UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA MONITOREO DE UN SOCKET AUTOAJUSTABLE

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO MECATRÓNICO

P R E S E N T A :

MARCELINO ALMANZA MAZAS

DIRECTORA DE TESIS:
M.I HANNA LESLYE GARCÍA GUERRA

MÉXICO D.F. **2014**

Agradecimientos

Le agradezco a Dios, por haber estado conmigo y guiado a lo largo de mi carrera.

Agradezco a mis padres, por ser mis dos pilares en mi vida dándome la fortaleza para continuar en mis momentos de debilidad, gracias por estar conmigo en las buenas y en las malas, ya que sin ellos no sería lo que soy.

A mi hermana Yazmin y a mi tía Evelin, por todo el cariño que me han brindado y las alegrías compartidas durante todo este tiempo.

A mi directora de tesis Hanna, por todos sus comentarios, consejos, sobre todo la paciencia y el tiempo que me tuvo. Gracias por brindarme su amistad.

Agradezco al proyecto PAPIIT el apoyo brindado durante el desarrollo de este trabajo. Esta tesis se desarrolló en el marco del proyecto PAPIIT IT 102512 “Diseño de sistemas mecatrónicos aplicados al ser humano”

Contenido

Índice de figuras

Objetivo

Diseñar e implementar una aplicación móvil para el monitoreo de los cambios de presión, temperatura y humedad, que junto con un sistema de instrumentación, adquirirá y registrará las variaciones dentro del socket autoajustable.

Alcances

Este trabajo presenta el diseño de una aplicación móvil enfocado al socket autoajustable para prótesis de miembro inferior a nivel transfemoral [1] y a la idea de poder llevar el monitoreo de ciertos parámetros en el teléfono móvil.

Se desarrolló una aplicación en el sistema operativo Android, que funge como herramienta de monitoreo, apoyado de sistema con sensores que indican los cambios que suceden dentro del socket autoajustable. Para crear la aplicación se tuvo que buscar una plataforma para el desarrollo de aplicaciones y además, se escogió sensores específicos que se adecuaran a las necesidades del diseño.

Las fases de este proceso de diseño son:

- El diseño conceptual de la aplicación.
- El diseño de configuración de la aplicación e instrumentación.
- El diseño a detalle de la aplicación y el circuito de sensado.

A lo largo de este trabajo se explicará a detalle cada una de estas etapas de diseño, también es importante hacer referencia, que el análisis de los costos en materiales y la factibilidad financiera no se incluirán en este trabajo de tesis.

Introducción

En el presente trabajo se muestra el desarrollo de una aplicación en la plataforma de Android, donde el usuario observará los cambios de la temperatura, la humedad y la fuerza ejercida dentro del socket. Los datos obtenidos tendrán el objetivo de crear una base de datos con información del usuario.

Contará con un sistema electrónico que se encargará de adquirir las variaciones de fuerza ejercida dentro del socket autoajustable, temperatura y humedad por medio de sensores, un microcontrolador y un módulo Bluetooth, para la comunicación inalámbrica. Formando un sistema de monitoreo entre el teléfono móvil y del socket autoajustable para prótesis de miembro inferior a nivel transfemoral.

En el primer capítulo se presentan conceptos acerca de la amputación, los diferentes niveles de amputación, prótesis transfemoral, los problemas que se tienen al usar una prótesis transfemoral y las partes que la conforman. También se expone el significado de un sistema operativo pero orientado a los teléfonos móviles, se da una explicación de los más populares en el mercado y el más usado en México.

En el segundo capítulo se explica el diseño conceptual llevado a cabo, se analiza y se busca una solución al problema planteado, exponiendo conceptos acerca de las características de los dispositivos tecnológicos a emplear y generando varias alternativas comparando sus ventajas y desventajas, para ayudar a seleccionar la mejor solución. A lo largo del tercer capítulo se explica la aplicación, el sistema de instrumentación que se realizó para los sensores ocupados y el funcionamiento de la aplicación móvil. En el cuarto capítulo se expone el diseño a detalle, es decir, especificaciones del sistema tanto para el circuito como la aplicación móvil, la localización de los sensores cómo se adquieren los datos y se almacenan en la memoria microSD.

Finalmente en el capítulo cinco, se exponen los resultados obtenidos a lo largo de los demás capítulos, el trabajo a futuro y las conclusiones generadas a partir de este trabajo.

Capítulo 1. Antecedentes

1.1 Conceptos básicos

Los seres humanos están sujetos a la pérdida de algún miembro inferior por diferentes razones, algunas son: enfermedades, accidentes, deformaciones congénitas entre otras. Perder una parte del cuerpo humano hace que las actividades diarias se vuelvan complicadas, pero gracias al desarrollo de la tecnología y la ciencia, se han creado nuevos sistemas para poder reemplazar el miembro amputado y llevar al usuario a incorporarse a su vida cotidiana. Para poder entender los conceptos básicos de este tema, es importante definir lo que es una prótesis, una amputación y comprender los diferentes niveles de amputación en un miembro inferior.

1.1.1 Amputación

La amputación consiste en la separación completa de una parte del organismo del resto del cuerpo y la desarticulación es una amputación pero a través de una articulación [2].

Otra definición de amputación: Es la remoción de una parte o la totalidad de una parte del cuerpo, realizada como un procedimiento quirúrgico, generalmente para evitar la propagación de la gangrena como una complicación de la congelación, la lesión, la diabetes, la arteriosclerosis, o cualquier otra enfermedad que afecta la circulación de la sangre. También se realiza para evitar la propagación del cáncer de hueso y para reducir la pérdida de sangre y la infección en una persona que ha sufrido un daño grave e irreparable a una extremidad. Cuando se realiza una amputación, los cirujanos generalmente cortan por encima de la zona enferma o lesionada de manera que una porción de tejido sano permanece para amortiguar el hueso. A veces la ubicación de un corte puede depender en parte de su idoneidad para ser equipado con una extremidad artificial o prótesis [3].

1.1.2 Niveles de amputación inferior

Existen diversos niveles de amputación en la extremidad inferior, los cuales se realizan dependiendo del criterio del médico, pero siempre procurando que el miembro residual sea útil para el uso de alguna prótesis. A continuación se muestran los diferentes niveles de corte (figura 1):

- Hemipelvectomía es la pérdida de la pierna completa junto con la mitad de la pelvis lateral.
- Desarticulación de la cadera es la pérdida completa del fémur.

- Amputación transfemoral se refiere al nivel de amputación en el hueso del fémur, la incisión puede clasificarse en: corto (menos del 35% del femur), mediano (entre 35% y 60% del fémur) y largo (entre 60% y 85% del fémur) estos porcentajes indican el hueso del fémur presente.
- Amputación supracondílea es donde la rótula se conserva y se aplica al extremo del hueso.
- Desarticulación de la rodilla se refiere a la amputación a nivel de la rodilla
- Amputación transtibial se refiere a la amputación debajo de la rodilla, la incisión puede clasificarse en: muy corto (menos del 20%), corto, estándar (entre el 20% y 50%) y largo (más del 50%), estos porcentajes indican el hueso de la tibia presente.
- Amputación de Syme se refiere a la desarticulación del tobillo en la que se mantiene el talón para una buena carga del peso.

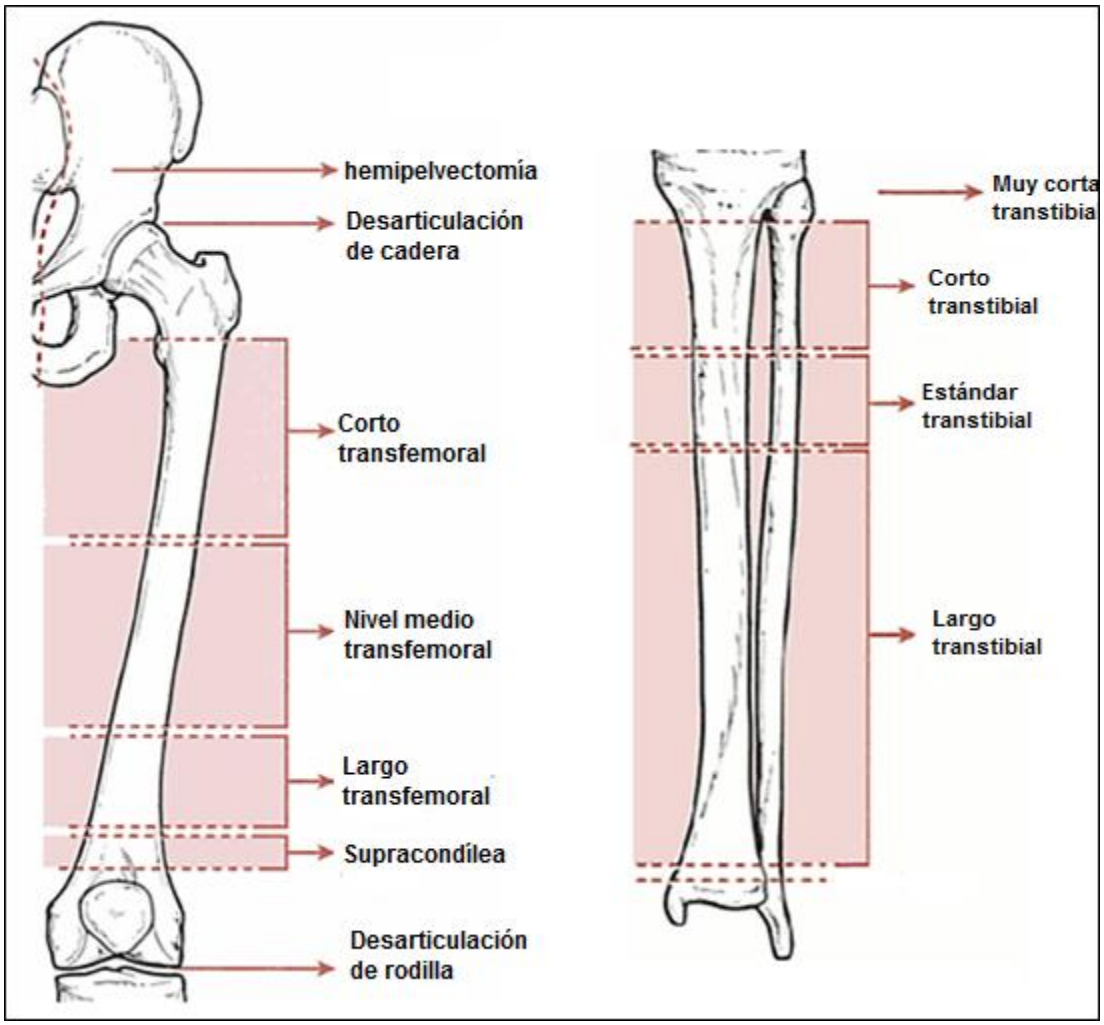


Figura 1. Niveles de amputación en miembro inferior [4]

1.1.3 Prótesis

Es necesario definir la palabra prótesis, para poder comprender el sistema artificial de alguna extremidad, prótesis es:

- Un sustituto fabricado para una parte del cuerpo que falta, tal como un miembro artificial que sustituye a un miembro amputado[5].
- Un sustituto artificial de una parte del cuerpo que falta [6].

Las personas comúnmente asocian prótesis con la falta de los brazos y las piernas, pero el término también se utiliza para referirse a los dispositivos internos tales como las válvulas del corazón, así como los ojos protésicos y otras partes del cuerpo. La intención detrás de una prótesis es para mejorar la calidad de vida para el usuario, mientras que también da a la persona más libertad e independencia.

La prótesis para una extremidad inferior es un sistema artificial que busca reemplazar las partes más importantes como la pierna, la rodilla, el tobillo y el pie, dependiendo del nivel de amputación realizada.

A continuación se explica las partes más importantes que componen una prótesis transfemoral (figura 2).

1.- Socket: Es la parte que alberga la extremidad residual y tiene comunicación con los demás componentes de trabajo, deben de estar hechos a la medida del usuario, pueden tener un socket-socket de interior suave, flexible y que proporcione confort.

Hay dos diseños básicos de socket:
- Socket cuadrilateral
- Socket de contención isquiática

2.- Sistema de suspensión: Es el que sostiene la extremidad residual dentro del socket, la adaptación puede ser de diferentes maneras por ejemplo: con una banda pélvica, un encaje de succión entre otros.

3.- Sistema de rodilla: Desempeña la función de evitar el pandeo mientras el usuario está de pie y permite que la pierna artificial pueda continuar avanzando hacia adelante a voluntad.

4.- Sistema del pie: Proporciona, principalmente el apoyo necesario a todo el sistema protésico cuando el usuario está de pie o caminando.

Figura 2. Prótesis femoral [36]

1.1.4 Problemas con el uso de prótesis

Los problemas cutáneos son una de las afecciones más comunes que padecen los usuarios de prótesis de extremidades inferiores en la actualidad. Alrededor del 75% de los amputados que utilizan este tipo de prótesis sufren problemas cutáneos. De hecho, los casos de afecciones dermatológicas en amputados superan en un 65% los de la población en general [7].

Con una prótesis, el cuerpo se somete a condiciones mecánicas y térmicas anormales, como el contacto entre el socket y la piel. Esto puede traumatizar el tejido por un exceso de tensión, fricción o calor. Además, la piel reacciona a los aumentos de temperatura con la transpiración, que no puede evaporarse en el entorno cerrado de la prótesis. Esto genera más calor y humedad, lo cual ablanda la piel y deteriora su integridad normal (maceración).

La presión es otro factor mecánico que se agrega en el socket protésico. Algunas partes de la anatomía humana están preparadas para amortiguar la presión, como el tejido adiposo del talón. Al amputar una extremidad, se elimina o altera una parte de la anatomía que normalmente distribuye la fuerza. Por lo tanto, se deben usar áreas anatómicas que no están preparadas para soportar la fuerza aplicada. Un socket inadecuado puede incrementar la fuerza sobre las áreas blandas y acelerar el deterioro de la piel.

Las úlceras por presión a menudo pueden corregirse con ajustes menores de la prótesis. Sin embargo, en ocasiones las áreas afectadas pueden ser más grandes y requerir un tiempo de recuperación sin la prótesis, o un socket completamente nuevo.

La dermatitis por contacto irritante y la dermatitis por contacto alérgica son dos de los problemas más comunes que afectan a los usuarios de prótesis.
Ambas pueden aparecer cuando la piel se expone a un material que produce un deterioro cutáneo. Si la prótesis tiene un componente irritante o alérgico conocido, se lo debe cambiar por otro material.

Si no se la trata, la dermatitis puede producir inflamación crónica, daño celular y carcinogénesis (cáncer). Por lo tanto, es muy importante que todos los usuarios de prótesis consulten a un médico cuando la terapia conservadora resulte ineficaz o cuando la lesión no cicatrice. La evaluación de estas lesiones es fundamental para poder descartar diversos tipos de cáncer.

Los problemas cutáneos son muy comunes en los amputados. La prevención de complicaciones cutáneas comienza por una higiene adecuada e inspecciones diarias de la piel. Si se detecta una lesión cutánea que no puede solucionar o que no cicatriza, el primer paso es consultar a un protésico.

Como los amputados suelen exigirle demasiado a la piel y a menudo descartan la opción de no usar una prótesis, en ocasiones subestiman la importancia de la higiene y la inspección cutáneas. Los problemas cutáneos deben tomarse con seriedad. Un simple deterioro de la piel puede acarrear problemas más graves, como una reintervención quirúrgica.

Las amputaciones pueden provocar atrofia muscular, esto se produce por no realizar una cirugía en el momento de la amputación, con lo que existe un predominio de unos grupos musculares sobre otros, con la consecuente pérdida de masa muscular y reducción del volumen del muñón. Cuanto menor sea la longitud del muñón, mayor es el número de músculos implicado, con lo que aumentan las posibilidades de atrofia muscular. Por ello, se debe recordar, al realizar la amputación, preservar la máxima longitud de muñón y realizar una mioplastia (reparación plástica de un músculo) adecuada [8].

1.1.5 Estadísticas en México

De acuerdo a lo consultado la primera causa de amputación de algún miembro en la población mexicana, es la diabetes mal controlada, seguida de accidentes, tumores de hueso o enfermedades infecciosas.

Información reciente de investigaciones del Instituto Nacional de Salud Pública (INSP) revela que la prevalencia de diabetes mellitus en la población mexicana es de 14.4%, lo que significa que cerca de 14 millones de personas sufran esta enfermedad y que 74 mil mueran cada año por dicha causa.

De hecho, el 25% de los diabéticos presenta afectación cardiovascular o insuficiencia arterial, lo que en un futuro terminará con la amputación de una o ambas piernas, según el INSP [9].

"Más de 80% de nuestras ventas en México están dirigidas a los pacientes con diabetes, que perdieron sus extremidades inferiores en diversos niveles: dedos, empeine, pie completo, por debajo de la rodilla, arriba de ella o incluso hasta llegar a la cadera", afirmó el ortesista Xico Rojas Vargas, director del área de Producción de Prótesis y Órtesis de Ottobock [10].

Desde el año 2001 en México se puso un programa de acción para dicha enfermedad en el que consiste informar y fomentar a la población acerca de los estilos de vida saludables, realizar campañas de comunicación entre otras.

Se muestra en la (figura 3) que el mayor número de amputaciones son de los miembros inferiores, de acuerdo a los varios niveles de amputación (desarticulación de cadera, desarticulación de rodilla, desarticulación del tobillo, amputación parcial de pie, amputación transtibial y amputación transfemoral) esta última es la que nos interesa.

EGRESOS HOSPITALARIOS POR AMPUTACIÓN DE MIEMBRO SUPERIOR Y DE MIEMBRO INFERIOR REGISTRADOS EN LAS UNIDADES MEDICAS DE LA SECRETARÍA DE SALUD Y DE LOS SERVICIOS ESTATALES DE SALUD, 2004-2006

Procedimiento	2004	2005	2006
8400 Amputación de miembro superior, no especificada de otra manera	108	68	72
8401 Amputación y desarticulación de dedo de mano	791	768	795
8402 Amputación y desarticulación de dedo pulgar	60	95	91
8403 Amputación a través de mano	73	84	84
8404 Desarticulación de muñeca	11	12	13
8405 Amputación a través de antebrazo	51	56	42
8406 Desarticulación de codo	5	6	11
8407 Amputación a través de húmero	20	9	16
8408 Desarticulación del hombro	20	12	14
8409 Amputación intertoracoescapular	3	6	7
8410 Amputación de miembro inferior, no especificada de otra manera	501	487	549
8411 Amputación de dedo de pie	2,809	3,215	3,623
8412 Amputación a través de pie	573	635	665
8413 Desarticulación de tobillo	10	11	12
8414 Amputación de tobillo a través de tibia y peroné	75	91	81
8415 Otra amputación debajo de la rodilla	357	373	388
8416 Desarticulación de rodilla	23	29	45
8417 Amputación por encima de la rodilla	1,770	1,814	1,987
8418 Desarticulación de cadera	49	58	72
8419 Amputación abdominopélvica	17	43	32
Total:	**7,325**	**7,872**	**8,599**

Figura 3. Egresos hospitalarios por amputación [11]

De acuerdo a los padecimientos el no tener un miembro del cuerpo genera una discapacidad, los resultados arrojados por una encuesta realizada por la Secretaría de Salud nos dice que la principal causa de la discapacidad fue una enfermedad con 40 %, un accidente (23 %) y la edad avanzada (16 %) además de los 771 172 derechohabientes del IMSS con discapacidad, el 56.6 % de ellos tiene limitación de tipo motriz, esto es, dificultades para caminar o usar brazos o manos. Sin embargo, la mayor proporción corresponde a limitaciones para caminar, con un 51.2 %[12].

La pérdida de extremidades en personas ha presentado un incremento en los últimos años por ejemplo en Estados Unidos, se estima que 1 de cada 200 personas han sufrido alguna amputación, según datos proporcionados por la National Limb Loss Information Center en el año 2006 [13].

La Organización Panamericana de la Salud (OPS) y la Organización Mundial de la Salud (OMS) precisan que el total de las amputaciones de extremidades inferiores, entre el 40% y 85% están relacionadas con la diabetes. En México, la Secretaría

de Salud informa que en el año 2006 se amputaron 75 mil extremidades inferiores, 7 de cada 10 amputaciones de pierna se realizan a personas con diabetes [14].

