring(3,5);
                                String SpO = recDataString.substring(6,9);
                                //String Bateria =
recDataString.substring(11,14);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("Alerta");
                                mubicado.setText("");
                                porcentbat.setText("80 %");

                                NotificationCompat.Builder notificacion =
new NotificationCompat.Builder (Visualizacion.this);

                                notificacion.setSmallIcon
(R.drawable.alerta);
                                notificacion.setTicker ("Alerta Bebe");
                                notificacion.setWhen
(System.currentTimeMillis ());
                                notificacion.setContentTitle ("Alerta");
                                notificacion.setContentText ("Revisar
Constantes Vitales");
                                notificacion.setContentInfo ("FC - SPO2");

                                Uri sonido = RingtoneManager.getDefaultUri
(Notification.DEFAULT_SOUND);
                                notificacion.setSound (sonido);

                                vibracion.vibrate (3000);

                                Bitmap icono = BitmapFactory.decodeResource
(getResources (),R.drawable.bebealerta);
                                notificacion.setLargeIcon (icono);

                                Notification n = notificacion.build ();
                                NotificationManager nm=
(NotificationManager) getSystemService (NOTIFICATION_SERVICE);

                                nm.notify (1, n);
```

```
                            }

                            if (recDataString.charAt(0) == '1' &&
recDataString.charAt(1) == 'H' && recDataString.charAt(2) == '0' &&
recDataString.charAt(5) == 'S' && recDataString.charAt(6) == '0'){
                                String HR = recDataString.substring(3,5);
                                String SpO = recDataString.substring(7,9);
                                //String Bateria =
recDataString.substring(11,14);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("Alerta");
                                mubicado.setText("");
                                porcentbat.setText("80 %");

                                NotificationCompat.Builder notificacion =
new NotificationCompat.Builder (Visualizacion.this);

                                notificacion.setSmallIcon
(R.drawable.alerta);
                                notificacion.setTicker ("Alerta Bebe");
                                notificacion.setWhen
(System.currentTimeMillis ());
                                notificacion.setContentTitle ("Alerta");
                                notificacion.setContentText ("Revisar
Constantes Vitales");
                                notificacion.setContentInfo ("FC - SPO2");

                                Uri sonido = RingtoneManager.getDefaultUri
(Notification.DEFAULT_SOUND);
                                notificacion.setSound (sonido);

                                vibracion.vibrate (3000);

                                Bitmap icono = BitmapFactory.decodeResource
(getResources (),R.drawable.bebealerta);
                                notificacion.setLargeIcon (icono);

                                Notification n = notificacion.build ();
                                NotificationManager nm=
(NotificationManager) getSystemService (NOTIFICATION_SERVICE);

                                nm.notify (1, n);

                            }

                            if (recDataString.charAt(0) == '1' &&
recDataString.charAt(1) == 'H' && recDataString.charAt(2) != '0' &&
recDataString.charAt(5) == 'S' && recDataString.charAt(6) == '0'){
                                String HR = recDataString.substring(2,5);
                                String SpO = recDataString.substring(7,9);
                                //String Bateria =
recDataString.substring(11,14);

                                hr.setText(HR);
                                spo.setText(SpO + "%");
                                alerta.setText ("Alerta");
```

```
                                mubicado.setText("");
                                porcentbat.setText("80 %");

                                NotificationCompat.Builder notificacion =
new NotificationCompat.Builder (Visualizacion.this);

                                notificacion.setSmallIcon
(R.drawable.alerta);
                                notificacion.setTicker ("Alerta Bebe");
                                notificacion.setWhen
(System.currentTimeMillis ());
                                notificacion.setContentTitle ("Alerta");
                                notificacion.setContentText ("Revisar
Constantes Vitales");
                                notificacion.setContentInfo ("FC - SPO2");

                                Uri sonido = RingtoneManager.getDefaultUri
(Notification.DEFAULT_SOUND);
                                notificacion.setSound (sonido);

                                vibracion.vibrate (3000);

                                Bitmap icono = BitmapFactory.decodeResource
(getResources (),R.drawable.bebealerta);
                                notificacion.setLargeIcon (icono);

                                Notification n = notificacion.build ();
                                NotificationManager nm=
(NotificationManager) getSystemService (NOTIFICATION_SERVICE);

                                nm.notify (1, n);

                            }

                            // condicion cuando no existe dedo
                            if(recDataString.charAt(0) == 'N' &&
recDataString.charAt(1) == 'D'){

                                //String Bateria =
recDataString.substring(3,6);

                                //String ND = recDataString.substring(0,2);
                                hr.setText("ppm");
                                spo.setText("%");
                                mubicado.setText("Mal Ubicado");
                                alerta.setText ("");
                                porcentbat.setText("80 %");

                            }

                        }

                        recDataString.delete(0, recDataString.length());
//clear all string data
                        // strIncom =" ";
                        dataInPrint = " ";
                    }
                }
```

```
            }
        };

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

        botonregistro ();

        visualizacionTiempo ();

        recibirDato ();
    }

    public void botonregistro(){
        final Button btn_Registro = (Button) findViewById(btn_registro);
        btn_Registro.setOnClickListener (new View.OnClickListener () {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(Visualizacion.this,
Registro_info.class);
                Visualizacion.this.startActivity(intent);
            }
        });
    }

    public void visualizacionTiempo(){
        //Visualización tiempo

        Thread time = new Thread(){
            @Override
            public void run(){
                try{
                    while(!interrupted()){
                        Thread.sleep(1000);
                        runOnUiThread(new Runnable() {
                            @RequiresApi(api = Build.VERSION_CODES.N)
                            @Override
                            public void run() {
                                TextView tdate = (TextView)
findViewById(R.id.tvdate);
                                long date = System.currentTimeMillis();
                                SimpleDateFormat fecha = new
SimpleDateFormat("MMM dd yyyy\nhh:mm:ss a");
                                String dateString = fecha.format(date);
                                tdate.setText(dateString);
                            }
                        });
                    }
                }catch (InterruptedException e) {
                }
            }
        };
        time.start();
    }

    public void recibirDato(){
```

```
        Bundle extras = getIntent ().getExtras ();
        String nombre_usuario = extras.getString("dato01");
        int edad_usuario = extras.getInt("dato02");

       final TextView etWelcomemessage = (TextView)
findViewById(R.id.tvWelcome);

        String message = nombre_usuario + " Bienvenido " + "Edad: " +
edad_usuario + "m";
        etWelcomemessage.setText(message);

    }

    private BluetoothSocket createBluetoothSocket(BluetoothDevice device)
throws IOException {

        return  device.createRfcommSocketToServiceRecord(BTMODULEUUID);
        //creates secure outgoing connecetion with BT device using UUID
    }

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

        //Get MAC address from DeviceListActivity via intent
        Intent intent = getIntent();

        //Get the MAC address from the DeviceListActivty via EXTRA
        address =
intent.getStringExtra(DeviceListActivity.EXTRA_DEVICE_ADDRESS);