1.2 Definición de socket autoajustable

Como se explicó anteriormente, el socket es una parte esencial de la prótesis ya que da alojo a la extremidad sobrante y conecta con el resto de la pierna artificial.

En la Facultad de Ingeniería de la UNAM, se ha propuesto el diseño de un tipo de socket que cubre las necesidades que se presentan en el muñón, cuidando el confort, la funcionalidad y el ajuste con el miembro residual. En cuestión del ajuste, se desarrolló un sistema cuyo principal objetivo es compensar la pérdida de masa muscular en el muñón, éste consta de cuatro bolsas con diferentes geometrías de acuerdo a cada cara del muñón (figura 4).

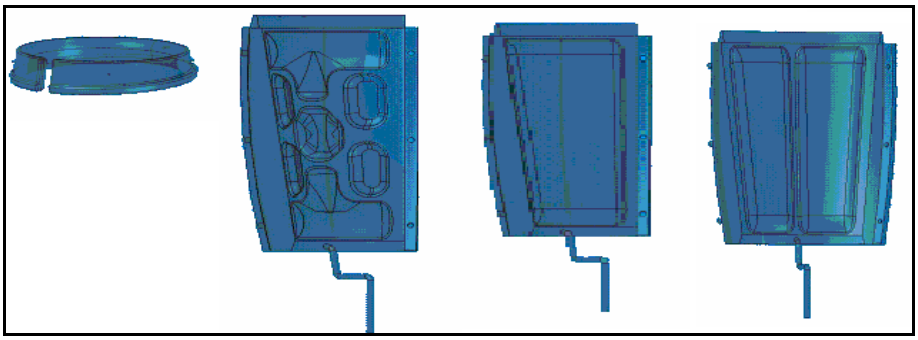


Figura 4. Sistema de ajuste [1]

Los sujetadores hacen que se ocupe mayor área de contacto además compensarán los cambios de masa muscular en la extremidad amputada, por medio del llenado de aire en las mismas bolsas bajo el control de un circuito electrónico.

Otros elementos importantes que componen al socket son la abrazadera que se encarga de sujetar el socket con el muñón, la estructura de apoyo que proporciona soporte al socket, la malla que amortigua los movimientos realizados en las actividades diarias y la base que tiene la función de sostener la estructura de apoyo, además de servir de unión entre la pierna artificial y el socket (figura 5).

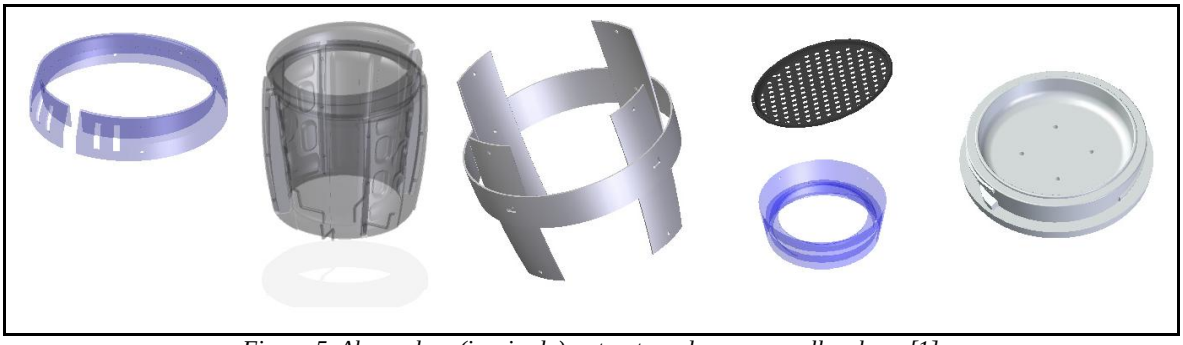


Figura 5. Abrazadera (izquierda), estructura de apoyo, malla y base [1]

Figura 6. Finalmente con el ensamble de las piezas forman el socket autoajustable [1]

1.3 Sistemas operativos móviles

Atrás quedaron los días cuando los teléfonos móviles eran sólo unos dispositivos para hacer llamadas telefónicas y comunicarse. Debido a que el sistema operativo móvil se actualiza conforme el paso de los años para hacerlo un dispositivo más inteligente y capaz de realizar lo mismo que una computadora, ahora los celulares son similares a las computadoras portátiles, en el que podemos enviar correos electrónicos, leer libros, navegar en internet y jugar, pero comúnmente, estos son conocidos como 'Smartphone' teléfonos inteligentes.

1.3.1 Definición

Un sistema operativo (SO, frecuentemente OS, del inglés Operating System) es una colección de software que gestiona los recursos de hardware y proporciona servicios comunes a los programas de una computadora. El sistema operativo es un componente vital del software en un sistema informático. Los programas de aplicación por lo general requieren un sistema operativo para funcionar [15].

De acuerdo a lo anterior, se puede deducir que un sistema operativo móvil es un programa que se encarga de realizar la comunicación simple y segura entre el dispositivo y el usuario, además se encarga de administrar los recursos del teléfono como son: memoria, procesador, aplicaciones, puertos de entrada y salida, interacción con los demás programas etc.

Los OS en los celulares se han desarrollado rápidamente, por la gran competencia mundial que existe entre las empresas y a la demanda que hay entre los usuarios de alrededor del mundo, provocando que los productos de las diferentes compañías se estén innovando cada vez y más.

Al grado que la telefonía celular se ha convertido en parte importante de la vida cotidiana y conforme la tecnología avanza en materia de telecomunicaciones es y será elemental que se diseñen sistemas que sean rápidos, de fácil comunicación con el usuario, accesibles y modernos.

1.3.2 Sistemas operativos móviles actuales

Se están utilizando diversos sistemas operativos en los teléfonos móviles. Algunos de los más conocidos en el mercado son Symbian, BlackBerry OS, iOS, Windows Mobile y Android. La mayoría de ellos están asociados con marcas específicas de teléfonos porque son fabricados por empresas específicas, mientras que otros son de código abierto y están disponible en una variedad de plataformas, a continuación solo se explica brevemente algunos de los sistemas más comerciales, para tener una idea qué características tiene cada uno de ellos.

Symbian

Symbian es un sistema operativo para móviles (OS), se encuentra principalmente en los teléfonos Nokia ofreciendo un alto nivel de integración con la comunicación y la gestión de información personal (PIM) funcionalidad.

Figura 7. Sistema operativo Symbian[25]

Symbian OS combina middleware de comunicaciones inalámbricas a través de un buzón integrado y la integración de Java y la funcionalidad de PIM (agenda y contactos).

Symbian es de código abierto, lo que significa que cualquier persona puede utilizarlo sin tener que pagar. Es ampliamente utilizado, pero no es el más avanzado y con todas las funciones de los sistemas operativos de los teléfonos móviles. La mayoría de los teléfonos que utilizan Symbian son dispositivos de gama baja, no teléfonos inteligentes con todas las funciones. Muchos fabricantes, como Nokia, que había sido el mayor defensor del sistema operativo, han cambiado a otros sistemas operativos (figura 7).

BlackBerry OS

El sistema operativo BlackBerry es un sistema operativo móvil desarrollado por Research In Motion, es de los más conocidos en los teléfonos móviles. Fue realizado para uso en los populares dispositivos de de la empresa BlackBerry.

La plataforma BlackBerry es popular entre los usuarios corporativos, ya que ofrece sincronización con otro software de negocio Microsoft Exchange, Lotus Domino, Novell Group Wise de correo electrónico y, si se utiliza con BlackBerry Enterprise Server (figura 8). Haciéndolo a la medida para las empresas, con la funcionalidad de tomar poder sobre la personalidad o la apariencia. Pero se centra en la mensajería, correo electrónico y otras funciones de comunicación. Los reproductores multimedia y otras aplicaciones basadas en el arrastre son menos comunes para el dispositivo.

Figura 8. Sistema operativo BlackBerry 6.0 [28]

iOS

El sistema operativo móvil de Apple es conocido como iOS desarrollado originalmente para iPhone (figura 9), siendo después usado en el iPad, iPad 2 y iPod Touch.

La seguridad del sistema operativo y la compatibilidad ha sido un punto de importante para algunos porque lo que respecta a iOS es totalmente de código cerrado, Apple no autoriza que su sistema operativo sea usado en otros teléfonos. Adobe Flash, por ejemplo, no funciona en el sistema operativo incorporado. Sin embargo, iOS sí tiene una amplia variedad de aplicaciones y una interfaz que muchos prefieren por su facilidad de uso.

Figura 9. iOS Apple [29]

Windows Mobile

Windows Mobile, también conocido como Windows Phone (figura 10), es la versión móvil del sistema operativo de Microsoft.

Debido a esto, es fácilmente compatible con muchos programas de Windows, como Microsoft Office, por lo que es una opción popular para los empresarios. Windows Mobile fue diseñado originalmente para la línea de Pocket PC de Microsoft antes de ser adaptado para su uso en los teléfonos.

Figura 10. Windows Mobile [31]

Android

Android está basado en el sistema operativo Linux. Originalmente fue desarrollado por una empresa independiente, que más tarde fue comprado por Google, aunque el sistema operativo sigue siendo libre y de código abierto.

Android es elogiado por su flexibilidad de poder trabajar en diferentes plataformas de programación. Cualquier persona puede desarrollar aplicaciones para el sistema operativo, y cualquier compañía puede lanzar un teléfono con él. Las actualizaciones para el sistema operativo móvil Android se han desarrollado bajo nombres de postres en inglés como Cupcake, Donut, Eclair, Gingerbread, Honeycomb, Ice Cream Sándwich, Jelly bean, Kit-kat, asignados por orden alfabético, cada nueva versión tiene mejoras y actualizaciones (figura 11).

Figura 11. Las diferentes actualizaciones de Android[30]

Los sistemas mencionados anteriormente son diferentes entre ellos, cada uno posee características que otros no tienen, además de que son hechos por diferentes compañías y para distintos teléfonos, unos pueden ser más rápidos al iniciar o soportar más aplicaciones etcétera. Existen otros sistemas para dispositivos móviles, por ejemplo WebOS de Palm, Bada y Maemo de Noki, éstos son sólo algunos de los sistemas operativos que se utilizan en los teléfonos inteligentes de todo el mundo. Teniendo en cuenta la naturaleza cambiante de la industria de la tecnología, aún más, están obligados a aparecer en el mercado de la telefonía móvil.

Cabe mencionar que en México, el sistema operativo móvil de Apple es la plataforma con mayor uso por sus teléfonos inteligentes y tabletas, además, este país es el que presenta mayor tráfico en Internet proveniente de dispositivos móviles en América Latina.

El estudio de comScore titulado *Futuro Digital Latinoamérica 2013* reveló que el sistema operativo iOS es la plataforma líder en México (figura 12), seguido por Android, BlackBerry, Windows Phone y Symbian; la participación de mercado se basa en smartphones y tabletas; en el resto de Latinoamérica, Android es el operativo más utilizado [16].

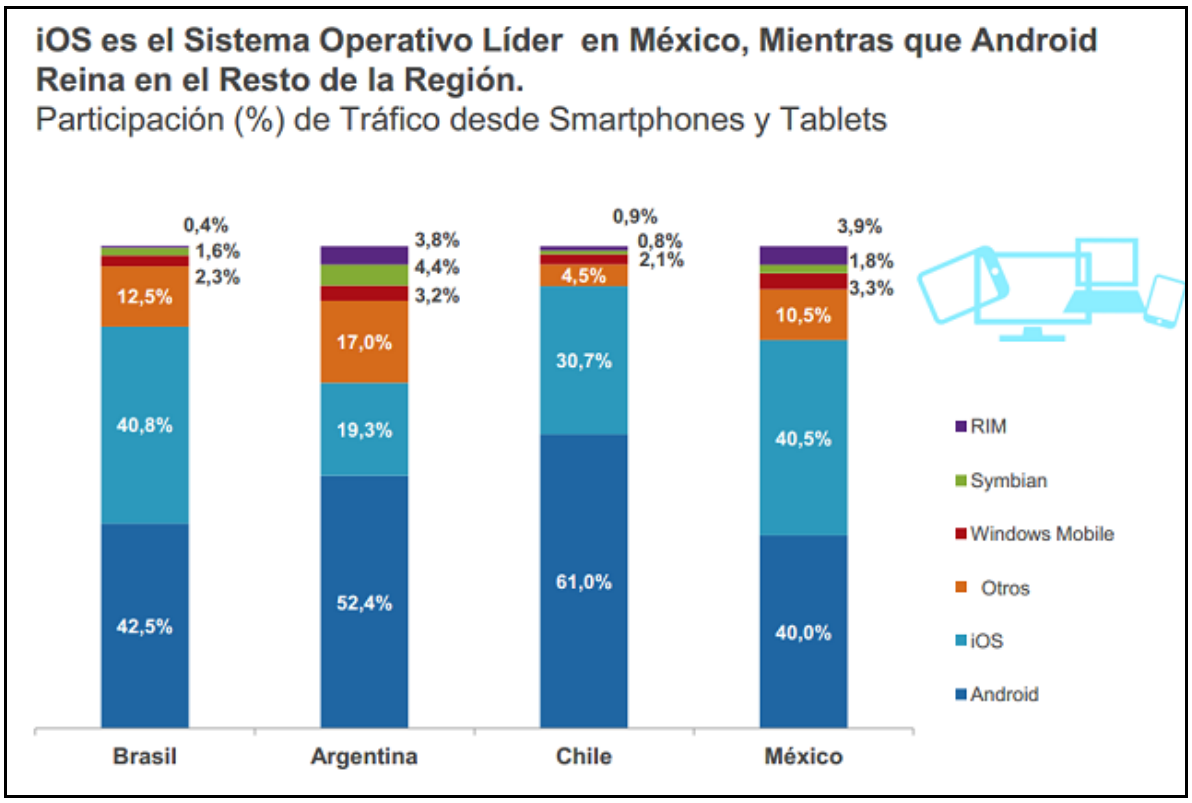


Figura 12. Estudio de comScore [17]

1.3.3 Definición de aplicación móvil

Una aplicación móvil es un programa informático creado para realizar una tarea específica dentro de un sistema operativo móvil, también llamada app. Las aplicaciones nacen por alguna necesidad concreta de los usuarios, y se usan para facilitar o permitir la ejecución de ciertas tareas en donde se ha detectado una cierta necesidad. Debido a la gran diversidad de aplicaciones que existen, pueden estar dirigidas a diferentes sectores, como son: el empresarial, el educativo, el recreativo, de comunicaciones, bancario entre otros.

1.3.4 Aplicaciones móviles actuales

Las empresas dedicadas al desarrollo de prótesis, para reemplazo de algún miembro amputado, han visto en el mundo actual, que el uso de los dispositivos móviles se vuelve cada vez más común entre las personas para comunicarse y realizar otras actividades cotidianas, se han enfocado en crear sus propias aplicaciones para teléfonos inteligentes que puedan conectarse con sus productos. Las aplicaciones móviles para ciertas prótesis tienen el propósito de realizar tareas específicas como recabar información acerca del funcionamiento del sistema y el usuario, seleccionar posiciones determinadas en las prótesis y entre otras más. A continuación se muestran aplicaciones móviles desarrolladas por las empresas Touchbionics y Orhtocare innovation.

La empresa Touchbionics, desarrolló una mano artificial, llamada *i-limb ultra revolution*, capaz de realizar movimientos con los dedos articulados con los que cuenta, muy similares a los de una mano real. Cuenta con una aplicación móvil desarrollada en el sistema operativo de Apple (iOS) para el control de la prótesis de miembro superior, llamada *biosim* (figura 13) usa el Bluetooth del teléfono para activar 24 posiciones en la mano. Otras funciones de dicha aplicación es que puede recopilar las posiciones más usadas por el usuario, como escribir, tomar papeles o el uso del mouse de una computadora, también incluye una gráfica de las señales de control muscular y es compatible con iPad, iPod o iPhone. La aplicación es gratuita, debido a que solo funciona con la prótesis de la empresa Touchbionics y el precio de la extremidad artificial se encuentra entre $60,000 y $120,000 (£39,000-£78,000) dependiendo del país [18].

Figura 13.Aplicación biosim[26]

Por otra parte la empresa Orhtocare innovation, creó una conexión entre el paciente, el dispositivo y el clínico. Cambió la vista en los resultados del tratamiento, debido a que *galileo* evalúa el funcionamiento, proporciona un control, crea documentos y la evidencia del impacto de la prótesis, al ser usada por el usuario (figura 14). Los médicos con ésta aplicación tienen ahora la documentación de las actividades, realizadas por el paciente [19].

Figura 14. Aplicación galileo[19]

Estas dos aplicaciones desarrolladas por diferentes empresas, cada una de ellas realizan tareas específicas, una para posicionar una mano y la otra para una prótesis inferior, ambas fueron hechas para el sistema operativo de Apple (iOS).

Capítulo 2. Diseño conceptual

2.1 Identificación de las necesidades

Para poder identificar las necesidades, se estudian y analizan escritos en libros, páginas web y otras fuentes de información, relacionados a los temas de amputación a miembro inferior, al uso de prótesis y aplicaciones móviles, también apoyándose en estadísticas para hacer posible una investigación donde se pueda extraer características importantes que ayuden a identificar las necesidades primordiales.

Las personas que usan una prótesis experimentan una adaptación física, porque perder una extremidad es solo el comienzo y no el fin de un tratamiento de cuidados especiales, que se deben llevar a cabo, las amputaciones a cualquier nivel suelen estar acompañadas de diferentes problemas relacionadas con la piel, debido a que está sometida a muchos maltratos, la mayoría de las prótesis de pierna cuentan con un socket muy ajustado en el que el aire no puede circular fácilmente y puede retener la transpiración [20].

Es importante señalar que el socket tiene que permitir la carga de peso, pero si ésta no se produce de manera uniforme puede provocar tensión o excoriar áreas localizadas de la piel del muñón. Por ejemplo, la piel experimentaría un estiramiento intermitente a causa de la fricción que se produce contra el borde del socket y la superficie interior del mismo (con algunas prótesis se usan fundas para muñones con el fin de reducir dicha fricción). Además, la piel del muñón es vulnerable a posibles reacciones irritantes o alérgicas a los materiales usados en la fabricación del socket protésico o cuentan con un socket muy ajustado en el que el aire no puede circular fácilmente y puede retener la transpiración.

Otro término necesario mencionar, son los edemas los cuales puede llegar a formarse por una cantidad excesiva de fluido, se acumula en los tejidos blandos de la extremidad restante, llegando incluso excoriarse, ulcerarse o gangrenarse debido a la alteración de la irrigación sanguínea. Los edemas pueden bloquear los canales linfáticos y vasculares e impedir la nutrición necesaria de los tejidos, estos problemas están asociados a factores mecánicos de algunas prótesis que contribuyen a la formación de los edemas, ulceraciones, irritaciones entre otros problemas de la piel, causadas por la estrechez del socket, una mala adaptación y alineación.

Dadas estas características, en la aplicación móvil de monitoreo (AppMonitor) se busca cubrir algunos de los factores que pueden causar daños en la piel de una persona amputada, enlistando las siguientes necesidades identificadas:

- AppMonitor trabaje en un segundo plano en el teléfono.
- AppMonitor avise cuando sobrepase cierta temperatura y humedad dentro del socket autoajustable.
- AppMonitor avise si existe un buen ajuste entre muñón y socket.
- AppMonitor registre las variaciones dentro del socket.
- El sistema de sensado debe hacer su trabajo en tiempo real.

El problema principal, es el uso de la prótesis en un día normal, conforme transcurre el día dentro del socket de la prótesis empieza aumentar la temperatura, debido a la fricción entre el muñón y el mismo socket, al igual la humedad tiende aumentar causando sudoración favoreciendo la fricción, que puede llegar a causar problemas en la piel de la extremidad restante.

2.2 Identificación del problema

Este trabajo tiene como finalidad el desarrollo de una aplicación móvil para el monitoreo de la temperatura, humedad y cambios de fuerza dentro del socket autoajustable, para el desarrollo del app se debe elegir la plataforma en Android con la que se va a desarrollar. Los sensores que se van implementar en el socket autoajustable para medir dichas variables y también seleccionar un medio viable, inalámbrico y compatible entre el microcontrolador y el celular.