        //create device and set the MAC address
        BluetoothDevice device = btAdapter.getRemoteDevice(address);

        try {
            btSocket = createBluetoothSocket(device);
        } catch (IOException e) {
            Toast.makeText(getBaseContext(), "Socket creation failed",
Toast.LENGTH_LONG).show();
        }
        // Establish the Bluetooth socket connection.
        try
        {
            btSocket.connect();
        } catch (IOException e) {
            try
            {
                btSocket.close();
            } catch (IOException e2)
            {
                //insert code to deal with this
            }
        }
        mConnectedThread = new ConnectedThread(btSocket);
        mConnectedThread.start();

        //I send a character when resuming.beginning transmission to check
device is connected
```

```
        //If it is not an exception will be thrown in the write method and
finish() will be called
        mConnectedThread.write("x");
    }

    @Override
    public void onPause()
    {
        super.onPause();
        try
        {
            //Don't leave Bluetooth sockets open when leaving activity
            btSocket.close();
        } catch (IOException e2) {
            //insert code to deal with this
        }
    }

    //Checks that the Android device Bluetooth is available and prompts to
be turned on if off
    private void checkBTState() {

        if(btAdapter==null) {
            Toast.makeText(getBaseContext(), "Device does not support
bluetooth", Toast.LENGTH_LONG).show();
        } else {
            if (btAdapter.isEnabled()) {
            } else {
                Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(enableBtIntent, 1);
            }
        }
    }

    //create new class for connect thread
    private class ConnectedThread extends Thread {
        private final InputStream mmInStream;
        private final OutputStream mmOutStream;

        //creation of the connect thread
        public ConnectedThread(BluetoothSocket socket) {
            InputStream tmpIn = null;
            OutputStream tmpOut = null;

            try {
                //Create I/O streams for connection
                tmpIn = socket.getInputStream();
                tmpOut = socket.getOutputStream();
            } catch (IOException e) { }

            mmInStream = tmpIn;
            mmOutStream = tmpOut;
        }


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

```
            int bytes;

            // Keep looping to listen for received messages
            while (true) {
                try {
                    bytes = mmInStream.read(buffer);           //read bytes
from input buffer
                    String readMessage = new String(buffer, 0, bytes);
                    // Send the obtained bytes to the UI Activity via
handler
                    bluetoothIn.obtainMessage(handlerState, bytes, -1,
readMessage).sendToTarget();
                } catch (IOException e) {
                    break;
                }
            }
        }
        //write method
        public void write(String input) {
            byte[] msgBuffer = input.getBytes();           //converts
entered String into bytes
            try {
                mmOutStream.write(msgBuffer);                //write bytes
over BT connection via outstream
            } catch (IOException e) {
                //if you cannot write, close the application
                Toast.makeText(getBaseContext(), "Connection Failure",
Toast.LENGTH_LONG).show();
                finish();

            }
        }
    }
}
```

ANEXO L

Código de programación Login_Request Android Studio

```
package com.example.albertoruiz.babyhealth;

import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.toolbox.StringRequest;

import java.util.HashMap;
import java.util.Map;

/**
 * Created by albertoruiz on 22/11/17.
 */

public class LoginRequest extends StringRequest{

    //private static final String LOGIN_REQUEST_URL =
"http://192.168.0.109/prueba/Login.php";
    //private static final String LOGIN_REQUEST_URL =
"http://192.168.0.108/prueba/Login.php";
    //private static final String LOGIN_REQUEST_URL =
"http://192.168.0.104/prueba/Login.php";
    //private static final String LOGIN_REQUEST_URL =
"http://192.168.0.105/prueba/Login.php";
    private static final String LOGIN_REQUEST_URL =
"http://192.168.1.6/prueba/Login.php";
    private Map<String, String> params;

    public LoginRequest(String usuario, String password,
Response.Listener<String> listener){
        super (Request.Method.POST, LOGIN_REQUEST_URL, listener, null);
        params = new HashMap<>();
        params.put("usuario", usuario);
        params.put("password", password);
    }

    @Override
    public Map<String, String> getParams() {
        return params;
    }

}
```

ANEXO M

Código de programación Register_Request Android Studio

```
package com.example.albertoruiz.babyhealth;
import com.android.volley.Response;
import com.android.volley.toolbox.StringRequest;
import java.util.HashMap;
import java.util.Map;

/**
 * Created by albertoruiz on 20/11/17.
 */

public class RegisterRequest extends StringRequest {

    //private static final String REGISTER_REQUEST_URL =
"http://192.168.0.109/prueba/registro.php";
    //private static final String REGISTER_REQUEST_URL =
"http://192.168.0.108/prueba/registro.php";
    //private static final String REGISTER_REQUEST_URL =
"http://192.168.0.104/prueba/registro.php";
    //private static final String REGISTER_REQUEST_URL =
"http://192.168.0.105/prueba/registro.php";
    private static final String REGISTER_REQUEST_URL =
"http://192.168.1.6/prueba/registro.php";
    private Map<String, String> params;

    public RegisterRequest(String nombre, String apellido, String usuario,
String password, int telefono, int edad, Response.Listener<String>
listener){
        super (Method.POST, REGISTER_REQUEST_URL, listener, null);
        params = new HashMap<>();
        params.put("nombre", nombre);
        params.put("apellido", apellido);
        params.put("usuario", usuario);
        params.put("password", password);
        params.put("telefono", telefono + " ");
        params.put("edad", edad + " ");
    }

    @Override
    public Map<String, String> getParams() {
        return params;
    }
}
```