Para poder hacerle frente, se debe descomponer el problema complejo en subsistemas más sencillos, para aclarar el problema en el siguiente diagrama (figura 15) muestra a grandes rasgos los componentes que definen el problema principal.

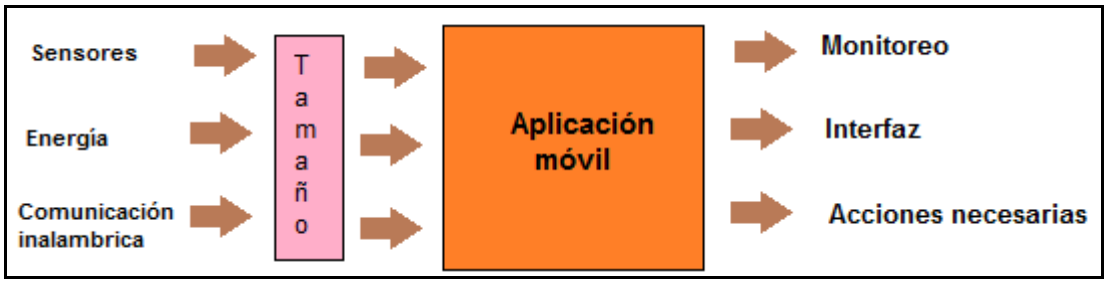


Figura 15.Diagrama general

En este segundo diagrama (figura 16) los cuadros de color rosa indican los subsistemas como: el sensado, la comunicación inalámbrica (entre el teléfono celular y el microcontrolador), el desarrollo de la aplicación móvil en Android y el medio para almacenar los datos adquiridos que incluyen la fecha y hora al momento que ocurrieron. En el siguiente tema (2.3) se explica la propuesta de conceptos para cada subsistema del problema principal.

En el cuadro de color anaranjado indica, la aplicación de monitoreo instalada en el celular, el cual tiene una interfaz que permite al usuario visualizar los cambios de

las variables ya menciondas anteriormente. Con la finalidad que el usuario decida si es tiempo de descansar del socket, darle un respiro a la extremidad amputada o simplemente cambiar de posición en la que se encuentra para corregir la fuerza ejercida dentro del socket y asi evitar posibles problemas relacionados con la piel del muñón, que en el capítulo de antecedentes se describieron.

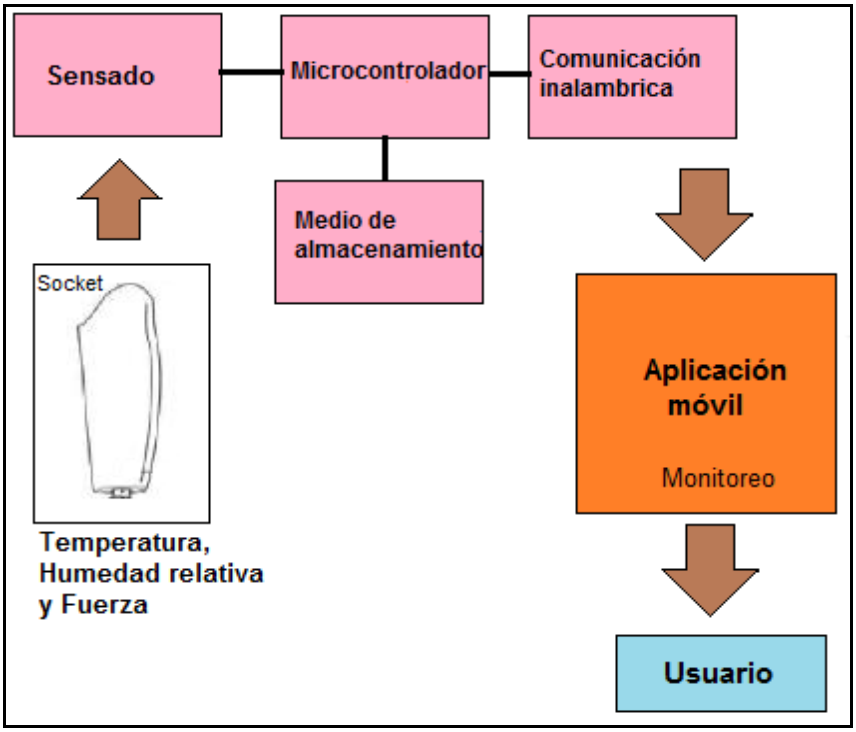


Figura 16. Subsistemas

2.3 Conceptos propuestos para la aplicación y la implementación

2.3.1 Conceptos para el desarrollo de la aplicación

El sistema operativo Android es una plataforma abierta, lo que significa que no está atado a ningún fabricante de hardware. Como se da a notar, esta apertura es lo que le ha permitido a Android abrirse paso en el mercado rápidamente.
Este mismo código fuente abierto es el que permite a los fabricantes de teléfonos móviles y tabletas, crear interfaces de usuario y añadir características para incorporarlas a sus dispositivos y como desarrollador, se tiene la oportunidad de desarrollar aplicaciones para un mercado que crece día a día, un mercado con miles de usuarios para los que puedes escribir una aplicación que puede ser descargada y usada por ellos.
A continuación se proponen cuatro conceptos de distintos entornos para el desarrollo de la aplicación móvil y se exponen de forma breve cada una de ellas.

Basic4 Android

Basic4Android es un entorno comercial que permite desarrollar aplicaciones para Android, haciéndolo en un lenguaje muy similar a Visual Basic, esto quiere decir que se tiene un entorno gráfico en donde se puede añadir botones fácilmente, ventanas, otros componentes y personalizar sus propiedades, sin embargo al compilar, es decir, en el fondo del programa seguirá siendo lenguaje Java. Las Aplicaciones creadas Basic4android son igual de buenas, y rápidas como codificada en lenguaje Java.

Una herramienta importante es que nos permite usar cómodamente ciertas librerías que nos facilitarán el trabajo, el programador puede usar las bibliotecas existentes creadas por otros usuarios para extender aún más su funcionalidad.
Algunas de estas librerías nos permitirán trabajar con el GPS del móvil, el Bluetooth, interacción con sitios web usando HTTP, tratamiento multimedia con archivos locales, controlando la cámara del móvil, además de trabajar con reconocimiento de voz entre otras funciones.

Esta plataforma no es gratuita, la versión económica se encuentra alrededor de $630.00 pesos y la versión completa cuesta aproximadamente$3,200.00 pesos mexicanos, la diferencia son las actualizaciones, en la de menor precio se tiene completo acceso a la plataforma pero con solo dos meses de actualizaciones, y la de mayor costo tiene dos años de actualizaciones.

App Inventor

App Inventor es una herramienta de diseño, un lenguaje de programación y un entorno de desarrollo de aplicaciones para móviles y tabletas que funcionen con el sistema operativo Android, es un software libre, no es necesaria la adquisición de licencias.

Esta plataforma de desarrollo fue creada por Google hace un tiempo con el propósito de que más personas conocieran el sistema operativo Android; esta estupenda herramienta usa el navegador de internet como centro principal de trabajo, y almacena todo esto en servidores que están disponibles cada vez que entres a internet, haciendo que puedas trabajar la aplicación en cualquier computadora que cuente con internet, además App Inventor permite también ejecutar las aplicaciones en un emulador o en un teléfono móvil.

La programación se realiza usando bloques, están hechos con elementos comunes a la mayoría de los lenguajes de programación existentes y se distinguen por colores dependiendo la función que desempeñen, tan solo es arrastrarlos al área de trabajo con la acción que se necesite.
Se colocan bloques para construir bucles, condiciones, variables, etc. que permiten pensar lógicamente y solucionar los problemas de forma metódica,

diferente a otros lenguajes de programación donde usualmente ocurre como encontrar el punto y coma o los paréntesis, que están donde no deben y producen errores de compilación o ejecución.

Mono para Android

Mono es una plataforma de software diseñado para permitir a los desarrolladores crear fácilmente aplicaciones en la plataforma, basada en los lenguajes C# y .NET que Microsoft desarrolló, las cuales son muy usados en diferentes ambientes.
Mono para Android permite crear aplicaciones nativas que se ejecutan en Android. Estas aplicaciones se ven y se sienten como aplicaciones nativas de Java, con Mono para Android, las aplicaciones se compilan en código ejecutable que se ejecuta en los dispositivos Android.

Algunos componentes que conforman Mono:

- C # Compiler - C # compilador de Mono es característica completa para C # 1.0, 2.0, 3.0 y 4.0 (ECMA).
- Mono Runtime - El tiempo de ejecución ejecuta el Common Language Infrastructure ECMA (CLI). El tiempo de ejecución proporciona un (JIT) compilador Just-in-Time, un compilador Ahead-de-tiempo (AOT), un gestor de biblioteca, el recolector de basura, un sistema de rosca y la funcionalidad de interoperabilidad.
- Base Class Library - La plataforma Mono proporciona un completo conjunto de clases que proporcionan una base sólida para construir aplicaciones en. Estas clases son compatibles con. NET Framework de Microsoft.
- Mono Class Library - Mono también ofrece muchas clases que van más allá de la biblioteca de clases base proporcionada por Microsoft. Estos proporcionan funcionalidades adicionales que son útiles, especialmente en la construcción de las aplicaciones de Linux. Algunos ejemplos son las clases para Gtk +, archivos Zip, LDAP, OpenGL, El Cairo, POSIX, etc

Por otro lado está el tema del costo, la versión más económica de Mono es de $4,100.00 pesos mexicanos. Pero existe una versión libre con 30 días para probar las herramientas y librerías de este entorno de programación.

SDK Eclipse

El SDK (Software Development Kit) de Android es un conjunto de herramientas y programas de desarrollo que permite al programador crear aplicaciones para un determinado paquete de software, estructura de software, plataforma de hardware, sistema de computadora, consulta de videojuego, sistema operativo o similar.

Comprende un depurador de código, biblioteca, un simulador de teléfono basado en QEMU, documentación, ejemplos de código y tutoriales. Las plataformas de desarrollo soportadas incluyen Linux (cualquier distribución moderna), Max OS X

10.4.9 o posterior, y Windows XP o posterior. La plataforma integral de desarrollo (IDE, Integrated Development Environment) soportada oficialmente es Eclipse junto con el complemento ADT (Android Development Tools plugin), aunque también puede utilizarse un editor de texto para escribir ficheros Java y Xml y utilizar comandos en un terminal (son necesarios los paquetes JDK, Java Development Kit y Apache Ant) para crear y depurar aplicaciones. Además, pueden controlarse dispositivos Android que estén conectados (ejemplo reiniciarlos, instalar aplicaciones en modo remoto).

Las Actualizaciones del SDK están coordinadas con el desarrollo general de Android. El SDK soporta también versiones antiguas de Android, por si los programadores necesitan instalar aplicaciones en dispositivos ya obsoletos o más antiguos. Las herramientas de desarrollo son componentes descargables, de modo que una vez instalada la última versión, pueden instalarse versiones anteriores y hacer pruebas de compatibilidad.

Una aplicación Android está compuesta por un conjunto de ficheros empaquetados en formato .apk y guardada en el directorio /data/app del sistema operativo Android (este directorio necesita permisos de súper-usuario,root, por razones de seguridad). Un paquete APK incluye ficheros .dex(ejecutables Dalvik, un código intermedio compilado), recursos, etc.

2.3.2 Conceptos para la comunicación inalámbrica

La comunicación inalámbrica es muy importante ya que por medio de ella se transmiten los datos entre el teléfono móvil y el socket autoajustable, para llevar a cabo el monitoreo entre estos dos dispositivos se debe tomar en cuenta puntos importantes como: la distancia que existe entre los dispositivos, el tamaño del módulo de comunicación para que pueda implementarse en el sistema de prótesis y además que sea compatible con cualquier teléfono móvil, a continuación se explican algunos de los conceptos propuestos acerca de los módulos de comunicación posibles para ser tomados en cuenta.

Módulo Bluetooth

La comunicación a través del Bluetooth se define como un estándar de comunicaciones inalámbricas de corto alcance mediante señales de radiofrecuencia que permite la transmisión de datos y voz [21].

La tecnología inalámbrica del Bluetooth fue desarrollada para reemplazar cables y como sistema de comunicación de corto alcance. Diferentes dispositivos pueden emparejarse con otros dispositivos siempre y cuando estén dentro del rango de comunicación, una de las bondades del Bluetooth es que no necesita un programa en específico o una licencia para su uso.

En la actualidad la tecnología Bluetooth se encuentra en: consolas de videojuegos, dispositivos móviles, sistemas de audio, altavoces y computadoras entre otras.

El dispositivo para realizar la comunicación inalámbrica es el modulo serial Bluetooth HC-06 (figura 17) es una pieza muy económica que se puede encontrar fácilmente en el mercado, se puede usar con cualquier microcontrolador. Utiliza el protocolo UART (Universal Asynchronous Receiver-Transmitter) para hacer más fácil enviar y recibir datos de forma inalámbrica. Algunas de sus particularidades son que tiene un tamaño pequeño con buenas características de transmisión y recepción que le brindan un alcance muy amplio, de diez metros por tratarse de un sistema local Bluetooth, tiene un bajo consumo de corriente tanto en funcionamiento como en modo de espera, las especificaciones técnicas se encuentran en la sección de apartados.

Es usado para convertir el puerto serial a Bluetooth, este dispositivo permite tener comunicación serial con otros dispositivos como teléfonos inteligentes, computadoras etc. Se puede usar solamente en modo esclavo es decir que necesita otro dispositivo que lo detecte y empareje una comunicación con él, es posible modificar los baudios de transmisión (número de cambios de estados en una señal por segundo) por medio de códigos dados por el fabricante, también posee una memoria que almacena el dispositivo que se le vínculo con anterioridad sin solicitarle una validación nuevamente y cuenta con un led indicador, si está conectado o está desconectando.

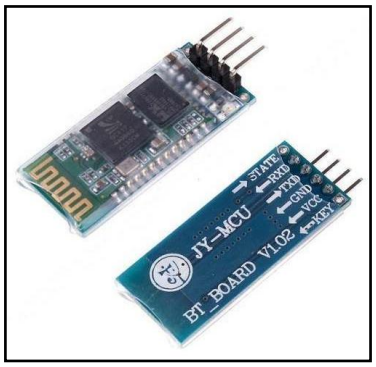


Figura 17.Bluetooth HC-06 [23]

Módulo XBee

El módulo XBee de Digi International es un transceptor inalámbrico (figura 18). El XBee utiliza un protocolo totalmente implementado para las comunicaciones de datos que proporciona características necesarias para las comunicaciones en una red de sensores inalámbricos (WSN), son dispositivos que usan la radio frecuencia para transferencia/envío de datos. Tiene características tales como direccionamiento, reintentos que ayudan a garantizar la entrega segura de datos en el nodo deseado. El XBee también tiene aplicaciones adicionales más allá de las comunicaciones de datos, puede ser usado en la vigilancia y el control de dispositivos remotos.

Figura 18. XBee [23]

El XBee viene en varias versiones, pero todas tienen un funcionamiento similar. Las diferencias entre las versiones XBee incluyen la potencia, el estilo de antena, la frecuencia de operación y capacidades de red.

La comunicación XBee y un teléfono móvil con Android se vuelve más compleja debido a que los celulares no cuentan con un chip que pueda mantener una relación con él, si se quisiera realizar dicho enlace, se deberá implementar y comprar otro dispositivo llamado IOIO que tiene que ser conectado vía USB con el celular fungiendo como puente entre Android y dispositivos externos, en este caso con XBee, haciendo que se eleve el costo del proyecto y aumente el tamaño de la estructura del circuito en la prótesis.

2.3.3 Conceptos para el sensado

En este apartado se explica brevemente los conceptos propuestos para los sensores que se van a encargar de obtener los variables de temperatura, fuerza y humedad dentro del socket autoajustable, para que puedan ser monitoreados desde el teléfono móvil con el fin de prevenir posibles daños en la piel. Deben contar con ciertas características relacionadas al tamaño, el costo y consumo de energía.

Sensor DH11

El DHT11 es un sensor de temperatura y humedad, de bajo costo. Es decir un sensor dos en uno, utiliza un sensor de humedad capacitivo y un componente de medición de temperatura NTC para medir el aire circundante que se conecta a un pequeño microcontrolador de alto rendimiento de 8 bits, que ofrece una excelente calidad, respuesta rápida, capacidad anti-interferencia y tiene un precio accesible.

Después de haber tomado la temperatura y humedad, da como salida una señal digital en el pin de datos, que asegura una alta fiabilidad y una excelente estabilidad a largo plazo. El único inconveniente de este sensor es que sólo se puede obtener nuevos datos de una vez cada dos segundos.
Cada elemento DHT11 está estrictamente calibrado en el laboratorio que es extremadamente preciso en la calibración de humedad. Los coeficientes de calibración se almacenan como programas en la memoria OTP, que son utilizados por la señal interna del sensor proceso de detección. La interfaz serie de un solo cable hace que la integración del sistema sea rápida y fácil. Su pequeño tamaño, hace que tenga un bajo consumo de energía y lo convierta en una en la mejor opción para diversas aplicaciones, incluyendo las más exigentes.

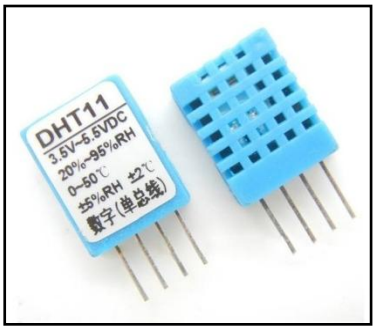


Figura 19. DHT11 [23]

Tiene cualidades importantes como las anteriormente mencionadas, que lo convierten en una parte importante del sistema para la adquisición de la temperatura y humedad dentro del socket autoajustable, al ser un dispositivo pequeño no ocupa mucho espacio y no necesita un gran sistema de acondicionamiento para el sensor, gracias a la señal de salida no es analógica sino digital.

Sensor FSR

El sensor FSR por sus siglas en ingles significa (Force Sensing Resistor) pertenece a la familia sensores piezo resistivos son similares a los sensores piezo eléctricos, pero a diferencia de estos últimos, es que al aplicar una fuerza mecánica sobre ellos, se da una variación de resistencia y no de voltaje, el comportamiento que éstos presentan se llama “efecto piezo resistivo”.
Los FSR son dispositivos con una gruesa filmina de polímero (PTF, polymerthick film) que exhibe una disminución en la resistencia cuando se da un incremento de fuerza aplicada en la superficie activa. Los FSR no son celdas de carga ni galgas extensiometricas, sin embargo poseen características similares, pero no son recomendables parara mediciones de precisión, ya que solo pueden obtenerse resultados cualitativos.

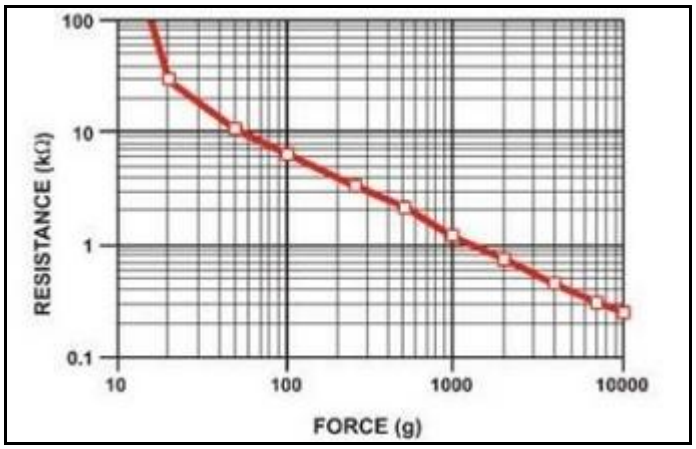


Figura 20. Resistencia-Fuerza [22]

El comportamiento típico de un sensor piezo-resistivo al aplicarle una fuerza se muestra (figura 20), en dicha gráfica se observa que conforme aumenta la fuerza aplicada al sensor, su resistencia disminuye, es decir, el valor de la resistencia es inversamente proporcional a la fuerza aplicada.