ANEXO N

Código de programación DeviceListActivity Android Studio

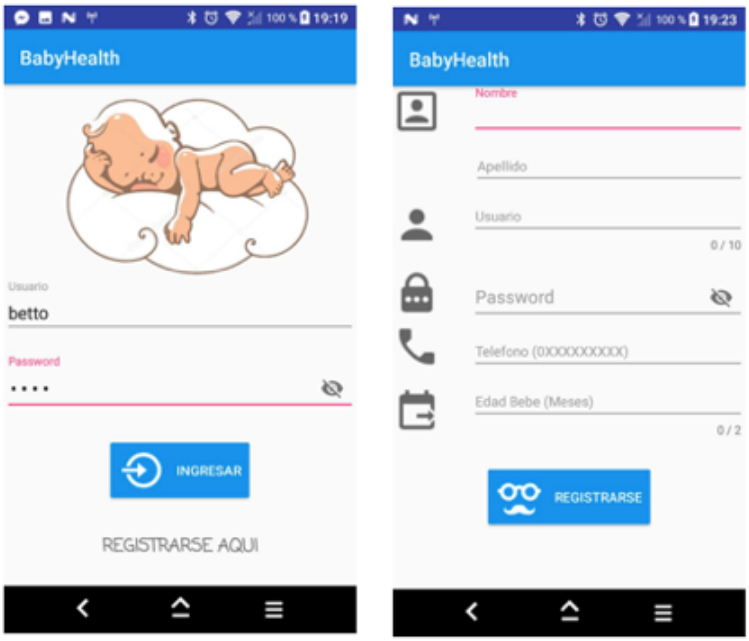
```
package com.example.albertoruiz.babyhealth;

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.Set;

public class DeviceListActivity extends Activity {
    // Debugging for LOGCAT
    private static final String TAG = "DeviceListActivity";
    private static final boolean D = true;


    // declare button for launching website and textview for connection
status
    Button tlbutton;
    TextView textView1;

    // EXTRA string to send on to mainactivity
    public static String EXTRA_DEVICE_ADDRESS = "device_address";

    // Member fields
    private BluetoothAdapter mBtAdapter;
    private ArrayAdapter<String> mPairedDevicesArrayAdapter;

    String nombre;
    int edad;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.device_list);

        Intent intent = getIntent();
        nombre = intent.getStringExtra("nombre");
        edad = intent.getIntExtra("edad", -1);

        TextView tvnombre_edad = (TextView)
findViewById(R.id.tvnombre_edad);

        // Display user details
        String message_nombre = nombre + " " + edad + "";
        tvnombre_edad.setText(message_nombre);

    }

    @Override
    public void onResume()
    {
        super.onResume();
        //***************
        checkBTState();

        textView1 = (TextView) findViewById(R.id.connecting);
        textView1.setTextSize(40);
        textView1.setText(" ");

        // Initialize array adapter for paired devices
        mPairedDevicesArrayAdapter = new ArrayAdapter<String>(this,
R.layout.device_name);

        // Find and set up the ListView for paired devices
        ListView pairedListView = (ListView)
findViewById(R.id.paired_devices);
        pairedListView.setAdapter(mPairedDevicesArrayAdapter);
        pairedListView.setOnItemClickListener(mDeviceClickListener);

        // Get the local Bluetooth adapter
        mBtAdapter = BluetoothAdapter.getDefaultAdapter();

        // Get a set of currently paired devices and append to
'pairedDevices'
        Set<BluetoothDevice> pairedDevices = mBtAdapter.getBondedDevices();

        // Add previosuly paired devices to the array
        if (pairedDevices.size() > 0) {

findViewById(R.id.title_paired_devices).setVisibility(View.VISIBLE);//make
title viewable
            for (BluetoothDevice device : pairedDevices) {
                mPairedDevicesArrayAdapter.add(device.getName() + "\n" +
device.getAddress());
            }
        } else {
            String noDevices =
getResources().getText(R.string.none_paired).toString();
            mPairedDevicesArrayAdapter.add(noDevices);
        }
```

```
    }


    // Set up on-click listener for the list (nicked this - unsure)
    private OnItemClickListener mDeviceClickListener = new
OnItemClickListener() {
        public void onItemClick(AdapterView<?> av, final View v, int arg2,
long arg3) {

            textView1.setText("Connecting...");
            // Get the device MAC address, which is the last 17 chars in the
View
            String info = ((TextView) v).getText().toString();
            String address = info.substring(info.length() - 17);

            // Make an intent to start next activity while taking an extra
which is the MAC address.
            Intent i = new Intent(DeviceListActivity.this,
Visualizacion.class);
            i.putExtra(EXTRA_DEVICE_ADDRESS, address);
            i.putExtra("dato01", nombre);
            i.putExtra("dato02", edad);
            startActivity(i);
        }
    };

    private void checkBTState() {
        // Check device has Bluetooth and that it is turned on
        mBtAdapter=BluetoothAdapter.getDefaultAdapter(); // CHECK THIS OUT
THAT IT WORKS!!!
        if(mBtAdapter==null) {
            Toast.makeText(getBaseContext(), "Device does not support
Bluetooth", Toast.LENGTH_SHORT).show();
        } else {
            if (mBtAdapter.isEnabled()) {
                Log.d(TAG, "...Bluetooth ON...");
            } else {
                //Prompt user to turn on Bluetooth
                Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(enableBtIntent, 1);

            }
        }
    }
}
```

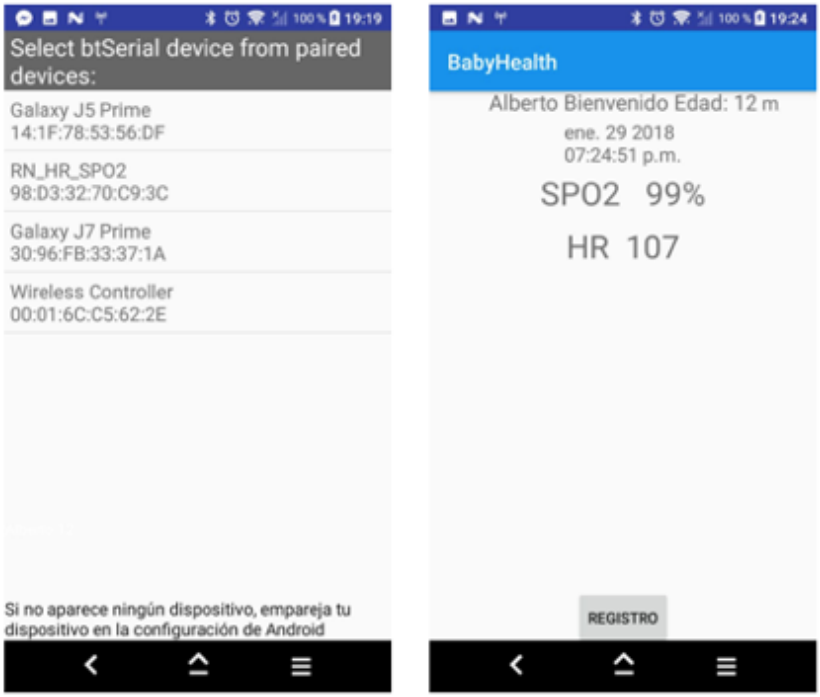
ANEXO O

Diseño de interfaz de Login y Registro



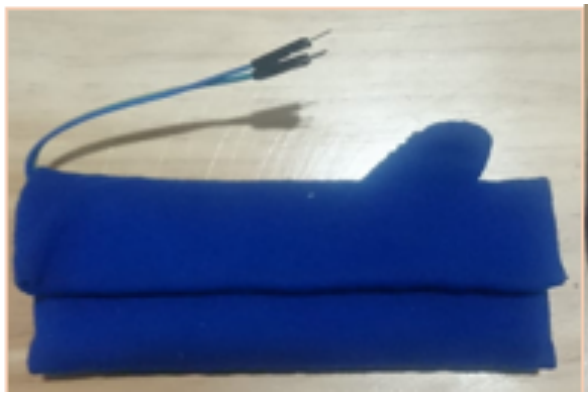
ANEXO P

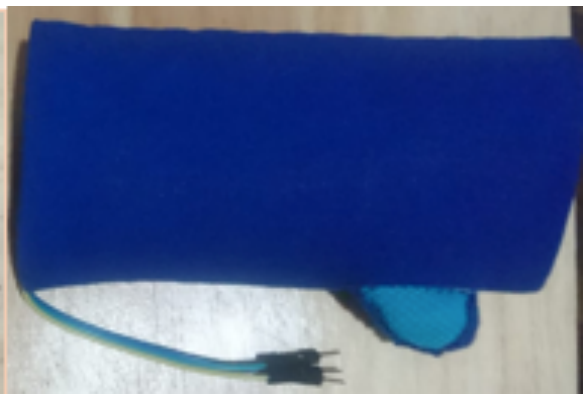
Diseño de interfaz de Emparejamiento y Visualización



ANEXO Q

Prototipo del Guante diferentes vistas





ANEXO R

Código PHP functions

```
<?php
header( 'Content-Type: text/html;charset=utf-8' );
function ejecutarSQLCommand($commando){
  $mysqli = new mysqli("localhost", "root", "Bettodany1712", "babyheath");
/* check connection */
if ($mysqli->connect_errno) {
    printf("Connect failed: %s\n", $mysqli->connect_error);
    exit();
}if ( $mysqli->multi_query($commando)) {
     if ($resultset = $mysqli->store_result()) {
    while ($row = $resultset->fetch_array(MYSQLI_BOTH)) {
    }
    $resultset->free();
     }
}
$mysqli->close();
}
function getSQLResultSet($commando){
  $mysqli = new mysqli("localhost", "root", "Bettodany1712", "babyheath");
/* check connection */
if ($mysqli->connect_errno) {
    printf("Connect failed: %s\n", $mysqli->connect_error);
    exit();
}if ( $mysqli->multi_query($commando)) {
    return $mysqli->store_result();
}
$mysqli->close();
}
?>
```

ANEXO S

Código PHP Login

```
<?php
   //include ('functions.php');
   $con = mysqli_connect("localhost", "root", "Bettodany1712", "babyheath");
   $usuario = $_POST["usuario"];
   $password = $_POST["password"];
   $statement = mysqli_prepare($con, "SELECT * FROM registro WHERE usuario =
'$usuario' AND password = '$password'");
   mysqli_stmt_bind_param($statement, 'ss', $usuario, $password);
   mysqli_stmt_execute($statement);
   mysqli_stmt_store_result($statement);
   mysqli_stmt_bind_result($statement, $user_id, $nombre, $apellido, $usuario,
$password, $telefono, $edad);
   $response = array();
   $response["success"] = false;
   while(mysqli_stmt_fetch($statement)){
      $response["success"] = true;
      $response["nombre"] = $nombre;
      $response["apellido"] = $apellido;
      $response["usuario"] = $usuario;
      $response["password"] = $password;
      $response["telefono"] = $telefono;
      $response["edad"] = $edad;
   }
   echo json_encode($response);
?>
```

ANEXO T

Código PHP Register

```
<?php include ('functions.php');
   $nombre=$_POST['nombre'];
   $apellido=$_POST['apellido'];
   $usuario=$_POST['usuario'];
   $password=$_POST['password'];
   $telefono=$_POST['telefono'];
   $edad=$_POST['edad'];
   ejecutarSQLCommand("INSERT INTO  `registro` (nombre, apellido, usuario, password, telefono, edad)
   VALUES (
   '$nombre',
   '$apellido',
   '$usuario',
   '$password',
   '$telefono',
   '$edad')
   ON DUPLICATE KEY UPDATE
   `nombre`= '$nombre',
   `apellido`='$apellido',
   `usuario`='$usuario',
   `password`='$password',
   `telefono`='$telefono',
   `edad`='$edad';");
   $response = array();
   $response["success"] = true;
   echo json_encode($response);
?>
```

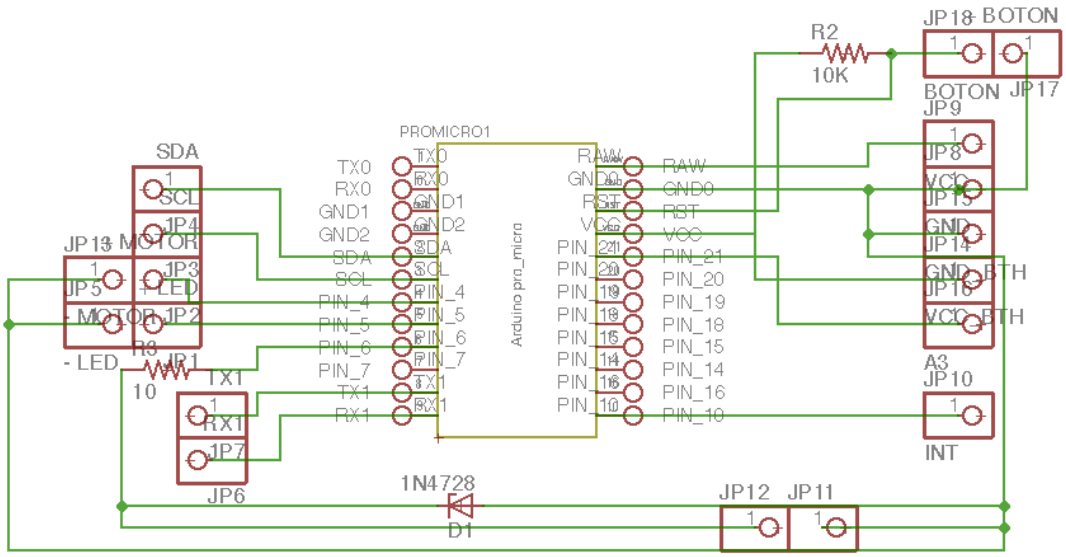
ANEXO U

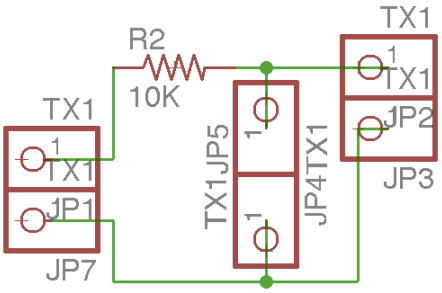
Código PHP Monitoreo

```
<?php include ('functions.php');
   $nombre=$_POST['nombre'];
   $edad=$_POST['edad'];
   $HR=$_POST['HR'];
   $SPO2=$_POST['SPO2'];
   $Alerta=$_POST['Alerta'];
   ejecutarSQLCommand("INSERT INTO  `monitoreo` (nombre, edad, HR, SPO2, Alerta)
   VALUES (
   '$nombre',
   '$edad',
   '$HR',
   '$SPO2',
   '$Alerta',
   ON DUPLICATE KEY UPDATE
   `nombre`= '$nombre',
   `edad`='$edad',
   `HR`='$HR',
   `SPO2`='$SPO2',
   `Alerta`='$Alerta';");
   $response = array();
   $response["success"] = true;
   echo json_encode($response);
?>
```