Los sensores piezo-resistivos pueden ser utilizados para mediciones dinámicas, al poseer una cierta flexibilidad, que otros sensores del mismo tipo no tienen. El propósito contar con el uso de este tipo de sensores es para realizar un arreglo con cuatro de ellos y que sean instalados dentro del socket autoajustable, cada uno de ellos este apuntando a cada cara de la extremidad sobrante, para así recabar la información de la fuerza que es aplicada en cada cara del muñón.

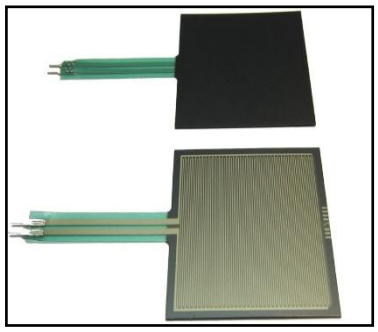


Figura 21.Sensor FSR [23]

2.3.4 Concepto reloj en tiempo real

Cuando se lleva a cabo un registro de datos es necesario saber cuándo y a qué hora ocurre cada muestreo de las señales dentro del sistema. Es muy importante tener un dispositivo confiable que brinde la seguridad que mantendrá siempre la fecha y la hora actual, que pueda hacer un seguimiento de cuando ocurrieron ciertos eventos.

Si se pierde la marca de tiempo, las mediciones realizadas solo serán datos que no servirán de mucho, no se sabrá el momento de cuando ocurrió dichos cambios, o que estaba realizando para que se presentaran dichas anomalías en las lecturas de las señales. En este apartado se propone usar un reloj en tiempo real dentro del sistema que permite mediciones de tiempo para que pueda ser fácilmente modificado desde Arduino y poder hacer un seguimiento de cuándo ocurrieron ciertos eventos. Es decir, se propone una solución vista desde hardware y no desde software, porque es posible realizarlo desde la programación pero al momento de que Arduino pierde su fuente de poder, dígase comunicación vía USB con la computadora, también pierde la función del tiempo actual programada. Es por eso que la propuesta es usar un hardware de reloj en tiempo real.

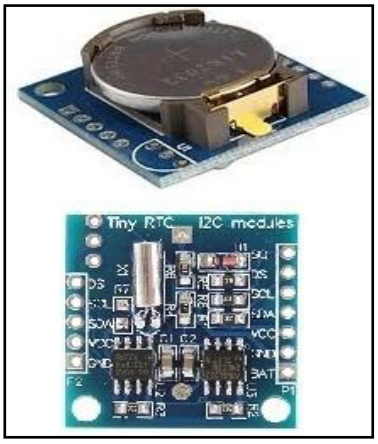


Figura 22. Reloj de tiempo real [23]

El reloj en tiempo real hace exactamente lo que su nombre dice. Su objetivo es establecer el tiempo una sola vez, y seguirá manteniendo la hora y fecha actual, hasta que se agote la batería de reloj. El concepto propuesto es utilizar una placa que cuenta con: el circuito DS1307 que es un reloj en tiempo real integrado, que logra comunicarse con la placa Arduino a través de una conexión I2C, una batería de tipo reloj de 3.0 V, que le permiten mantener el tiempo durante varios años, un cristal de 32.768 KHz que funciona como un oscilador que permite mantener la precisión del tiempo y además de un capacitor y dos resistencias. Todo esto en una pequeña placa, lista para ser montada en el Arduino.

Este dispositivo tiene la función de mantener la fecha y hora actual, en las muestras que son realizadas por los diferentes sensores (de fuerza, temperatura y humedad), para poder llevar un registro de las mismas y ser almacenadas en una tarjeta microSD.

2.3.5 Concepto para el almacenamiento de datos

Las personas que usan dispositivos electrónicos portátiles a menudo requieren almacenar sus datos de una manera que pueda ser portátil. Esto es usado especialmente en los usuarios de teléfonos móviles, sistemas portátiles de

posicionamiento global (GPS), de video, reproductores de audio y tabletas, para diversos fines como almacenar números telefónicos, imágenes, música, documentos etcétera.

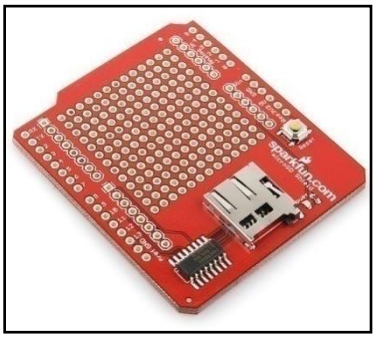


Figura 23.Placa microSD [23]

Una de las principales formas de almacenar datos en dispositivos móviles, es la tarjeta microSD, es una de las más pequeñas tarjetas de memoria flash en el mercado. El nombre proviene del fabricante que fue SanDisk, que se presta el SD al nombre y la otra parte del nombre viene del pequeño tamaño de la tarjeta.

Al estar usando la placa de desarrollo Arduino, se tiene el inconveniente de no contar con suficiente memoria para almacenar datos, es necesario recurrir a otro dispositivo de almacenamiento masivo, el cual cuente con ciertas particularidades, como poseer un pequeño tamaño y una mayor capacidad de memoria que Arduino.

Por ello se propone usar la placa *Sparkfun* microSD (figura 23), se equipa al Arduino para aumentar la capacidad de almacenamiento masivo, por lo que puede ser utilizarlo para el registro de datos.

Comunicación entre Arduino y la tarjeta microSD se consigue a través de una interfaz de SPI. Los pines del conector microSD SCK, DI y DO se desglosan a los pines SPI estándar del ATmega168/328 's digitales (11-13), mientras que el pin CS se rompe a D8 pin de Arduino. Cuenta con librerías para Arduino donde asume diferentes ejemplos para poder programar. Esta placa de Sparkfun cuenta con una ranura para insertar la tarjeta microSD, indicador de energía LED rojo y un botón de reinicio, posee pines para que sea montado a la placa Arduino.

Con esta tarjeta es posible aumentar la memoria para guardar las mediciones realizadas por los sensores de temperatura, humedad y variación de fuerza aplicada dentro del socket autoajustable y crear un archivo para llevar un registro de todas las variaciones ocurridas durante el uso del sistema.

2.4 Análisis de los conceptos propuestos

A partir de los conceptos propuestos anteriormente se hace una combinación de ellos, para dar posibles soluciones al problema, exponiendo las ventajas y desventajas de cada opción.

Combinación de conceptos

Combinación de Conceptos	Desventajas	Ventajas
Solución 1: Basic 4 Android Comunicación Bluetooth DH11, FSR, microSD	El costo de la licencia es elevado para programar una aplicación, Bluetooth tiene poco alcance.	Se programa en lenguaje Visual Basic, el teléfono celular es compatible con el Bluetooth, los sensores ocupan poco espacio.
Solución 2: Java Eclipse Comunicación XBee DH11, FSR, microSD	Programar en java se vuelve complejo, al realizar comunicación inalámbrica y otro XBee para el celular.	La plataforma es gratuita, existen ejemplos, con XBee tiene más alcance de comunicación y los sensores ocupan poco espacio.
Solución 3: Mono Comunicación Bluetooth DH11, FSR, microSD	Cuenta con muy pocos días de prueba y es necesario comprar una licencia para el programa, poco alcance de comunicación.	Se programa en lenguaje C#, cuenta con librerías, el celular es compatible con el Bluetooth y los sensores ocupan poco espacio.
Solución 4: App Inventor Comunicación Xbee DH11, FSR, microSD	El teléfono celular no cuenta con un módulo Xbee para comunicarse. No existen muchas librerías para realizar una aplicación.	Es gratuita la plataforma, es amigable a la programación, Xbee tiene más alcance y los sensores reducen el tamaño del circuito.
Solución 5: Basic 4 Android Comunicación Xbee DH11, FSR, microSD	Elevado costo de la licencia para el programa y otro módulo Xbee para el celular.	Se programa en lenguaje Visual Basic, Xbee tiene más alcance y los sensores reducen el tamaño del circuito.
Solución 6: Java Eclipse Comunicación Bluetooth DH11, FSR, microSD	Programar en java se vuelve complejo, al realizar comunicación inalámbrica, poco alcance de comunicación.	Es gratuita la plataforma, existen ejemplos, el celular es compatible con el Bluetooth y los sensores ocupan poco espacio.
Solución 7: Mono Comunicación Xbee DH11, FSR, microSD	Elevado costo de la licencia para el programa y el teléfono no es compatible con Xbee.	Se programa en lenguaje C#, cuenta con librerías, Xbee tiene más alcance y los sensores reducen el tamaño del circuito.
Solución 8: App Inventor Comunicación Bluetooth DH11, FSR, microSD	No existen muchas librerías para realizar una aplicación, poco alcance de comunicación entre el teléfono y el Bluetooth.	Es gratuita la plataforma, es amigable a la programación, el celular es compatible con el Bluetooth y los sensores reducen el tamaño del circuito.

De acuerdo con la tabla anterior, se generaron ocho posibles opciones con sus ventajas y desventajas, para escoger la mejor opción se realizó una matriz de decisión donde se tomaron en cuenta las siguientes características asignándoles un porcentaje de acuerdo a la importancia de cada uno:

Factores a tomar en cuenta	[%]
Compatibilidad de comunicación	40
Buen alcance de comunicación	30
Menor tamaño del circuito	10
Menor costo	20

En la matriz de decisión se asignó un cierto puntaje a cada solución propuesta, para el puntaje se eligió una escala del uno al tres, cada número tiene el significado de: 1-malo, 2-regular y 3-bueno, después se sumó el porcentaje de los puntos de cada solución para obtener un total y que este último número indique la mejor solución.

	Compatibilidad de comunicación (0.4)		Buen alcance de comunicación (0.3)		Menor tamaño del circuito (0.1)		Menor Costo (0.2)		Total
Opciones	Puntos	Puntos [%]	Puntos	Puntos [%]	Puntos	Puntos [%]	Puntos	Puntos [%]	[%]
Solución 1	3	1.2	2	0.6	3	0.3	2	0.4	2.5
Solución 2	2	0.8	3	0.9	2	0.2	3	0.6	2.5
Solución 3	3	1.2	2	0.6	3	0.3	1	0.2	2.3
Solución 4	2	0.8	3	0.9	2	0.2	3	0.6	2.5
Solución 5	2	0.8	3	0.9	2	0.2	2	0.4	2.3
Solución 6	3	1.2	2	0.6	3	0.3	2	0.4	2.5
Solución 7	2	0.8	3	0.9	2	0.2	1	0.2	2.1
Solución 8	3	1.2	2	0.6	3	0.3	3	0.6	**2.7**

2.4.1 Reflexión del concepto seleccionado

La matriz de decisión arrojó como mejor opción la número ocho, la cual incluye los conceptos de Bluetooth para la comunicación inalámbrica, los sensores FSR, DHT11, la tarjeta de memoria microSD y usar la plataforma App inventor para desarrollar la aplicación móvil en Android.

La bondad de usar comunicación Bluetooth es que cuenta con un pequeño tamaño, tiene un bajo costo y es compatible con los dispositivos móviles y la distancia de alcance es aproximadamente de cinco a diez metros lo cual es suficiente, porque no existe una gran distancia al momento de que el socket autoajustable empareje una comunicación con el teléfono celular del usuario.

La tarjeta de memoria microSD tiene la particularidad de poseer un pequeño tamaño y tener una gran capacidad de memoria, lo grandioso es que este

aumento de la capacidad no significa un aumento correspondiente en el tamaño físico de la tarjeta de memoria

En cuanto a los sensores, al ser pequeños, no ocupan gran tamaño dentro del socket, además el sensor de fuerza tiene un grado de flexibilidad que ayuda adaptarse a la superficie donde se coloca.

Finalmente, para el desarrollo de la aplicación, App inventor es una plataforma donde no se paga ni un peso por ocuparla, es amigable con el programador, se encuentra en una fase beta (de prueba) y cuenta con las suficientes herramientas para desarrollar una aplicación.

Capítulo 3. Configuración de la aplicación y sistema de instrumentación

3.1 Funcionamiento de la aplicación móvil

La aplicación móvil se desarrolló en una plataforma gratuita llamada App Inventor, esta herramienta cuenta actualmente con un cierto número de funciones, debido a que se encuentra en una fase beta. Se utilizó porque no necesita una licencia para hacer uso de ella y la programación es más amigable.

El monitoreo de la temperatura, la humedad y la fuerza ejercida por el muñón dentro del socket autoajustable son los principales propósitos de la aplicación.

A continuación se explica brevemente el funcionamiento de la aplicación móvil y en los apartados finales del trabajo se encontrará un manual para el uso e instalación de la aplicación.

La pantalla principal contiene los siguientes botones: conectar/desconectar, guardar/detener, lectura/monitoreo, gráfica/ocultar gráfica y el botón de salir, las funciones de cada uno de ellos son las siguientes:

-  Botón de conectar: tiene la función de conectar/desconectar el vínculo entre Arduino y el teléfono móvil. El trabajo de los demás botones depende de la conexión Bluetooth, no harán nada si no está conectado el teléfono con el Arduino. La aplicación se encarga de buscar la dirección MAC del módulo de Bluetooth para vincular ambos dispositivos (Bluetooth y celular) y así comenzar a recibir/enviar datos.

-  Botón de guardar: tiene la función de registrar y detener el almacenamiento de los datos dentro de la memoria microSD. Cuando el usuario presione este botón, mandará un mensaje indicando, que comenzará a guardar los datos y en la pantalla seguirá visualizándose los datos del monitoreo.

-  Botón de lectura: tiene la función de leer el archivo almacenado en la memoria microSD, se recomienda que el archivo sea leído en otro dispositivo para observar mejor todos los datos, debido a la gran cantidad de registros guardados. El usuario notará cuando el archivo almacenado es demasiado grande porque, al momento de visualizar los datos, los registros finales almacenados desplazarán a los del inicio. Otra característica de presionar este botón, es que en la pantalla se verán los datos almacenados y no los datos del monitoreo, esto no detiene el monitoreo, seguirá

funcionando en segundo plano y volverá a pantalla cuando se presione el mismo botón.

-  Botón de gráfica: tiene la función de mostrar y ocultar la gráfica, en ella se observa la variación de la temperatura de manera gráfica en tiempo real dentro del socket autoajustable, esta gráfica tiene como objetivo que el usuario pueda observar las variaciones de temperatura en forma gráfica (figura 25).

-  Botón de salir: como su nombre lo indica, solamente es para salir de la aplicación.

-  Botones físicos del celular (menú y return): para el botón menú, App Inventor le asigna por default la función de interrumpir la aplicación y el botón return tiene la función de salir de la aplicación.

Al iniciar la aplicación, mandará un mensaje de bienvenida y pedirá al usuario que escriba su nombre (figura 24-a), después mandará una notificación indicando la calibración de los sensores, para llevar a cabo esta tarea se debe presionar el botón cuando esté listo el usuario (figura 24-b). En la programación se propuso valores máximos en cuanto a fuerza, temperatura y humedad, para delimitar el rango permitido en los sensores, quiere decir que al ser sobrepasado dichos valores el teléfono celular mandará un tono de alerta y un mensaje indicando que se ha sobrepasado el valor calibrado.

Después de lo anterior, la aplicación realizará la pregunta ¿Desea comenzar a guardar datos en la microSD? (figura 24-c), si la respuesta es sí, la aplicación comienza a guardar los datos pero si la respuesta fue no, se tiene el botón de guardar para que se puedan almacenar los datos, en otro momento.

Figura 24. a) Nombre de usuario, b) Calibración de sensores, c) Almacenar datos

En la parte derecha de la pantalla, se visualiza el monitoreo del socket autoajustable, mostrando la fuerza ejercida en Newton sobre la cara anterior, posterior, medial y lateral, también se muestra el porcentaje humedad relativa y la temperatura en Celsius. Además, avisa si se ha sobrepasado un valor o si la tarjeta microSD no está insertada en el Arduino (figura 24-1). En la parte izquierda de la pantalla se encuentran los botones de control que anteriormente fueron explicados.

Figura 24-1. Memoria microSD no insertada

En la parte inferior de la aplicación se muestra el estado de la aplicación (figura 25), indicando la función que se está llevando a cabo, por ejemplo, si está conectado, almacenando datos, mostrando la gráfica o si se ha sobrepasado el valor calibrado de algún sensor. Es importante mencionar que la aplicación actualmente funciona solo con celulares que cuenten con sistema operativo Android. El código de programación se encuentra en los apartados finales de este trabajo.

Figura 25. Mensaje de estado, gráfico de temperatura y monitoreo

3.2 Componentes del sistema de instrumentación

Un sistema de instrumentación tiene como finalidad adquirir información del mundo físico, a la máxima velocidad posible, con la mayor exactitud que se pueda obtener y con el menor costo [32].

Un sistema de instrumentación cuenta con las siguientes etapas: transducción, acondicionamiento, procesamiento y presentación, cada una de ellas, tiene funciones y elementos específicos que procesan la información, para finalmente presentarla a un usuario final.

En la (figura 26) se muestra un esquema básico de cualquier sistema de instrumentación:

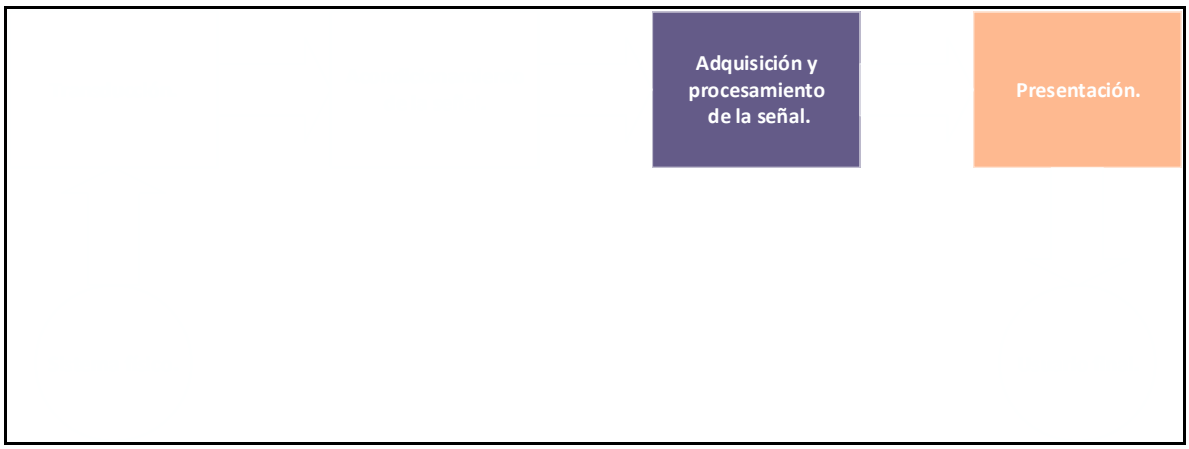


Figura 26. Componentes de un sistema de instrumentación

a) Transductor

El transductor es el componente que convierte la magnitud física a medir, en una señal eléctrica y se dividen en dos tipos:
Un primer dispositivo (llamado sensor) que situado en cierto medio, genera una señal de una determinada magnitud física convertible en otra forma física.
Un segundo dispositivo (llamado transductor) que realiza la conversión a señal eléctrica [33].

b) Acondicionamiento de la señal

Este bloque incluye todas aquellas transformaciones que deben realizarse sobre señales eléctricas que resultan en la salida del transductor, y que son previas al procesado para extraer la información que se mide o evalúa [34].

c) Procesamiento de señal

Incluye el conjunto de transformaciones a que debe ser sometida la señal eléctrica a fin de extraer de ella, la información que se busca. El procesamiento de la señal suele contener muy diversas operaciones, ya sean lineales, no lineales, de composición de múltiples señales, o de procesado digital de las señales[34].

d) Presentación de la información

La información resultante del proceso de medida debe ser presentada de forma comprensible al operador, o elaborada e integrada para que pueda ser interpretada por un sistema supervisor automático. Actualmente, los terminales alfanuméricos y gráficos basados en computadoras suelen ser el método más utilizado para presentar todo tipo de información [34].

De acuerdo a lo anterior se puede hacer una analogía con el sistema de instrumentación implementado en este trabajo.

Los sensores FSR, medirán la fuerza ejercida dentro del socket autoajustable, arrojando a la salida variaciones de voltaje, debido a que es una resistencia variable. El sensor DHT11, medirá la temperatura y humedad, lanzando una señal de salida digital.