ANEXO V

Circuito Esquemático de las placas de Procesamiento y regulador de voltaje 3.3V





ANEXO W

Términos y condiciones Android Studio

Este es el Acuerdo de licencia del kit para desarrollo de software de Android
1. Introducción
1.1 Se le otorga licencia para el kit de desarrollo de software de Android (al que se hace referencia en el Acuerdo de licencia como "SDK" y que incluye específicamente los archivos de sistema de Android, las API incorporadas y los complementos de las API de Google) sujeto a las condiciones del Acuerdo de licencia. El Acuerdo de licencia es un contrato legalmente vinculante entre usted y Google en relación con el uso del SDK. 1.2 "Android" se refiere a la pila de software de Android para dispositivos, disponible mediante el Proyecto de código abierto de Android, que se encuentra en la siguiente URL: http://source.android.com/, según se actualice de manera periódica. 1.3 Una "implementación compatible" hace referencia a cualquier dispositivo Android que (i) cumpla con el documento de Definición de compatibilidad de Android, que puede encontrarse en el sitio web de compatibilidad de Android (http://source.android.com/compatibility) y que puede actualizarse de manera periódica; y que (ii) supere el conjunto de pruebas de compatibilidad de Android (CTS). 1.4 "Google" se refiere a Google Inc., una corporación de Delaware cuya oficina principal se encuentra en 1600 Amphitheatre Parkway, Mountain View, CA 94043, Estados Unidos.
2. Aceptación de este Acuerdo de licencia
2.1 Para usar el SDK, primero debe aceptar este Acuerdo de licencia. Si no lo hace, no podrá usar el SDK. 2.2 Al hacer clic en Aceptar, acepta las condiciones del Acuerdo de licencia. 2.3 Si se le prohibió recibir el SDK según las leyes de Estados Unidos o de otros países, incluido el país en el que reside o desde el que usa el SDK, no podrá usarlo ni aceptar el Acuerdo de licencia. 2.4 Si acepta quedar vinculado por este Acuerdo de licencia en nombre de su empleador o de otra entidad, usted manifiesta y garantiza que posee la capacidad legal absoluta para vincular a su empleador o a dicha entidad a este Acuerdo de licencia. Si no tiene la autoridad necesaria, no acepte el Acuerdo de licencia ni use el SDK en nombre de su empleador o de otra entidad.
3. Licencia de SDK de Google
3.1 Sujeto a las condiciones del Acuerdo de licencia, Google le otorga una licencia limitada, mundial, libre de derechos de autor, no cedible, no exclusiva y no susceptible de someterse a otras licencias para usar el SDK, únicamente con el fin de desarrollar aplicaciones para implementaciones compatibles de Android. 3.2 No puede usar este SDK para desarrollar aplicaciones en otras plataformas (incluidas las implementaciones no compatibles de Android) o para desarrollar otro SDK. Si lo desea, puede desarrollar aplicaciones para otras plataformas, incluidas las implementaciones no compatibles de Android, siempre y cuando no se use este SDK con tal fin. 3.3 Usted acepta que Google o terceros poseen el derecho legal, la propiedad y el interés totales relacionados con el SDK, incluidos los Derechos de propiedad intelectual que tenga el SDK. "Derechos de propiedad intelectual" implica todo derecho que existe según la ley de patentes, la ley de derechos de autor, la Ley de secreto comercial, la ley de marca comercial y cualquier otro derecho de propiedad. Google se reserva todos los derechos que no se le hayan otorgado expresamente a usted. 3.4 No podrá usar el SDK para ningún fin que no esté permitido expresamente en este Acuerdo de licencia. Excepto en la medida en que las licencias de terceros lo exijan, no podrá copiar (salvo con fines de copia de seguridad), modificar, adaptar, redistribuir, descompilar ni desmontar SDK o partes de este. Tampoco podrá aplicar ingeniería inversa ni crear trabajos derivados de este. 3.5 El uso, la reproducción y la distribución de componentes del SDK con licencia bajo una licencia de software de código abierto se rigen únicamente por las condiciones de dicha licencia de software de código abierto y no por el Acuerdo de licencia. 3.6 Usted acepta que la forma y la naturaleza del SDK que Google proporciona pueden cambiar sin previo aviso y que las versiones futuras del SDK podrían no ser compatibles con aplicaciones desarrolladas en versiones anteriores del SDK. Usted acepta que Google puede dejar de proporcionarle a usted o a los usuarios en general (en forma permanente o temporal) el SDK (o cualquiera de sus características) cuando Google lo considere oportuno, sin previo aviso. 3.7 Ninguna de las disposiciones en este Acuerdo de licencia le otorga el derecho de usar los nombres comerciales, las marcas comerciales, las marcas comerciales del servicio, los logotipos, los nombres del dominio u otras características de marca distintivas de Google. 3.8 Usted acepta no quitar, bloquear ni alterar ningún aviso de derecho de propiedad (incluidos los avisos de derechos de autor y marcas comerciales) que podrían adjuntarse o incluirse en el SDK.
4. Su uso del SDK
4.1 Google acepta que no obtiene ningún derecho, título ni interés de parte suya (o de sus proveedores de licencias) en virtud de este Acuerdo de licencia en relación con cualquiera de las aplicaciones de software que desarrolle usando el SDK, incluido cualquier derecho de propiedad intelectual que tengan esas aplicaciones. 4.2 Usted se compromete a utilizar el SDK y crear aplicaciones únicamente para los fines permitidos por (a) este Acuerdo de licencia y (b) cualquier ley aplicable, normativa, prácticas o lineamientos generalmente aceptados en las jurisdicciones pertinentes (incluidas todas las leyes relacionadas con la exportación de datos o software hacia y desde los Estados Unidos y demás países pertinentes). 4.3 Usted acepta que si usa el SDK para desarrollar aplicaciones para el público general, protegerá la privacidad y los derechos legales de dichos usuarios. Si los usuarios le proporcionan nombres de usuario, contraseñas u otra información personal o de acceso, usted debe advertirles que la información estará disponible para su aplicación y debe proporcionarles avisos de privacidad y protección legalmente apropiados. Si su aplicación almacena información confidencial o personal proporcionada por los usuarios, deberá hacerlo de forma segura. Si el usuario le proporciona información de la cuenta de Google a su aplicación, solo podrá usar esa información para acceder a la cuenta de Google del usuario en el momento y para los fines que el usuario le haya dado permiso para hacerlo. 4.4 Usted acepta que no se involucrará en ninguna actividad con el SDK, incluido el desarrollo o la distribución de una aplicación, que interfiera, afecte, dañe o permita el acceso no autorizado a los servidores, redes o a otras propiedades o servicios de terceros que incluyan, entre otros, a Google o a cualquier proveedor de comunicaciones móviles. 4.5 Usted acepta que es el único responsable (y que Google no tiene ninguna responsabilidad ante usted ni ante terceros) de cualquier dato, contenido o recurso que usted cree, transmita o muestre mediante Android o aplicaciones para Android, y de las consecuencias de sus acciones (inclusive cualquier pérdida o daño que Google pueda sufrir). 4.7 Usted acepta que es el único responsable de cualquier incumplimiento de las obligaciones establecidas en este Acuerdo de licencia, de las Condiciones del servicio o el contrato con un tercero aplicables, o de cualquier normativa o ley aplicables, así como de las consecuencias derivadas de su incumplimiento, incluidos cualquier pérdida o daño que pueda sufrir Google o un tercero; y que Google no tiene ninguna responsabilidad ante usted ni ante terceros relacionada con el incumplimiento.
5. Sus credenciales de programador