Después seguirá el acondicionamiento de la señal, para esta etapa se utilizaron amplificadores operacionales, en específico el encapsulado LM348N. Este dispositivo electrónico servirá solamente para la señal de los sensores FSR, por que el acondicionamiento del sensor DHT11 ya lo lleva internamente y no es necesario acondicionar su señal de salida. En el siguiente capítulo se abordará con más detalle el acondicionamiento de las señales.

Para el procesamiento de ambas señales, se utilizó la tarjeta de desarrollo Arduino Uno, reservando puertos analógicos y digitales, que junto con su programación, procesará la información recabada y será transmitida al usuario final.

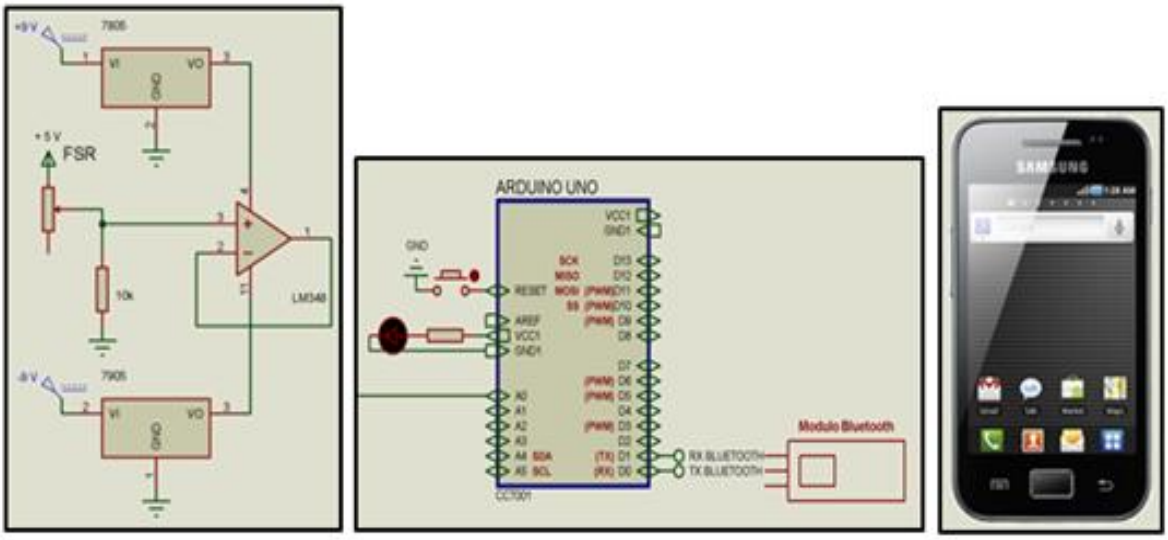


Figura 27.
a) Circuito de acondicionamiento de la señal b) Adquisición y procesamiento de datos c) Visualización de la señal

Finalmente, la información viajara a través del Bluetooth, hasta llegar al teléfono celular del usuario final, donde interpretara la información y tendrá la opción de calibrar los sensores y guardar la información en la tarjeta microSD entre otras. En la (figura 27), se observa el diagrama del sistema de instrumentación que se implementó en este trabajo.

Capítulo 4. Diseño a detalle

4.1 Especificaciones del sistema

Las especificaciones del sistema de monitoreo

Especificación	Parámetro
Alimentación simétrica	9 [Vdc] a -9 [Vdc]
Consumo aproximado	100 [mA] a 180 [mA]
Consumo máximo	200 [mA]
Temperatura de operación	0ºC – 50ºC
Microcontrolador	ATMEGA328P Velocidad de reloj: 16 [MHz] Alimentación: 5 [Vdc] 14 entradas digitales 6 salidas de PWM 8 bits 6 entradas analógicas 10 bits EEPROM de 1 [KB] Memoria Flash 32 [KB] SRAM 2 [KB] Encapsulado: 28 pines
Shield microSD	Socket para tarjeta microSD Alimentación: 3.3 [Vdc] Conexión: SPI (Serial Peripheral Interface) Led indicador y botón de reset Compatibilidad Arduino Uno Tarjeta microSD de 2 GBytes
Módulo RTC	Alimentación: 5 [Vdc] Registra segundos, minutos, horas, meses y años RAM 56-Byte, respaldada por batería. Batería de reloj:3.5 [Vdc] Conexión I^2C Consume menos de 500 [nA]en modo oscilador Rango de temperatura de trabajo: -40°C a +85°C
Módulo Bluetooth	Alimentación: 5 [Vdc] Alcance máximo:10 [m] Rango de consumo de corriente: 8mA -25 [mA] Nombre del Bluetooth: linvor Contraseña: 1234 Rango de velocidad de transmisión: 9600 baud - 38400 baud
Sensor de fuerza	Alimentación: 5 [Vdc] Rango de sensibilidad: 0.1 [N] a 10 [N] Grosor: 0.45 [mm] Tamaño:43.69 x 43.69 [mm] Resistencia en Stand-off > 10 [Mohms] Precisión: ± 5% al ± 25%. Rango de temperatura de operación:

	-30°Ca +70ºC
Sensor de temperatura y humedad	Alimentación: 5 [Vdc] Rango de temperatura: 0ºC a +50ºC Rango de humedad relativa: %20 RH a %90 RH Precisión de temperatura: +- 1ºC Precisión de humedad relativa: +- %5 RH Rango de consumo de corriente: 0.2[mA] a1[mA]
Especificaciones para la aplicación móvil	Parámetros
Dispositivo móvil	Bluetooth 2.0 Sistema operativo Android Espacio suficiente en memoria para instalar la aplicación móvil. Estar vinculado el teléfono celular con el módulo Bluetooth

4.2 Localización de los sensores

Es importante definir la localización de los sensores, porque ellos se encargan de registrar las variables de temperatura, humedad y fuerza dentro del socket autoajustable, Además se tomó en cuenta que los sensores no interfieran en ninguna de las funciones del socket autoajustable para no afectar el funcionamiento de alguna de las partes que lo conforman.

4.2.1 Localización de los sensores FSR

El objetivo de los sensores es, medir la fuerza ejercida por el muñón dentro del socket autoajustable, los cuatro sensores FSR se distribuyeron dentro del socket, uno para cada sujetador (cara posterior, anterior, medial y lateral).

El arreglo de los sensores FSR se localizará en la estructura de apoyo, específicamente sobre las barras verticales. Debido a que las barras son de un material sólido [1], se piensa que no presentará una gran deformación que afecte a los sensores, ellos estarán adheridos a cada barra vertical, evitando obstruir la unión con otras partes de la estructura como el anillo que rodea las barras u otros componentes.

Los cuatro sujetadores en la parte de atrás tienen una geometría que hace que embonen las barras verticales de la estructura de apoyo, creando un buen contacto entre los sujetadores y los sensores, para obtener una mejor medición de la fuerza aplicada dentro del socket autoajustable, también se tomó en cuenta la zona de mayor concentración de la presión se encuentra en la parte central de los sujetadores [1]. En la siguiente (figura 28) se muestra la distribución de los sensores.

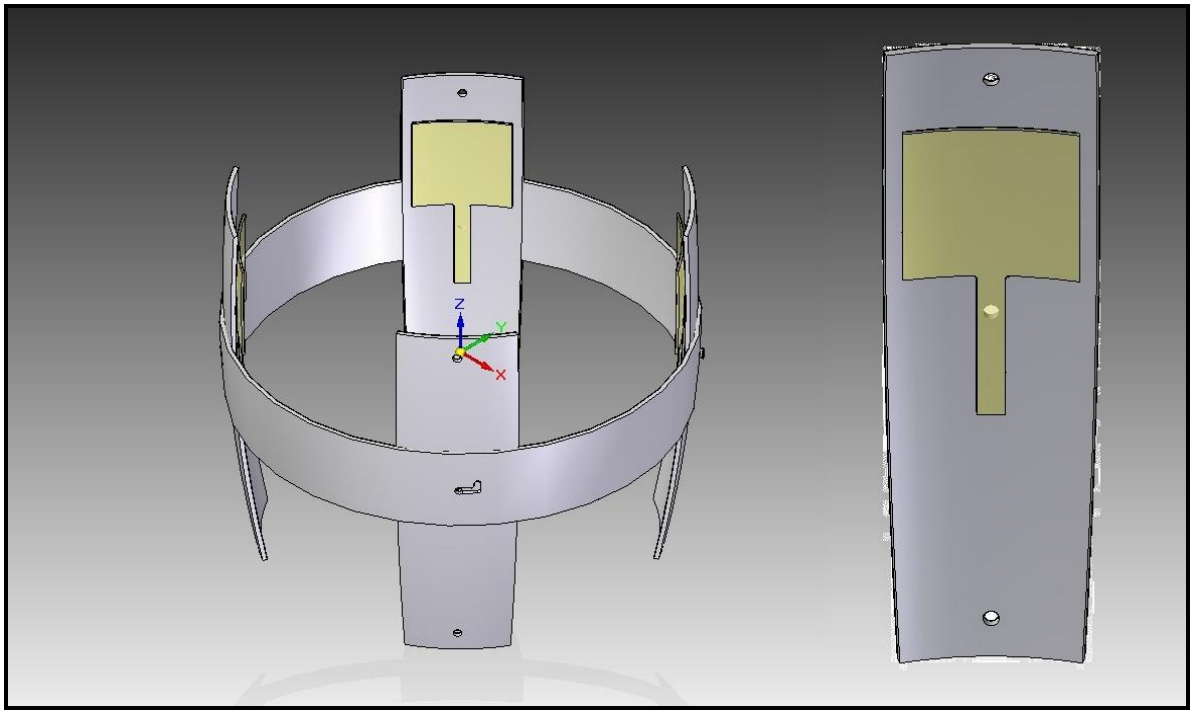


Figura 28. Distribución de los sensores FSR [1]

4.2.2 Localización del sensor DHT11

Este sensor se encargará de medir la humedad y la temperatura dentro del socket autoajustable. Pero se necesita que, la parte posterior del sensor quede libre, para que el aire circule dentro del sensor y pueda adquirir las mediciones correspondientes. Por eso se propone colocar el sensor en el sistema base, específicamente entre la malla elástica y la base [1], como se muestra en la (figura 29). La malla tiene agujeros ayudando al aire transitar del sistema de ajuste al sistema base donde se encuentra el sensor y se evita que interfiera con alguna de las partes del socket autoajustable.

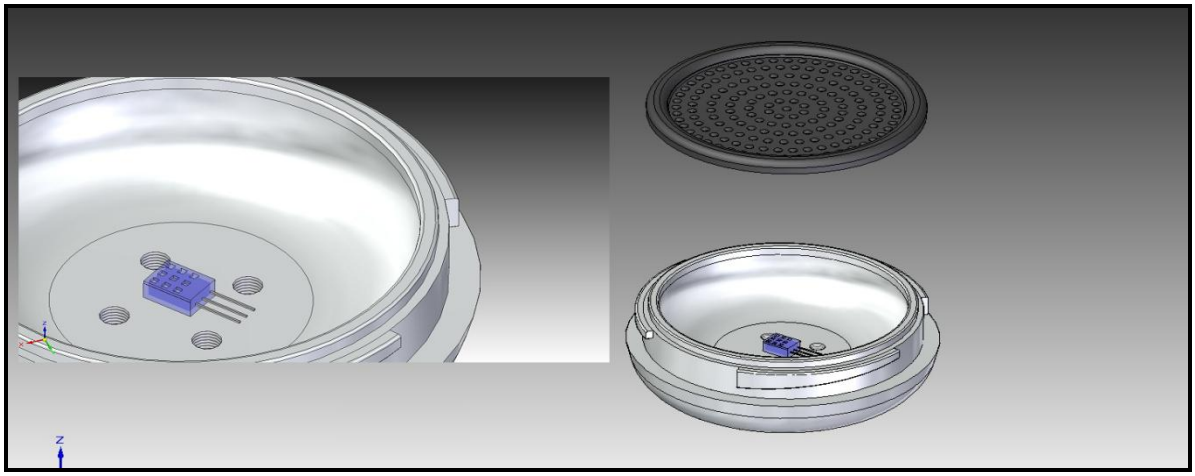


Figura 29. Localización del sensor DHT11 [1]

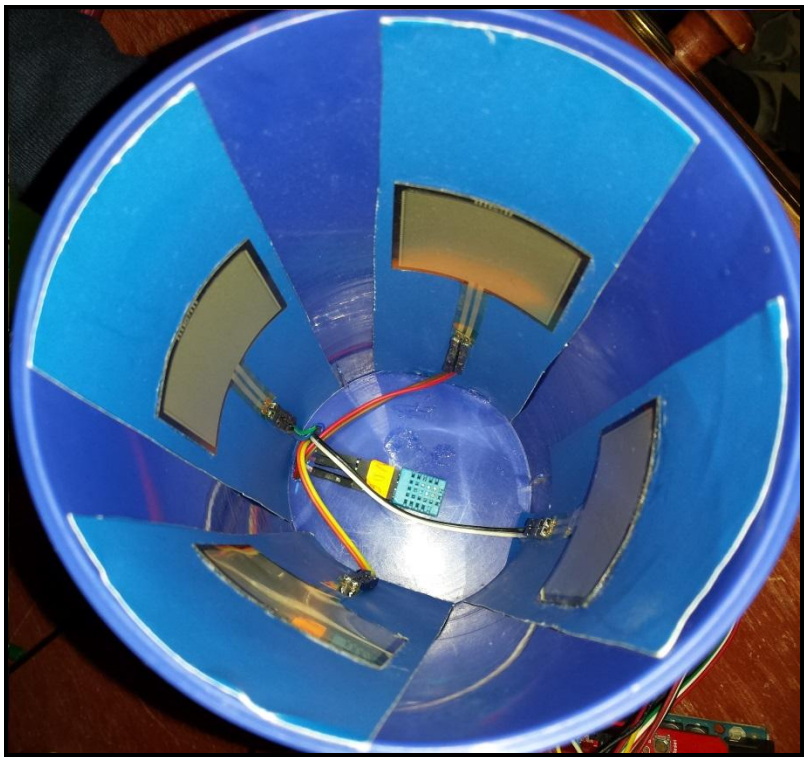


Figura 30. Localización de sensores en el modelo funcional

4.3 Instrumentación de los sensores

Para el desarrollo de la parte electrónica, es necesario contar con un circuito electrónico base que permita ser modificado fácilmente para adaptarlo a las diferentes necesidades de diseño. Actualmente los microcontroladores son los dispositivos que cubren esa característica.

Los microcontroladores son circuitos electrónicos digitales que son considerados computadoras en miniatura, ya que cuentan con un microprocesador, memoria de acceso aleatorio, memoria de programa y múltiples terminales de entrada y salida. Bajo estas circunstancias, los microcontroladores son circuitos muy flexibles que pueden ser programados para adaptarse a muy diferentes condiciones de diseño [35].

Para el presente trabajo se utiliza el microcontrolador ATmega328 montado en la placa Arduino Uno. Este dispositivo se elige debido a que se puede conseguir fácilmente en el mercado local, además de que existe una plataforma de programación libre, lo cual le confiere ventajas desde el punto de vista tanto económico como de portabilidad, también se le puede adicionar diferentes componentes electrónicos llamados *Shield*, que van montados sobre la placa Arduino Uno, formando una especie de torre (figura 31).

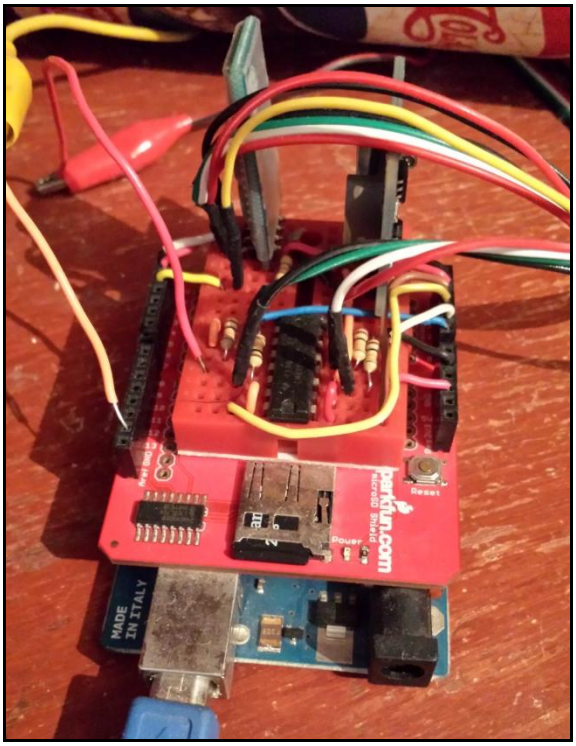


Figura 31. Placa Arduino Uno y Shield microSD

Un sensor es un dispositivo que convierte la variable física a medir en una variable de tipo eléctrica, ya sea voltaje o corriente. El acondicionamiento se ocupa del tratamiento de la variable eléctrica y el procesamiento se encarga de convertir la señal en información digital para que pueda ser desplegada o transmitida [33]. El desplegado es la representación de la información recabada y procesada que será mostrada al usuario y posteriormente, la comunicación Bluetooth se hará cargo de la transferencia de información entre el Arduino Uno y el celular.

4.3.1 Instrumentación del sensor FSR

Los sensores de fuerza resistivos están hechos con una película delgada de polímero (PTF), éstos bajan su resistencia cuando la fuerza aplicada se vuelve mayor sobre la superficie activa [36].

Los FSR no son sensores de carga, aunque presentan propiedades similares. Estos sensores no son muy recomendables para mediciones que necesiten un alto grado de precisión. Cuando son usados para mediciones dinámicas se pueden obtener únicamente mediciones cualitativas. La precisión en los rangos de fuerza se encuentra entre ± 5% al ± 25% [37]. La precisión no debe ser confundida con la resolución. La resolución de la fuerza en los sensores FSR es mucho mejor a 0.5% [37]. Existen tres modelos de sensores, lo único que los hace diferentes, es el tamaño del área activa del sensor. Los sensores son capaces de medir de 100 gramos hasta 10 kilogramos, dependiendo de las propiedades mecánicas a las cuales sea sometido [37].

En el caso de los sensores FSR produce una salida analógica, por lo que es necesario, convertir la señal analógica a digital y pueda ser leída por el Arduino Uno. Sin embargo muchos sensores solo producen señales muy pequeñas, hablando de mili volts, este tipo de señales no son lo suficientemente claras para convertirlas de analógica a digitales, es por eso que se decidió utilizar amplificadores operacionales para amplificar la señal de salida de los sensores FSR.

El amplificador operacional que se decidió utilizar fue el encapsulado LM348N (figura 32), este encapsulado consta de cuatro amplificadores operacionales independientes, de alta ganancia compensada internamente, los amplificadores operacionales tiene bajo consumo de corriente, están diseñados para trabajar con fuentes de alimentación simétricas [38]. Se decidió utilizar el encapsulado debido al ahorro de espacio en la tarjeta del circuito electrónico y además, cuenta con el número exacto de amplificadores operacionales para los sensores FSR con que se cuenta.

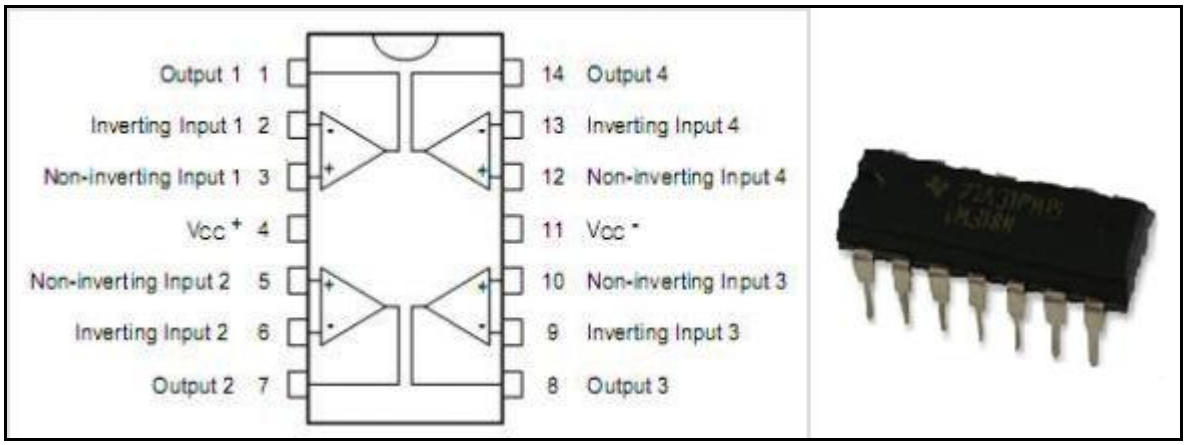


Figura 32. Amplificador operacional LM348N [38]

Para acondicionar la señal del sensor FSR se procedió a utilizar un divisor de voltaje junto con un amplificador operacional en configuración de seguidor del voltaje y la salida del amplificador operacional se conectó a un pin analógico del Arduino Uno, como se muestra en la (figura 33).