5.1 Usted acepta que será responsable de mantener la confidencialidad de las credenciales de programador que Google pueda haberle proporcionado o que usted mismo haya elegido, y que será el único responsable de todas las aplicaciones que se desarrollen con sus credenciales de programador.
6. Información y privacidad
6.1 Con el fin de innovar y mejorar continuamente el SDK, Google puede recopilar ciertas estadísticas de uso del software, que incluyen, entre otras, un identificador único, la dirección IP asociada, el número de versión de software y la información acerca de qué herramientas o servicios del SDK se están usando y cómo se están usando. Antes de que se recopile la información, el SDK se lo notificará y pedirá su consentimiento. Si no otorga el consentimiento, no se recopilará la información. 6.2 Los datos recopilados se examinan en conjunto para mejorar el SDK y se conservan de acuerdo con la Política de privacidad de Google.
7. Aplicaciones de terceros
7.1 Si usa el SDK para ejecutar aplicaciones desarrolladas por un tercero o que acceden a datos, contenido o recursos proporcionados por un tercero, usted acepta que Google no es responsable por dichas aplicaciones, datos, contenido o recursos. Usted comprende que todos los datos, contenido o recursos a los que pueda acceder mediante dicha aplicación de terceros son responsabilidad exclusiva de la persona de la cual se originaron y que Google no es responsable de ninguna pérdida ni daño que usted pueda experimentar como resultado del uso de dichas aplicaciones, datos, contenido o recursos de terceros o del acceso a ellos. 7.2 Debe tener en cuenta que los datos, el contenido y los recursos que recibe mediante dicha aplicación de terceros pueden estar protegidos por derechos de propiedad intelectual que son propiedad de sus proveedores (o de otras personas o empresas en su nombre). Usted no puede modificar, alquilar, arrendar, prestar, vender, distribuir ni crear trabajos derivados basados en estos datos, contenido o recursos (la totalidad o parte de ellos), a menos que los propietarios le hayan otorgado permiso específicamente para hacerlo. 7.3 Usted reconoce que el uso de dichos datos, contenido, recursos o aplicaciones de terceros puede estar sujeto a condiciones separadas entre usted y el tercero correspondiente. En ese caso, el Acuerdo de licencia no afecta su relación legal con terceros.
8. Uso de las API de Android
8.1 API de datos de Google 8.1.1 Si usted usa una API para recuperar datos de Google, usted reconoce que los datos pueden estar protegidos por derechos de propiedad intelectual que son propiedad de Google o de las partes que proporcionan los datos (u otras personas o empresas en su nombre). El uso de la API puede estar sujeto a Condiciones del servicio adicionales. Usted no puede modificar, alquilar, arrendar, prestar, vender, distribuir ni crear trabajos derivados basados en estos datos (la totalidad o parte de ellos), a menos que las Condiciones del servicio pertinentes lo permitan. 8.1.2 Si usa una API para recuperar los datos de un usuario de Google, usted reconoce y acepta que puede recuperar datos solamente con el consentimiento explícito del usuario en el momento y para los fines que el usuario le haya dado permiso para hacerlo.
9. Rescisión de este Acuerdo de licencia
9.1 El Acuerdo de licencia continuará vigente hasta que usted o Google decidan su rescisión, como se establece a continuación. 9.2 Si desea resolver el Acuerdo de licencia, podrá hacerlo mediante la cesación del uso del SDK y cualquiera de las credenciales de programador pertinentes. 9.3 En los siguientes casos, Google podrá resolver el Acuerdo de licencia en cualquier momento: (A) Usted no cumplió con alguna disposición del Acuerdo de licencia. (B) La ley exige que Google lo haga. (C) El socio con el que Google le ofreció algunas partes del SDK (como las API) resolvió su relación con Google o dejó de proporcionarle a usted determinadas partes del SDK. (D) Google decide dejar de proporcionar el SDK o determinadas partes del SDK a usuarios del país en el que usted reside o desde el que usa el servicio; o el aprovisionamiento del SDK o determinados servicios del SDK que Google brinda ya no son comercialmente viables, según Google. 9.4 Cuando el Acuerdo de licencia finaliza, todos los derechos legales, obligaciones y responsabilidades de los que usted y Google se hayan beneficiado, a los que hayan estado sujetos (o que se hayan aplicado durante el tiempo de vigencia del Acuerdo de licencia) o que se hayan expresado como continuos de manera indefinida, permanecerán sin cambios ante este cese. Las disposiciones del párrafo 14.7 se seguirán aplicando a dichos derechos, obligaciones y responsabilidades indefinidamente.
10. RENUNCIA DE GARANTÍAS
10.1 USTED COMPRENDE Y ACEPTA EXPRESAMENTE CUALQUIER RIESGO QUE EL USO DEL SDK PUEDA IMPLICAR Y QUE EL SDK SE PROPORCIONA "COMO ES" Y "ESTÁ DISPONIBLE" SIN NINGUNA GARANTÍA DE GOOGLE. 10.2 SU USO DEL SDK Y DE CUALQUIER MATERIAL QUE SE DESCARGUE O SE OBTENGA MEDIANTE EL USO DEL SDK QUEDA A SU TOTAL DISCRECIÓN. USTED SERÁ EL ÚNICO RESPONSABLE POR CUALQUIER DAÑO QUE SE PRODUZCA EN SU SISTEMA INFORMÁTICO O EN OTRO DISPOSITIVO, O POR CUALQUIER PÉRDIDA DE DATOS QUE RESULTE DE TAL USO. 10.3 GOOGLE EXPRESA SU RENUNCIA DE RESPONSABILIDAD CON RESPECTO A GARANTÍAS Y CONDICIONES DE CUALQUIER TIPO, YA SEA EXPRESA O IMPLÍCITA, Y QUE INCLUYE, ENTRE OTRAS, LAS GARANTÍAS Y CONDICIONES IMPLÍCITAS DE COMERCIALIZACIÓN, LA ADECUACIÓN PARA UN PROPÓSITO EN PARTICULAR Y LA NO VIOLACIÓN.
11. LIMITACIÓN DE RESPONSABILIDADES
11.1 USTED COMPRENDE Y ACEPTA EXPRESAMENTE QUE GOOGLE, SUS SUBSIDIARIAS Y AFILIADAS, Y SUS PROVEEDORES DE LICENCIAS NO SE RESPONSABILIZARÁN ANTE USTED BAJO NINGUNA TEORÍA DE RESPONSABILIDAD POR DAÑOS DIRECTOS, INDIRECTOS, IMPREVISTOS, ESPECIALES, DERIVADOS O EJEMPLARES EN LOS QUE USTED HAYA INCURRIDO, INCLUIDA LA PÉRDIDA DE DATOS, YA SEA QUE GOOGLE O SUS REPRESENTANTES HAYAN SIDO INFORMADOS O HAYAN TOMADO CONOCIMIENTO DE LA POSIBILIDAD DE QUE SE PRODUZCAN DICHAS PÉRDIDAS.
12. Indemnización
12.1 Hasta el grado máximo que permita la ley, usted acepta defender, indemnizar y eximir de responsabilidad a Google, sus filiales y sus respectivos directores, oficiales, empleados y agentes de cualquier reclamo, acción, juicio o proceso judicial, así como también de toda pérdida, responsabilidad, daño, costo y gasto (incluidos honorarios razonables del abogado) que se incurra por (a) el uso del SDK, (b) cualquier aplicación que usted desarrolle en el SDK que constituya una infracción de cualquier derecho de autor, marca comercial, secreto comercial, imagen comercial, patente u otro derecho de propiedad intelectual de cualquier persona, o de la difamación a cualquier persona o la infracción de sus derechos de publicidad o privacidad, y (c) cualquier incumplimiento que usted haga del Acuerdo de licencia.
13. Cambios en el Acuerdo de licencia
13.1 Google podrá realizar cambios al Acuerdo de licencia cuando distribuya nuevas versiones del SDK. Una vez realizados los cambios, Google pondrá a disposición una nueva versión del Acuerdo de licencia en el sitio web donde está disponible el SDK.
14. Términos legales generales
14.1 El Acuerdo de licencia constituye la totalidad del acuerdo legal entre usted y Google y rige el uso del SDK (excepto los servicios que Google pueda proporcionarle a usted con un acuerdo separado por escrito) y reemplaza completamente cualquier acuerdo anterior relacionado con el SDK. 14.2 Usted acuerda que si Google no ejerce o hace cumplir un derecho legal o solución que se haya incluido en este Acuerdo de licencia (o del que Google se beneficie según lo establecido por cualquier ley aplicable), no se considerará una renuncia formal de los derechos de Google y que esos derechos o soluciones seguirán estando disponibles para Google. 14.3 Si algún tribunal, que tenga jurisdicción para decidir sobre este asunto, dictamina que alguna disposición de este Acuerdo de licencia no es válida, se quitará esa disposición sin afectar al resto del Acuerdo de licencia. Las disposiciones restantes del Acuerdo de licencia continuarán siendo válidas y aplicables. 14.4 Usted reconoce y acepta que cada miembro del grupo de empresas del cual Google es la casa matriz serán los terceros beneficiarios del Acuerdo de licencia y que esas empresas tendrán derecho a imponer y depender directamente de toda disposición del Acuerdo de licencia que conceda un beneficio para ellos (o un derecho a su favor). Excepto lo anteriormente estipulado, no habrá ninguna otra persona o empresa como terceros beneficiarios de este Acuerdo de licencia. 14.5 RESTRICCIONES SOBRE LA EXPORTACIÓN. EL SDK ESTÁ SUJETO A LAS LEYES Y NORMATIVAS DE EXPORTACIÓN DE ESTADOS UNIDOS. USTED DEBE CUMPLIR CON TODAS LAS LEYES Y NORMATIVAS NACIONALES E INTERNACIONALES DE EXPORTACIÓN QUE SE APLICAN AL SDK. ESTAS LEYES INCLUYEN LAS RESTRICCIONES ACERCA DE LOS DESTINOS, LOS USUARIOS FINALES Y EL USO FINAL. 14.6 Ni usted ni Google pueden asignar o transferir los derechos otorgados sin la aprobación previa por escrito de la otra parte. Ni usted ni Google podrán delegar sus responsabilidades u obligaciones del Acuerdo de licencia sin la aprobación previa por escrito de la otra parte. 14.7 El Acuerdo de licencia y la relación con Google que surge del Acuerdo de licencia se regirán por las leyes del estado de California, excepto el conflicto con las disposiciones legales. Usted y Google aceptan someterse a la jurisdicción exclusiva de los tribunales ubicados en el condado de Santa Clara, California, para que resuelvan todo problema legal que surja o esté relacionado con este Acuerdo de licencia. No obstante, usted acepta que Google aún podrá solicitar recursos judiciales (o una clase equivalente de compensación legal urgente) en cualquier jurisdicción. *9 de diciembre de 2016*

ANEXO X

Estándares de Apps Móviles

Ministerio de Modernización
Dirección Nacional de Servicios Digitales
Versión: 0.1
Índice:

Objetivo:
El presente documento tiene como objetivo definir las pautas elementales a ser consideradas al momento de diseñar, desarrollar e implementar toda aplicación móvil del Estado Nacional, ya sea realizada desde el Estado o por un proveedor externo, con el fin de ser utilizados por ciudadanos.
Con los siguientes estándares, se busca homogeneizar la experiencia entre los diversos activos del Estado, para facilitar la comprensión y utilización, primando la Usabilidad, la Accesibilidad y la Experiencia.
Requisitos
Desarrollo
Las aplicaciones deben soportar las distintas resoluciones de los distintos tipos de dispositivos.
El instalador de las aplicaciones no debe superar los 10 MB.
No embeber imágenes y videos como contenido estático que hagan que los instaladores sean más pesados.
Las aplicaciones que no tengan backend propio pueden utilizar el portal de argentina.gob.ar para consumir contenidos dinámicos, respetando los lineamientos del Estándar de API definida por la Coordinación de Desarrollo de Servicios Digitales.
La lista de aplicaciones de cada ministerio debe estar actualizada. Si la aplicación no está más en funcionamiento (o con soporte) hay que removerla de las tiendas.
El código fuente de las aplicaciones deberá ser simple, fácil de comprender, escalable, flexible y deberá ser acompañado de la documentación necesaria para poder asegurar su continuidad soportando un futuro cambio de proveedor.
Generar y proveer los manuales de uso necesarios para el ciudadano que utiliza la aplicación móvil.
Control de Versiones:
versión v 0.0.1: Resolución de bugs de versiones actuales.
versión v.0.1.0: Nueva funcionalidad dentro de la versión actual.
versión v.1.0.0: Representan un cambio sustancial respecto de la funcionalidad o de la estética actual del sitio.
Las aplicaciones con sistemas de registración deben seguir los lineamientos definidos por la Coordinación de Identidad digital y Tarjeta Inteligente de Servicios Digitales.
¿Por qué?
Brinda un sistema homogéneo de registración para los ciudadanos en todas las propiedades digitales del gobierno.
Todo contenido que muestra la aplicación debe ser consumido mediante una API respetando los lineamientos del Estándar de API definida por la Coordinación de Desarrollo de Servicios Digitales y el Estándar de Contenido definida por la Coordinación de Contenidos de Servicios Digitales.
¿Por qué?
Permite actualizar el contenido de forma dinámica sin necesidad de generar una nueva versión de la aplicación y homogeneizando un lenguaje común para todas las propiedades digitales del gobierno.
La aplicación debe brindar la posibilidad de operar en forma total o parcial en modo offline (sin conexión). La sincronización y actualización de contenidos se llevarán a cabo una vez que el dispositivo se encuentre nuevamente en modo online.
¿Por qué?
Evitar el tráfico de datos innecesarios para el ciudadano y brindar la posibilidad de usar la aplicación aún cuando no se cuenta con una conexión a Internet.
El desarrollo se realizará para que sea compatible con la última versión más estable del sistema operativo y contemplando las versiones previas que aún siguen siendo populares.
Diseño
Los nombres de las aplicaciones deberán contar con la aprobación de la Dirección Nacional de Servicios Digitales.
Los iconos lanzadores de las aplicaciones serán creados y provistos por la Coordinación de Diseño de Servicios Digitales.
¿Por qué?
Brinda un sistema visual homogéneo para todas las propiedades digitales del gobierno.
Colores:
La paleta de colores a utilizar es la definida en el manual de identidad visual para web y aplicaciones móviles. Los usos son los siguientes:
Primario: #0072BC
Es el color primario que se utiliza dentro de la UI en elementos como links, botones, etc.
Secundario: #00B9F1 Se utiliza para ciertos elementos del contenido que necesitan ser destacados, por ejemplo en iconos.
Complementario: #FD4138 Se utiliza para elementos que necesiten un destaque diferencial y en ciertos elementos para dar calidez en páginas muy extensas que no contengan fotografías.
Neutros:
Texto: #111111
Gris claro: #767676
Bordes y detalles: #CCCCCC
Fondo: #F5F5F5
Blanco: #FFFFFF
Tipografías:
La familia tipográfica estándar es Roboto dada su alta legibilidad en medios digitales.
Para textos largos se usa Droid Serif para cansar menos al lector, ya que por su forma las letras se identifican más fácil.
Ambas familias tipográficas están hechas por Google y su uso es libre y gratuito bajo licencia Apache 2.0.
Estética y elementos gráficos:
Como estética se usa un sistema visual minimalista "flat" en donde predomina el diseño del contenido evitando el uso de ornamentos que distraigan.
Los elementos se componen de figuras simples con colores plenos, evitando el uso de degradados.
Se recomienda el uso de fotografías como parte del contenido, ya que hace más amena la lectura y ayuda a la comunicación. Ver los Estándares de Contenido para más información.
Usabilidad
Diseñar para todas las resoluciones de pantalla de computadoras, tablets y celulares posibles.
Usar convenciones de estética y funcionamiento de cada plataforma o sistema operativo (Ej: Guía Google, Guía iOS, Guía BlackBerry, Guía