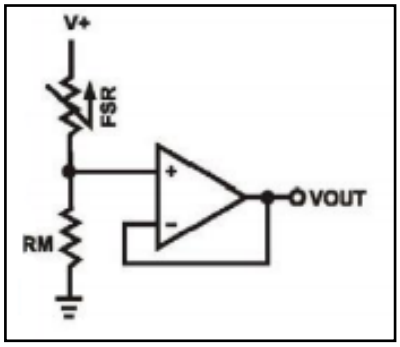


Figura 33. Acondicionamiento de la señal [37]

En el circuito anterior se realiza una conversión simple de fuerza a voltaje, el sensor de fuerza es colocado a una resistencia para formar un divisor de voltaje. La ecuación de salida de voltaje queda de la siguiente forma (figura 34):

$$V_{out} = (V +)/[1 + RFSR/RM]$$

Figura 34. Ecuación de salida de voltaje

En la configuración anterior, la salida de voltaje aumenta cuando se incrementa la fuerza. Si las resistencias son cambiadas entonces, la salida de voltaje se reducirá aumentando la fuerza. El resistor RM es elegido para maximizar en rango de sensibilidad en la fuerza y por supuesto para limitar la corriente. Finalmente se coloca un seguidor de voltaje tener una medición lo más exacta posible.

Se aplicó un voltaje simétrico de +5V a-5V para el divisor y se utilizó una resistencia de 10 Kohms, basándose en la gráfica de la hoja de especificaciones, del fabricante (figura 35).

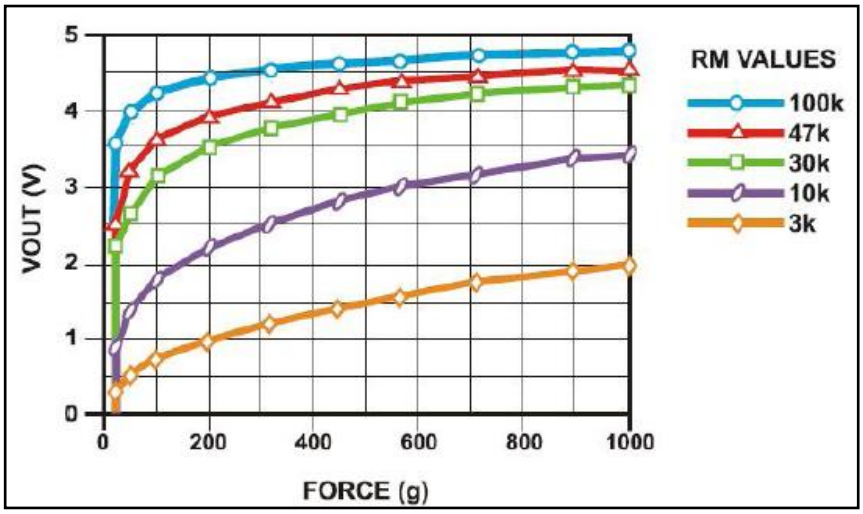


Figura 35. Gráfica de la resistencia RM, para el acondicionamiento de la señal [37]

Para la medición de la fuerza aplicada sobre los sensores, se utilizaron las gráficas de conductancia que proporciona el fabricante, porque son más lineales que las otras gráficas, estas gráficas se encuentran en la data-sheet (figura 36).

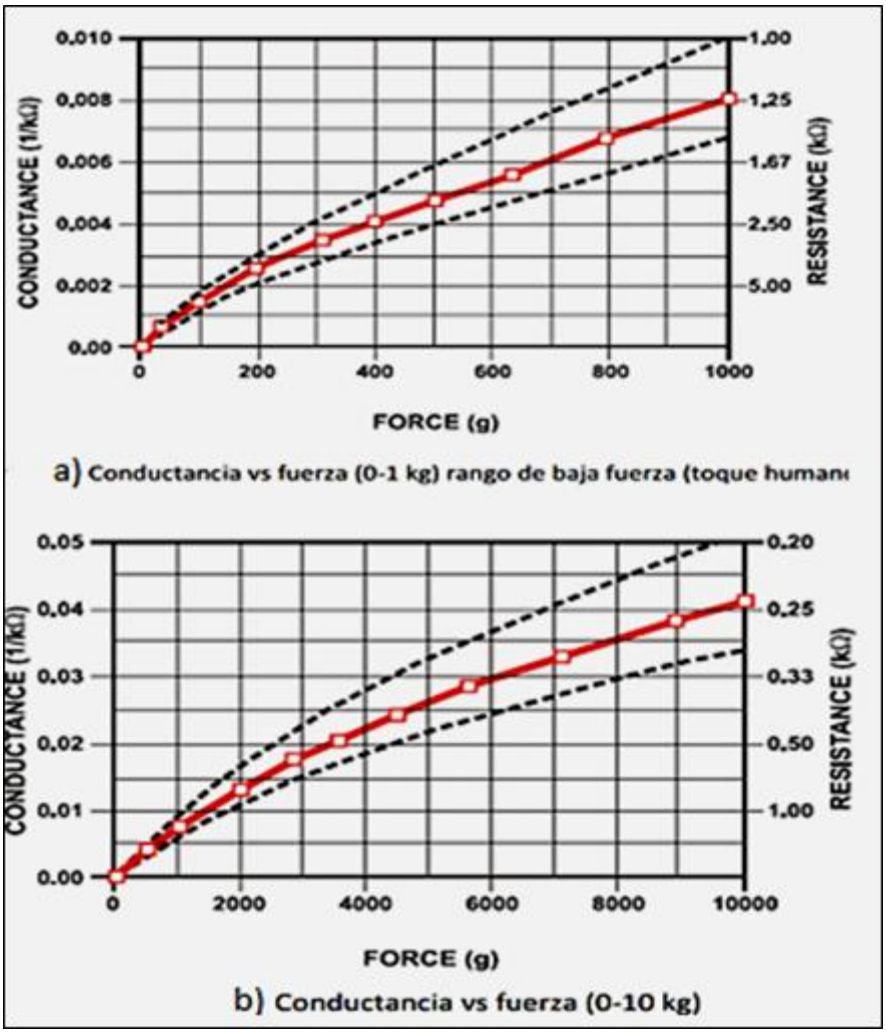


Figura 36. Gráficas de conductancia vs fuerza [37]

Para medir la fuerza aplicada el libro [45] propone lo siguiente:
En el caso que la medición sea de 0 a 1 Kg dice lo siguiente:

$$F\,[N] = C[\text{micro}\Omega]/80$$

Donde F es la fuerza aplicada, C es la conductancia del sensor FSR, cuando la medición sea de 0 a 10Kg propone que:

$$F\,[N] = C[\text{micro}\Omega] - 1000 / 30$$

Donde F es la fuerza aplicada, C es la conductancia del sensor FSR.

Las ecuaciones obtenidas por la regresión lineal de cada gráfica son las siguientes:
En el caso que la medición sea de 0 a 1 Kg: $F\,[N] = 8x10^{-7} * C[\text{micro}\Omega]$
Donde F es la fuerza aplicada, C es la conductancia del sensor FSR.

Cuando la medición sea de 0 a 10Kg propone: $F\,[N] = 4x10^{-7} * C[micro\Omega] + 0.003$
Donde F es la fuerza aplicada, C es la conductancia del sensor FSR.

Después de obtener las cuatro propuestas para medir la fuerza, dos dadas por el libro y las otras dos obtenidas por una regresión lineal, se realizaron pruebas con cada par de ecuaciones. La prueba consistió en programar en el microcontrolador la propuesta del libro, colocar distintos objetos con distintas masas sobre los sensores FSR y observar si se acerca el valor verdadero con lo que arroja la programación, haciendo lo mismo para las ecuaciones obtenidas por la regresión lineal.

4.3.2 Instrumentación del sensor DHT11

Los sensores DHT11 son unos pequeños dispositivos que nos permiten medir la temperatura y la humedad. A diferencia de otros sensores, éstos los tendremos que conectar a pines digitales, ya que la señal de salida es digital. Llevan un pequeño microcontrolador interno que realiza el tratamiento de señal.

El DHT11 se compone de un sensor capacitivo para medir la humedad y de un termistor. Ambos sensores están calibrados por lo que no es necesario añadir ningún circuito de tratamiento de señal [39]. Esto sin duda es una ventaja porque nos simplifica las cosas en el armado del circuito. Además, como los DHT11 han sido procurados en laboratorios, presentan una gran fiabilidad.

En la (figura 37) muestra la conexión entre el sensor y el Arduino Uno.

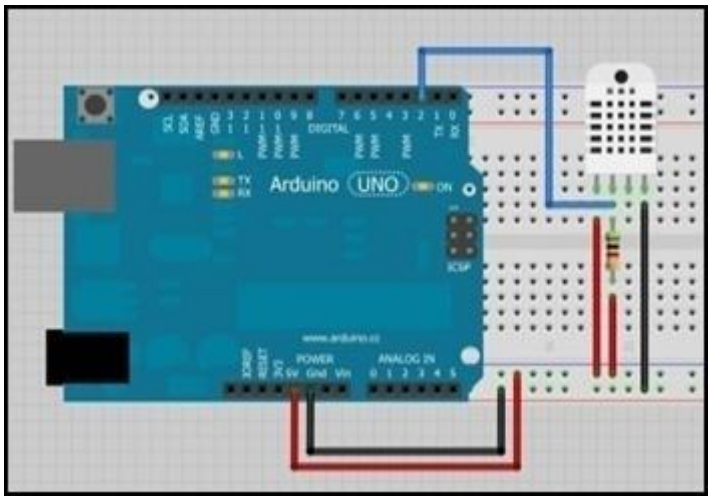


Figura 37. Acondicionamiento del sensor DHT11 [39]

Para realizar el procesamiento de la información se usó un puerto digital del Arduino Uno, el cual se encargará de cambiar la variable eléctrica a un valor proporcional binario con las unidades adecuadas de medición y enviarlo vía Bluetooth al teléfono móvil.

4.4 Adquisición y almacenamiento de datos

La tarjeta de adquisición de datos es un elemento indispensable, su importancia radica en poseer un elemento que obtenga las señales y sean procesadas.

En este trabajo se utilizó Arduino, que es una plataforma de hardware libre, basada en una placa con microcontrolador y entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares. Arduino se conecta a la computadora con cable USB estándar y contiene todo lo necesario para programar la placa. Una de las ventajas es que se puede añadir diferentes elementos llamados shields. Arduino tiene el encapsulado ATmega328, que consta de 14 pines de entrada/salida de los cuales 6 se pueden usar como salidas PWM, 6 entradas analógicas y un reloj de 16 MHz[40].

Al tener dos diferentes tipos de sensores, la adquisición de datos se dividió en dos tipos, para los sensores FSR y para el sensor DHT11.

Para los sensores FSR se utilizó el módulo ADC (convertidor analógico digital) integrado en la placa Arduino Uno, reservando cuatro puertos analógicos (A0-A3), para la entrada de los datos análogos y convertirlos en digitales. Para el sensor DHT11 se configuró el puerto digital (D2) para la entrada de datos y se utilizó la librería que proporciona Arduino para este sensor. Finalmente ambas señales son enviadas vía Bluetooth al teléfono celular.

El microcontrolador se encarga de la etapa de procesamiento, en el cual se aplican los algoritmos necesarios para tratar la señal, se utilizan las librerías correspondientes para representar aquellos valores de fuerza, temperatura y humedad. El código del programa se encuentra en los apartados finales de este trabajo.

Para la etapa de almacenamiento de datos, se usó el shield microSD, esta tarjeta va montada sobre la placa Arduino Uno y en la programación se utilizó la librería que el fabricante proporciona para Arduino. Al momento de almacenar los valores en la memoria, el nombre del archivo creado, se le es asignado directamente en la programación. Para este trabajo, el archivo se llama *test* con la extensión .txt, el nombre solamente se puede modificar directamente en el código.

El archivo también almacena la fecha y la hora del muestreo, este recurso lo toma del componente RTC, instalado sobre la placa Arduino Uno. Este componente también cuenta con su propia librería para Arduino y se programa solo una vez para dejar fija la fecha y hora. La característica importante del RTC es que al momento de quedarse sin energía la placa Arduino, el RTC no pierde la fecha y hora ya fijados anteriormente en su memoria, gracias a la pila de reloj con la que cuenta.

Finalmente los pines en la tarjeta Arduino Uno quedaron repartidos de la siguiente manera (figura 38): los pines de color rojo están conectados al shield microSD, los de color azul pertenecen al módulo Bluetooth, los de color verde están conectados a los sensores FSR, los de color anaranjado son del RTC y por último los de color rosa pertenecen al sensor DHT11.

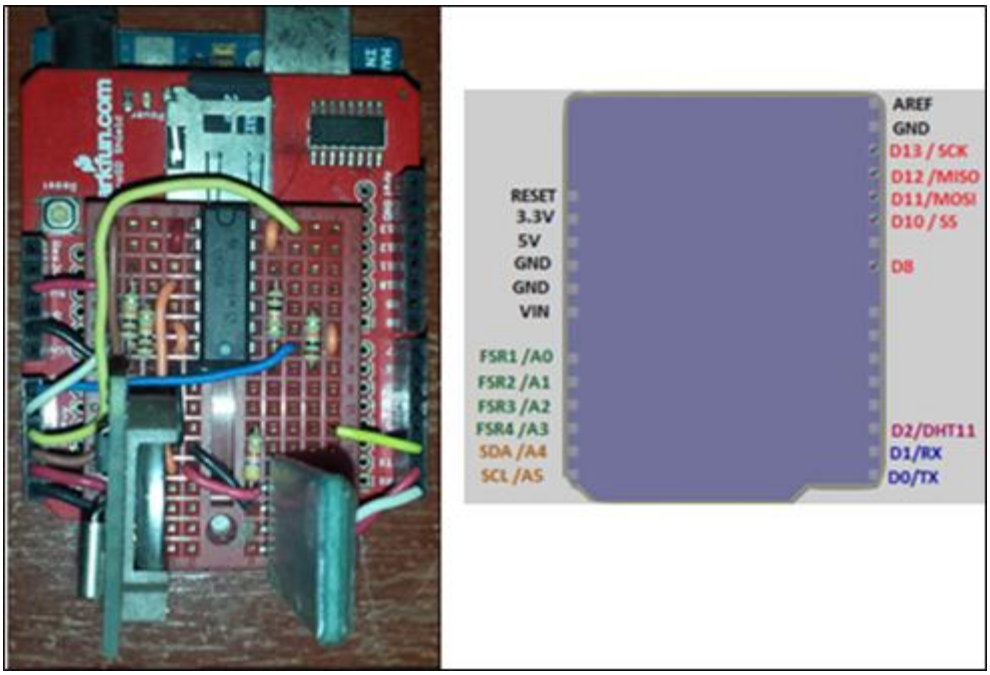


Figura 38. Pines en la tarjeta Arduino Uno

4.5 Partes de la aplicación móvil

La aplicación de monitoreo se desarrolló mediante bloques de programación, cada uno tiene una función propia y desempeña un papel específico; cada fragmento es un segmento único que contribuye al funcionamiento general de la aplicación. Es importante mencionar que algunos de estos elementos son el punto de entrada para que el usuario interactúe con la aplicación y en muchos casos unos componentes dependen de otros.

Desde el punto de vista de la programación, las principales componentes que conforman la aplicación son.

- Cliente Bluetooth: Realiza la función de vincular el móvil con el módulo Bluetooth del Arduino Uno, teniendo como características detectar la MAC del otro dispositivo, para enviar/recibir texto.
- Botones y etiquetas: Los botones son el medio por el cual el usuario puede manipular la aplicación y las funciones del Arduino, cada uno con funciones diferentes.

- Reloj: este componente es un contador de tiempo, que activa un determinado evento a intervalos regulares. Es empleado como temporizador (timer): una vez definido el intervalo de tiempo (1200ms), el temporizador se disparará en cada intervalo, recibiendo los datos del Bluetooth y mostrándolos en pantalla.
- Notificaciones: Las notificaciones son mensajes instantáneos que aparecen en pantalla para avisar de ciertas funciones activadas/desactivadas. Por ejemplo si está apagado el Bluetooth o si está conectado, si se han calibrado los sensores entre otras funciones
- Sonido: Es un sonido que es programado para alertar al usuario que se ha sobrepasado el valor calibrado y puede ir acompañado de la vibración del teléfono.
- Imagen: Este componente se utilizó para mostrar la gráfica de la temperatura. Debido a que App Inventor no cuenta con un objeto que dibuje una gráfica, se recurrió a una herramienta llamada Google Charts, la función de esta herramienta es que puede generar un gráfico a partir de un URL (en inglés Uniform Resource Locator), es decir genera una dirección web, con los datos y características específicas como color, tamaño, tipo de grafica etc. Una vez teniendo la dirección web solo basta pegarla en un navegador web para que aparezca la gráfica.

  En la (figura 39) muestra en modo general los componentes contribuyen al funcionamiento de la aplicación.

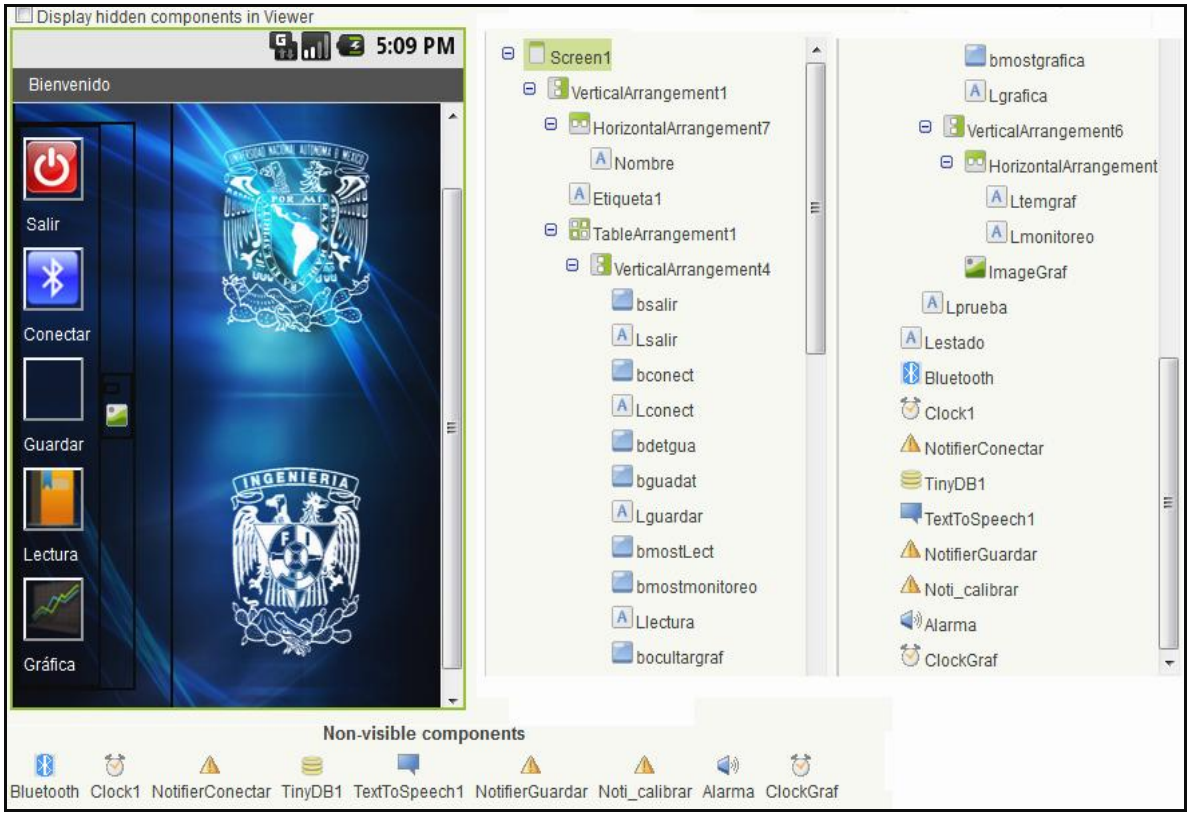


Figura 39.Componentes de la aplicación móvil [41]

En la pantalla también se puede dividir en las siguientes partes: área de botones, área de estado y área de monitoreo.

- Área de botones: En este espacio, se localizan los diferentes botones que el usuario utilizará, cada uno con funciones específicas ya mencionadas anteriormente.
- Área de estado: En esta parte, el usuario visualizará el estado de la aplicación, como la conexión Bluetooth (activada/desactivada), escritura, lectura de datos, si se ha sobrepasado algún valor, etc.
- Área de monitoreo: En este espacio, el usuario puede visualizar en tiempo real, la lectura de los sensores instalados dentro del socket autoajustable, además de avisar con un tono, si se sobrepasa el valor calibrado en los sensores, también la lectura de los datos almacenados y la gráfica.

Figura 40. Aplicación instalada en el móvil, junto con los sensores instalados en el modelo funcional y los componentes montados sobre la placa Arduino Uno

Capítulo 5. Conclusiones y trabajo a futuro

5.1 Resultados y conclusiones

Para el diseño y desarrollo de la aplicación se eligió App Inventor, porque es una plataforma gratuita, donde se puede construir aplicaciones mediante la selección de los componentes y luego decirles qué hacer. Además su programación es amigable porque utiliza un editor de bloques para ensamblar el comportamiento adecuado de los elementos, acorde con las acciones efectuadas por el usuario o la misma interacción con otros componentes.

Una de las pequeñas delimitaciones al estar desarrollando la aplicación, es que no se pudo agregar una segunda pantalla, debido a que se perdía la comunicación Bluetooth en el momento de pasar de una pantalla a otra, debido a esto se manipularon los elementos en pantalla ocultándolos y mostrándolos para que no interfirieran con otros componentes.

El caso de la gráfica de temperatura, los datos arrojados por el sensor son añadidos continuamente a una dirección URL que se encarga de dibujar la gráfica, ya que si no se hiciera esto, la imagen sería totalmente estática y no cambiaría continuamente. La herramienta que se usó, se llama Google Chart y es muy versátil, ya que se puede utilizar en cualquier dispositivo que cuente con internet, por ende, si el celular no cuenta con una conexión a internet, no será posible visualizar la gráfica. La gráfica de temperatura ayuda al usuario a visualizar la temperatura registrada durante el uso de la aplicación y el socket autoajustable además de llevar un control de cuándo dejar usar el socket autoajustable para darle un respiro al muñón y regresar a la temperatura correcta.

La tarjeta Arduino Uno, no cuenta con la suficiente memoria EPROM, limitando el tamaño del archivo donde se almacenan los datos adquiridos, es por ello que se propuso utilizar el shield microSD para aumentar dicha memoria. El registro que se almacena en la memoria microSD contiene los datos de la fuerza aplicada, porcentaje de humedad relativa, temperatura, fecha y hora en la que se registró y el nombre del usuario. El circuito en el modelo funcional posee ciertas dimensiones que ayudaron experimentar en él, pero se puede dimensionar y crear una placa con todos los componentes utilizados para reducir el tamaño del circuito y adaptarlo a la parte inferior del socket autoajustable o en alguna otra parte donde no interfiera con otros componentes.

Para medir la fuerza aplicada, se trabajó con las gráficas de conductancia contra fuerza, porque son más lineales en comparación con las otras gráficas que proporciona el fabricante. Las gráficas corresponden a diferentes ensayos de fuerza sobre el sensor, la primera corresponde a la prueba con un peso de hasta de 1 Kg y la segunda fue con un peso de hasta 10 Kg, cabe mencionar que estas pruebas fueron realizadas por el fabricante y las gráficas resultantes son expuestas en su data-sheet. Se tomaron en cuenta dos opciones la primera fue

utilizar la propuesta del libro, que provee dos ecuaciones para medir la fuerza de acuerdo al rango planteado en el data-sheet y la segunda fue obtener la regresión lineal por cada una de las dos gráficas.

Al probar las ecuaciones obtenidas de la regresión lineal, se observó que no se acercaban al valor real de la fuerza aplicada sobre el sensor y al experimentar con lo que propone el libro, se aproximó más al valor real de la fuerza aplicada sobre el sensor, es por esto que se utilizó en la programación las ecuaciones que propone el libro. Se usaron ambos rangos (0-1 Kg, 0-10 Kg) debido a que no se sabe la magnitud de la fuerza que pudiera actuar dentro del socket autoajustable.

La importancia de monitorear la fuerza aplicada del muñón sobre la estructura, es que el usuario puede prevenir daños sobre la piel del muñón, debido a que está expuesta a diferentes factores como cambios de humedad, temperatura y al constante contacto o roce con el socket autoajustable, todo esto puede causar deformaciones o heridas, que junto con otras enfermedades como la diabetes aumentaría el tiempo de sanación de la extremidad dañada o hasta tener otras complicaciones.

Al presionar el botón de calibrar, los sensores de fuerza le indican a la aplicación que el muñón está cómodo dentro del socket autoajustable, la aplicación previene al usuario por si alguna de las cuatro caras del muñón ejerce una mayor fuerza de lo calibrado al inicio. El cambio de magnitud de la fuerza puede significar que el usuario ha cambiado de posición, si está recargando más en esa cara debido al cansancio o cambio de forma del muñón etc. Esto traería como consecuencias problemas en la forma del muñón e incluso daños en la piel debido al mayor roce de dicha cara que junto con la humedad y una elevada temperatura causarían daño.

Acerca del modelo funcional, los cuatro sensores se adecuaron bien al modelo, debido a que son flexibles, el inconveniente de estar en el modelo, es que presentaron una pequeña flexión, pero es despreciable en las mediciones de fuerza. Dicho modelo sirvió para realizar las pruebas y corroborar que la aplicación móvil funcionara de acuerdo con lo programado.

Finalmente, el beneficio de tener la aplicación de monitoreo, es prevenir posibles heridas en la piel del muñón, alarmar al usuario, cuando se presente un cambio de fuerza en alguna de las caras, monitorear numérica y gráficamente las diferentes variables que se explicó anteriormente y llevar un registro en la memoria microSD.

Además se alcanzaron los objetivos de monitoreo de las diferentes variables dentro del socket autoajustable, la temperatura se pudo monitorear tanto numérica como gráficamente, se consiguió tener una de alarma si se sobrepasan los valores preestablecidos y se espera que los sensores puedan ser instalados en un prototipo del socket autoajustable, también que el usuario pueda usar la aplicación para que se realicen pruebas y con ello se haga una retroalimentación tanto para el sistema de sensores como para la aplicación móvil.

5.2 Trabajo a futuro

Se plantea como trabajo a futuro los siguientes puntos:

- Desarrollar una versión con nuevas mejoras y compatible con otros sistemas operativos móviles como iOS de Apple, Windows Phone de Microsoft entre otros.

- Contar con la instalación de la aplicación móvil y del sistema de sensores que se plantea en este trabajo, en un prototipo para llevar a cabo pruebas en usuarios.

- Unificar todos los sistemas en uno solo, para mejorar las dimensiones del sistema electrónico y no proporcione un excedente de peso y volumen al socket autoajustable.

- Realizar un estudio de factibilidad de producción y costo del sistema de sensores y del desarrollo de la aplicación móvil.

APÉNDICE

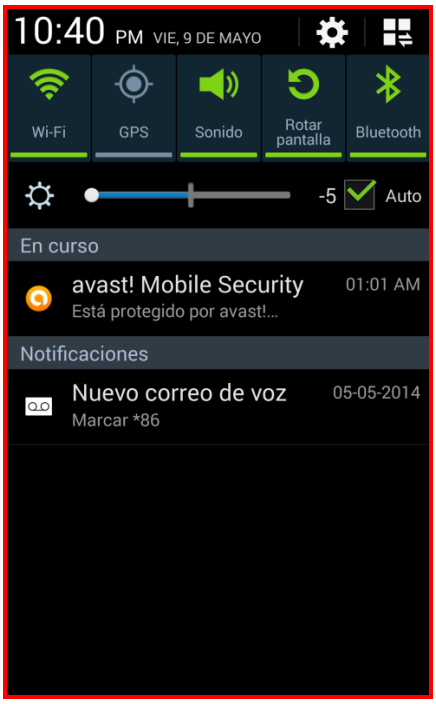
A1. MANUAL DE LA APLICACIÓN
A2. CÓDIGO DE PROGRAMACIÓN Código comentado de la aplicación en App Inventor
A3. CÓDIGO DE PROGRAMACIÓN Código comentado del microcontrolador ATMEGA328P
A4. CIRCUITO
A5. ESPECIFICACIONES Módulo Bluetooth HC-06
A6. ESPECIFICACIONES Sensor FSR 406
A7. ESPECIFICACIONES Sensor DHT11
A8. ESPECIFICACIONES Real time clock DS1307
A9. ESPECIFIACIONES Microcontrolador ATMEGA328P

A1. MANUAL DE LA APLICACIÓN

1. Encender el Bluetooth del teléfono móvil

2. En el celular buscar y vincular el Bluetooth del Arduino con el celular

   2.1 Buscar el dispositivo llamado “linvor”
   2.2 Al momento de vincular el dispositivo le va a pedir una contraseña, debe ingresar la siguiente: 1234



Paso 1

Paso 2

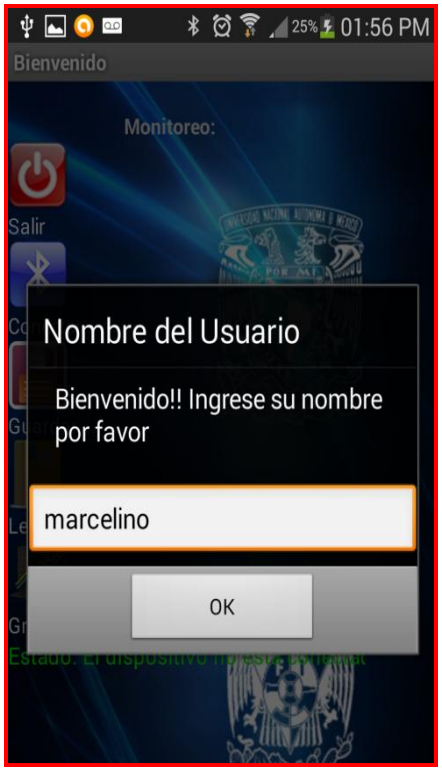
3. Descargar e instalar la aplicación (el archivo .apk)

4. Abrir la aplicación, e ingresar nombre de usuario y oprimir el botón *Aceptar*

5. Presionar el botón *Conectar* para iniciar el monitoreo

   Nota:

   **Los pasos 1-4 se hacen solamente una vez, ya después solo se deberá encender el Bluetooth y abrir la aplicación.*



Paso 4

Paso 5

6. Enseguida, en la pantalla aparece el letrero de calibrar los sensores. Presionar el botón *Calibrar*, cuando el usuario se sienta cómodo con el socket autoajustable.

7. Después, la aplicación pedirá si se desea comenzar a guardar los datos, se puede decidir entre sí o no. Presionar el botón *Si*, para comenzar a guardar los datos en la memoria microSD, provocando que el botón *Guardar* cambie de imagen y de leyenda, ahora dirá *Detener guardar*. Presionar el botón *No* para almacenar los datos en otro momento.

Paso 6

Paso 7

8. A continuación, aparecerá la siguiente pantalla, donde se muestra: la fuerza aplicada sobre cada sujetador, la hora/fecha actual, temperatura y humedad relativa. También se muestra el estado de la aplicación y los botones de control.

9. Si se presiona el botón *Guardar*, comenzará a almacenar los datos adquiridos, siempre y cuando al inicio de la aplicación no se haya presionado la opción de "no almacenar" mencionado en el punto 6. Al presionar dicho botón cambiará de imagen y ahora con ese mismo botón se puede detener el almacenamiento de los datos.

Notas:
**No extraer la memoria microSD mientras se está almacenando los datos, ya que se puede dañar severamente la memoria y el Arduino.*

**Antes de presionar el botón Guardar, se debe asegurar de tener insertada una memoria microSD en el Arduino. SI no está insertada, la aplicación le avisara que no tiene memoria para almacenar.*

**En caso de que no esté insertada la microSD, salga de la aplicación, inserte la tarjeta y vuelva a iniciar la aplicación.*

Paso 8

Paso 9

10. Presionar el botón *Mostrar datos*, para mostrar lo almacenado en la tarjeta microSD, esta función tarda aproximadamente cinco segundos en mostrar la información. Cabe mencionar que si el archivo almacenado es muy grande no podrá mostrar toda la información, es por eso que se recomienda leer el archivo en una PC.

11. Presionar el mismo botón, para ocultar dicha información y regresar al monitoreo de los sensores.

12. Presionar el botón *Gráfica*, para mostrar la gráfica de temperatura. La pantalla automáticamente cambiara a posición horizontal para que se pueda visualizar mejor la gráfica.

Paso 12

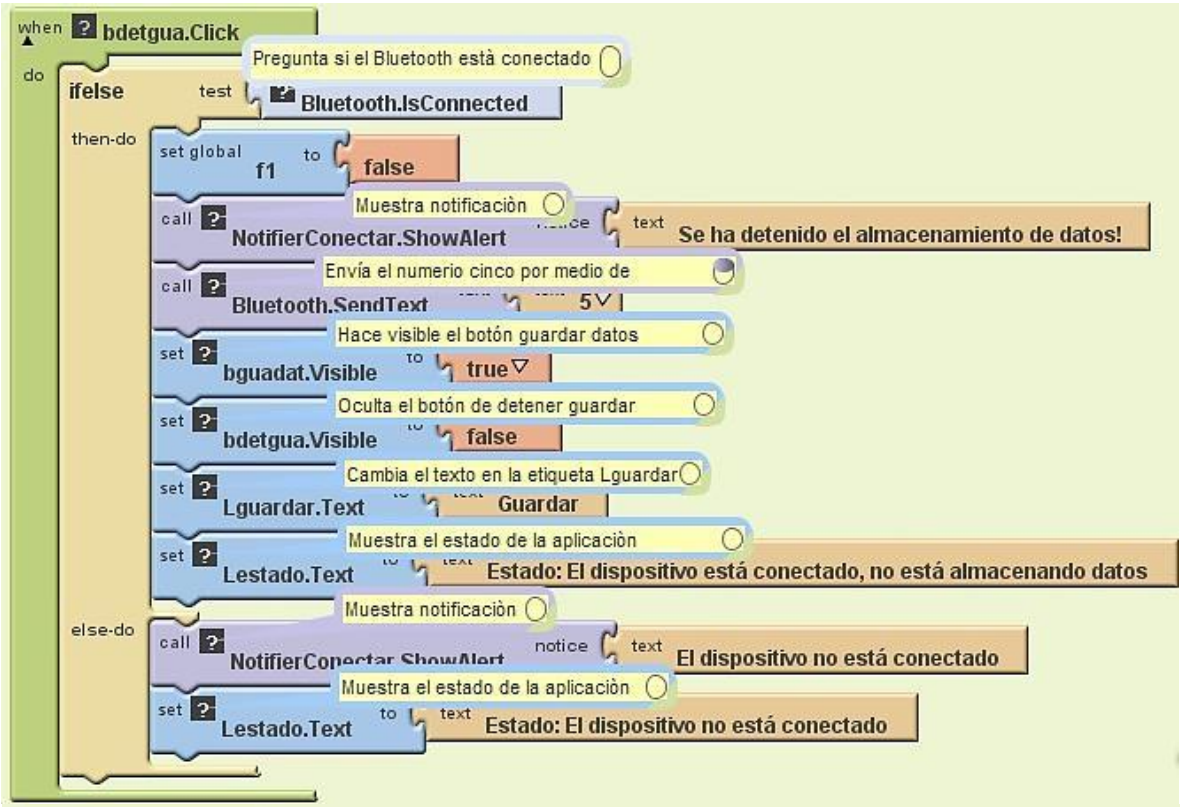
Nota:

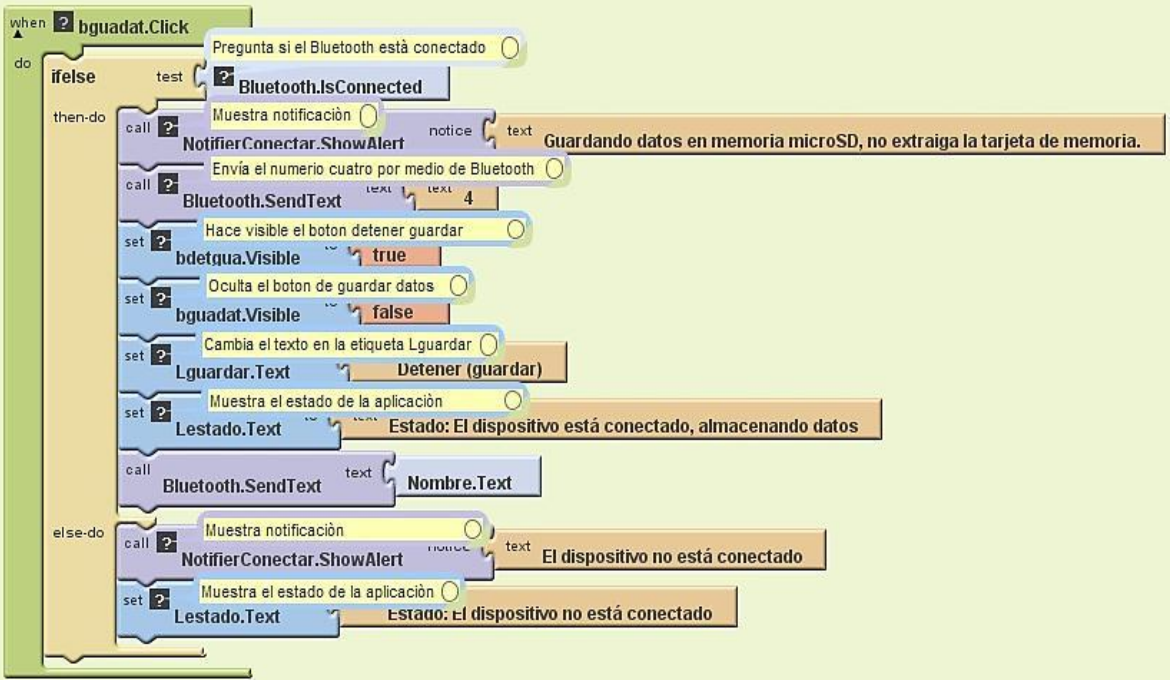
**No se podrá visualizar el monitoreo de los sensores cuando este mostrando la gráfica, debido a que ocupa toda la pantalla. Para visualizar los datos se deberá ocultar la gráfica presionando el botón Ocultar gráfica.*

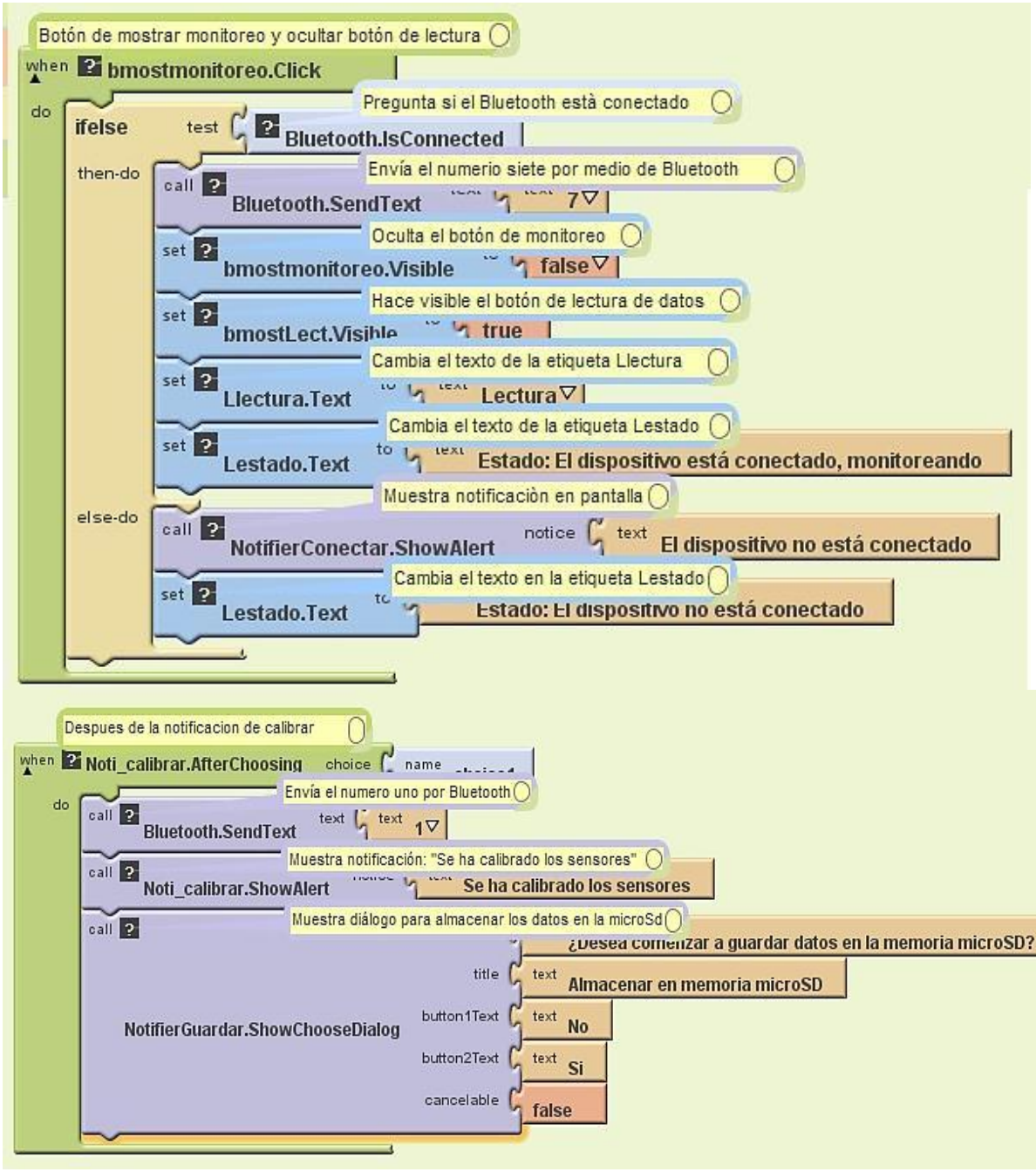
13. Presionar el botón *Salir*, para cerrar la aplicación. Si se encuentra conectado el Bluetooth, el mismo botón *Salir* hará la desconexión automáticamente.