Windows Phone. Para sitios web se deben respetar sólo los lineamientos de identidad visual).
Entender qué funcionalidades son las más usadas (sobre la base de la medición de analíticas y pruebas de usabilidad) para hacerlas más fácil de acceder en futuras versiones.
Mostrar al usuario el estado del sistema en todo momento. Dónde está (títulos y breadcrumbs), lo que está haciendo y de que se trata (títulos y descripciones), cuál es su progreso (indicadores de carga y barras de progreso), etc).
Brindar una respuesta inmediata a cada interacción (un cambio visual, un mensaje de carga, etc).
Los elementos interactivos deben tener un estilo para cuando está:
En estado normal
En foco
Activo (cuando se está haciendo clic o tap)
Visitado (para enlaces dentro de un párrafo).
Los botones o áreas interactivas deben ser grandes como para no necesitar precisión al hacer click o tap. Esta necesidad está basada en la ley de Fitts.
En aplicaciones móviles el mínimo de un área interactiva debe ser 44x44pts.
Los campos de formularios deben mostrar su etiqueta al estar completos. Ver ejemplos de uso correctos e incorrectos.
Seguridad
La comunicación con los web services debe ser encriptada usando un certificado SSL/TLS
Todas las URL que lanza una aplicación deben ser https.
Solo se deben pedir al usuario los permisos mínimos y estrictamente los que son necesarios.
Los datos de los usuarios usados en la registración deben ser guardados con seguridad.
Limitar cantidad de intentos de logins.
Se deben actualizar los frameworks de desarrollo para evitar vulnerabilidades de seguridad.
Métricas
Las aplicaciones que no cuentan con Google Analytics deben incorporar el ID y métricas provistos por la Coordinación de Análisis de Datos de Servicios Digitales. En caso de disponer, se deberá brindar permisos de administración a una cuenta proporcionada únicamente para el análisis de información.
¿Por qué?
Para conocer más acerca de Google Tag Manager, véase el apartado "Referencias".
Las aplicaciones deben incorporar registros de errores y fallas.
¿Por qué?
Permite registrar, identificar y rastrear las fallas que se generan dentro de las aplicaciones, y poder resolverlos para mantener la mejora continua de las aplicaciones y ofrecer una mejor experiencia de uso al ciudadano.
Aplicación nativa en Android
Caso 1: Aplicaciones sin código de seguimiento pre-implementado.
Para implementar código de seguimiento en aplicaciones Android deberá consultar el documento "Implementación de Google Tag Manager". Chequear link abajo.
Caso 2: Aplicaciones con código de seguimiento previamente implementado.
En caso de poseer código de seguimiento se deberá brindar permisos de administración a una cuenta proporcionada únicamente para el fin de análisis de información. Asimismo se deberá modificar el mismo para que se adapte a los estandares propuestos en el documento de "Guía de estandarización de métricas digitales".
Aplicación nativa en IOS
Caso 1: Aplicaciones sin código de seguimiento pre-implementado.
Para implementar código de seguimiento en aplicaciones IOS deberá consultar el documento "Implementación de Google Tag Manager". Chequear link abajo.
Caso 2: Aplicaciones con código de seguimiento previamente implementado.
En caso de poseer código de seguimiento se deberá brindar permisos de administración a una cuenta proporcionada únicamente para el fin de análisis de información. Asimismo se deberá modificar el mismo para que se adapte a los estandares propuestos en el documento de "Guía de estandarización de métricas digitales".
Publicación
Las aplicaciones se subirán a las distintas tiendas de Servicios Digitales, excepto en caso de aplicaciones ya existentes.
¿Por qué?
Mantener una tienda centralizada permite al ciudadano conocer de una manera directa cuáles otros servicios y productos están disponibles.
Las aplicaciones las firmará Servicios Digitales con la firma y certificados propietaria.
¿Por qué?
Para lograr mantener la continuidad del producto sin importar el desarrollador o tercero.
Los Ministerios deben entregar las cuentas de las distintas tiendas y/o repositorios donde estén alojadas las aplicaciones a Servicios Digitales y las firmas digitales (alias y contraseñas) y/o certificados para publicar aplicaciones.
¿Por qué?
Para llevar un control y un seguimiento de los lineamientos establecidos por la Dirección Nacional.
Los ministerios que dispongan de sus propios proveedores deberán trabajar en un repositorio provisto por Servicios Digitales.
¿Por qué?
Para lograr mantener la continuidad del producto sin importar el desarrollador o tercero.
Los identificadores de cada aplicación deben ser únicos para cada dispositivo y tener el siguiente formato:
ar.gob.nombre_aplicacion
¿Por qué?
Permite tener homogeneidad en los identificadores de todas las aplicaciones.
Permite que exista una única aplicación instalada dentro del dispositivo.
Una vez publicadas las aplicaciones en las tiendas, los identificadores no pueden ser cambiados ni removidos porque sino se considera una nueva aplicación, completamente distinta de la anterior y no una nueva versión del mismo.
Los datos y las capturas de pantallas de las aplicaciones a ser publicadas deben ser provistas por cada Ministerio/Secretaría, respetando las plantillas provistas por la Coordinación de Aplicaciones Móviles.
Los datos de contactos/web/detalle de cada aplicación deben pertenecer a cada Ministerio/Secretaría y ser provistos para la carga en las tiendas.
¿Por qué?
Es importante que el ciudadano pueda contactarse con las áreas que publican las aplicaciones para disponibilizar información que crea necesaria.