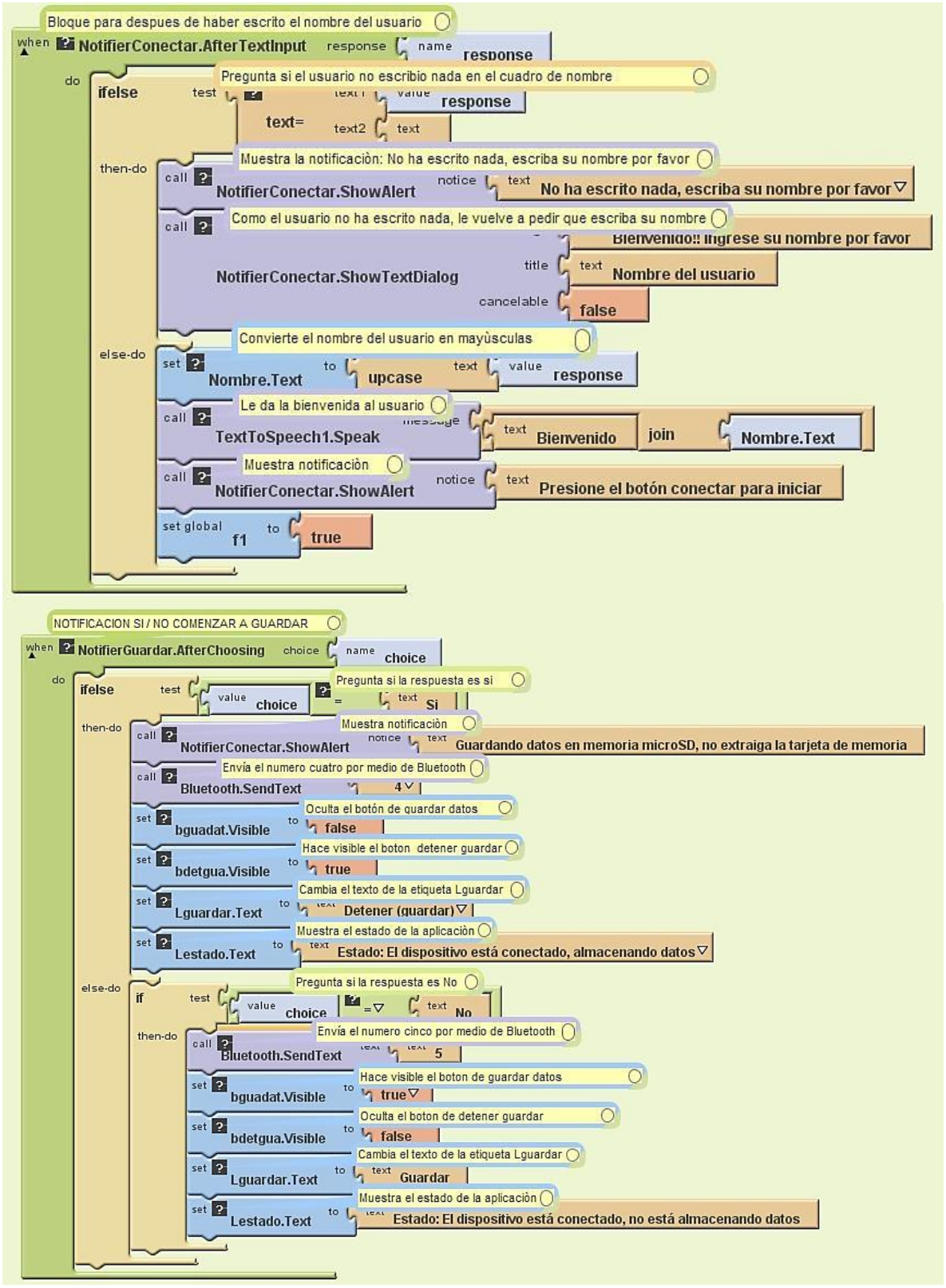
A2.CÓDIGO DE PROGRAMACIÓN

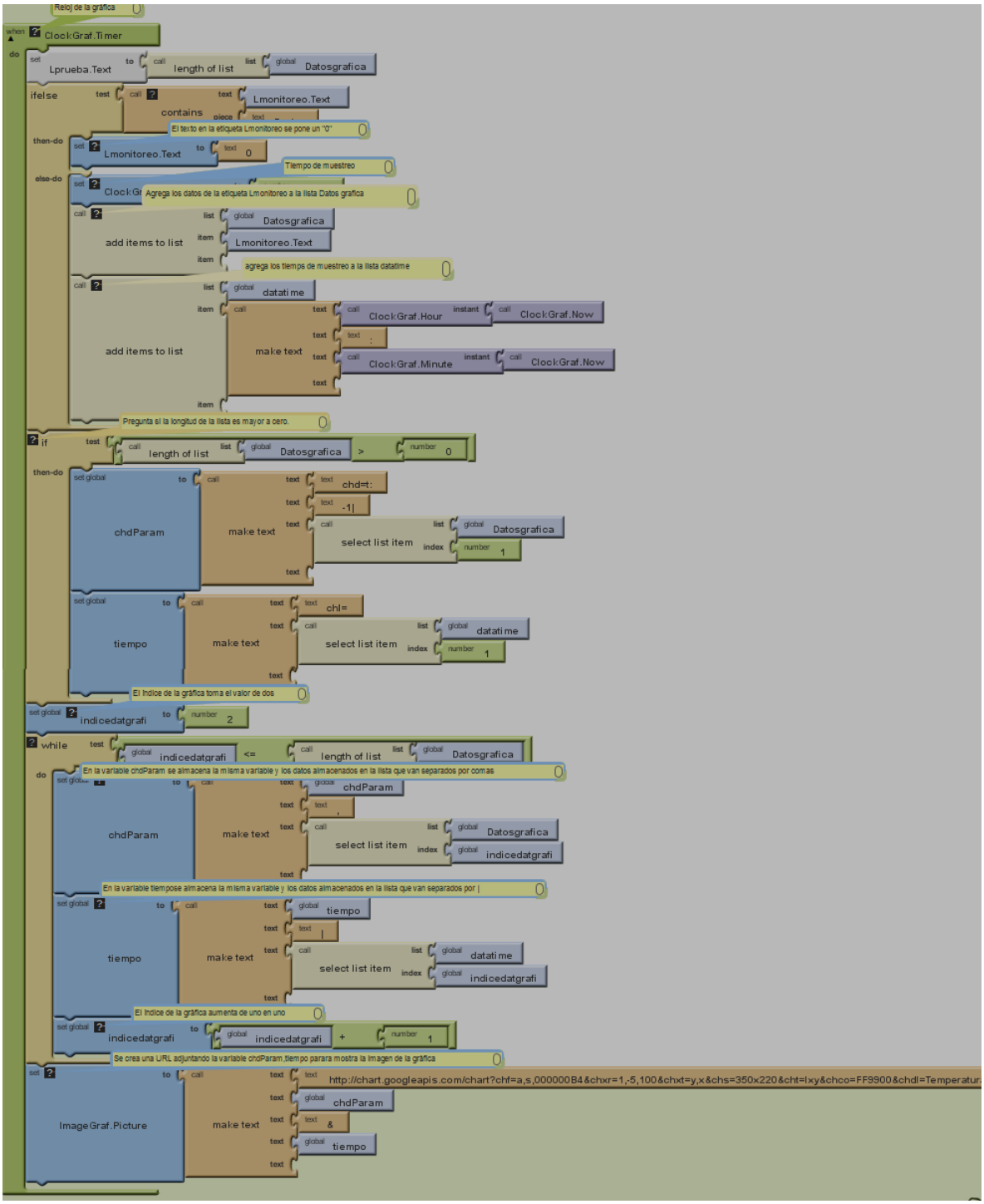
Código comentado de la aplicación en App Inventor

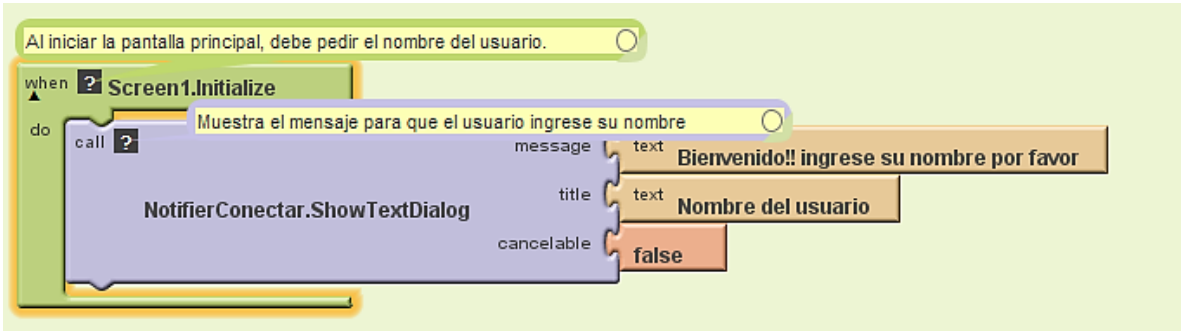

when bdetgua.Click
do
Pregunta si el Bluetooth està conectado
ifelse test Bluetooth.IsConnected
then-do
set global f1 to false
Muestra notificaciòn
call NotifierConectar.ShowAlert notice text Se ha detenido el almacenamiento de datos!
Envía el numerio cinco por medio de
call Bluetooth.SendText 5
Hace visible el botón guardar datos
set bguadat.Visible to true
Oculta el botón de detener guardar
set bdetgua.Visible false
Cambia el texto en la etiqueta Lguardar
set Lguardar.Text Guardar
Muestra el estado de la aplicaciòn
set Lestado.Text Estado: El dispositivo está conectado, no está almacenando datos
else-do
Muestra notificaciòn
call NotifierConectar.ShowAlert notice text El dispositivo no está conectado
Muestra el estado de la aplicaciòn
set Lestado.Text to text Estado: El dispositivo no está conectado

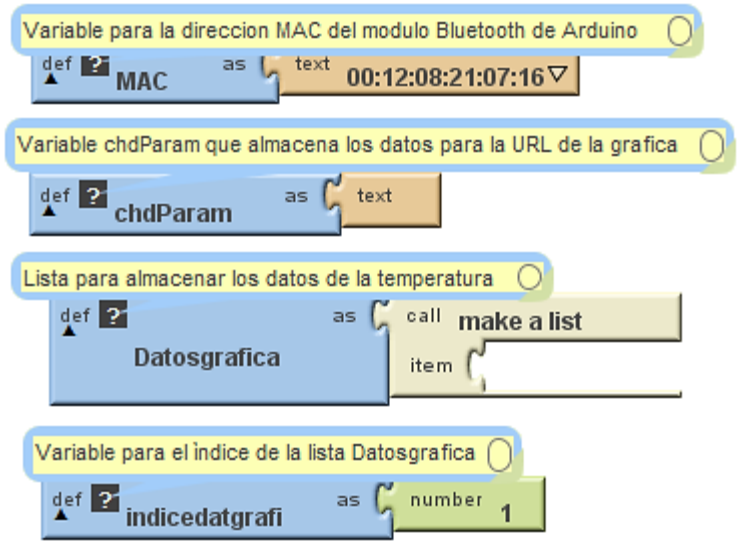

when bguadat.Click
do
Pregunta si el Bluetooth està conectado
ifelse test Bluetooth.IsConnected
then-do
Muestra notificaciòn
call NotifierConectar.ShowAlert notice text Guardando datos en memoria microSD, no extraiga la tarjeta de memoria.
Envía el numerio cuatro por medio de Bluetooth
call Bluetooth.SendText 4
Hace visible el boton detener guardar
set bdetgua.Visible true
Oculta el boton de guardar datos
set bguadat.Visible false
Cambia el texto en la etiqueta Lguardar
set Lguardar.Text Detener (guardar)
Muestra el estado de la aplicaciòn
set Lestado.Text Estado: El dispositivo está conectado, almacenando datos
call Bluetooth.SendText text Nombre.Text
else-do
Muestra notificaciòn
call NotifierConectar.ShowAlert text El dispositivo no está conectado
Muestra el estado de la aplicaciòn
set Lestado.Text Estado: El dispositivo no está conectado

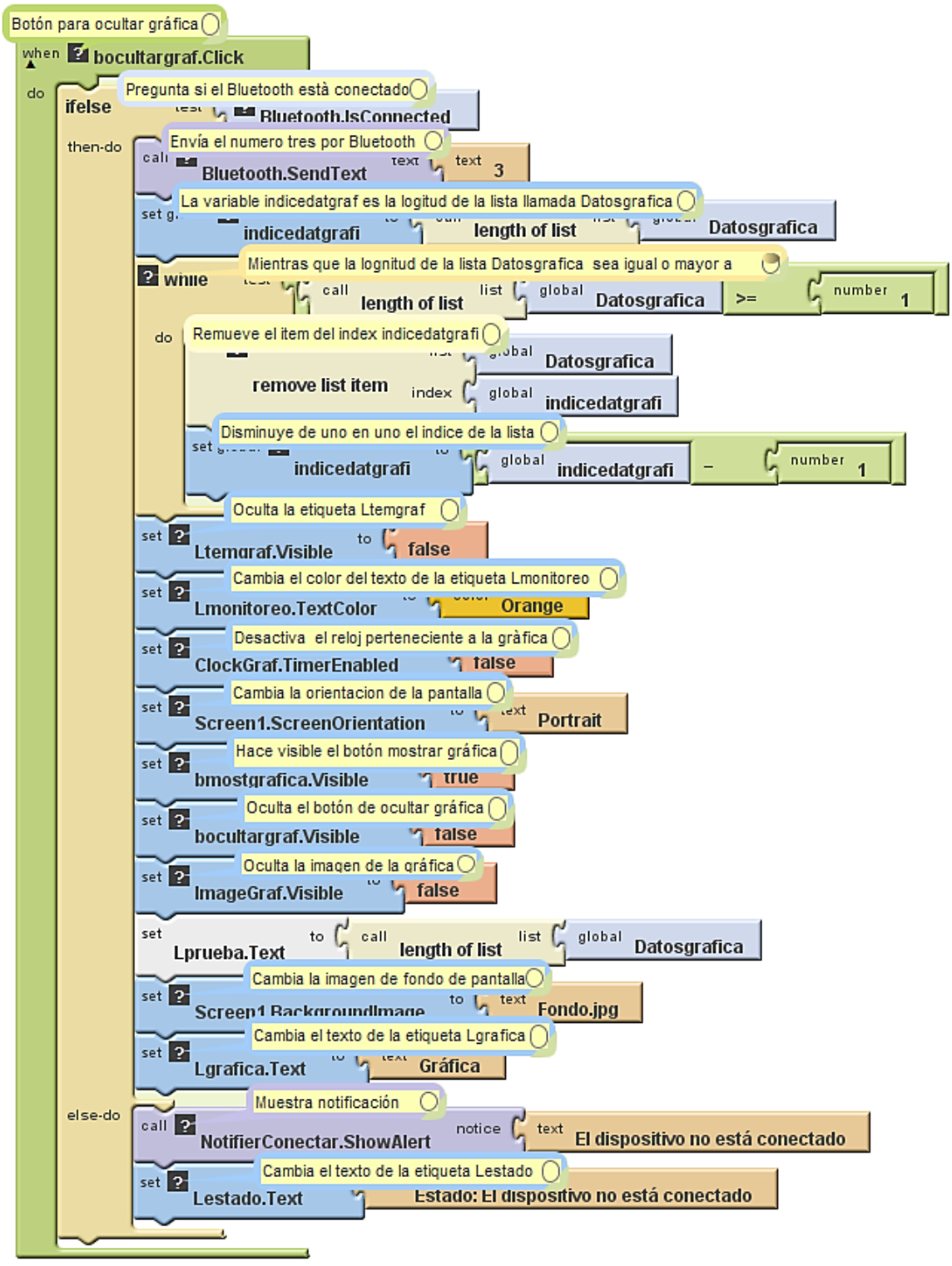

Botón de mostrar monitoreo y ocultar botón de lectura
when bmostmonitoreo.Click
do
Pregunta si el Bluetooth está conectado
ifelse test Bluetooth.IsConnected
then-do
Envía el numerio siete por medio de Bluetooth
call Bluetooth.SendText 7
Oculta el botón de monitoreo
set bmostmonitoreo.Visible false
Hace visible el botón de lectura de datos
set bmostLect.Visible true
Cambia el texto de la etiqueta Llectura
set Llectura.Text Lectura
Cambia el texto de la etiqueta Lestado
set Lestado.Text Estado: El dispositivo está conectado, monitoreando
Muestra notificaciòn en pantalla
else-do
call NotifierConectar.ShowAlert notice text El dispositivo no está conectado
Cambia el texto en la etiqueta Lestado
set Lestado.Text Estado: El dispositivo no está conectado
Despues de la notificacion de calibrar
when Noti_calibrar.AfterChoosing choice name
do
Envía el numero uno por Bluetooth
call Bluetooth.SendText text 1
Muestra notificación: "Se ha calibrado los sensores"
call Noti_calibrar.ShowAlert Se ha calibrado los sensores
Muestra diálogo para almacenar los datos en la microSd
call
¿Desea comenzar a guardar datos en la memoria microSD?
title Almacenar en memoria microSD
NotifierGuardar.ShowChooseDialog
button1Text No
button2Text Si
cancelable false

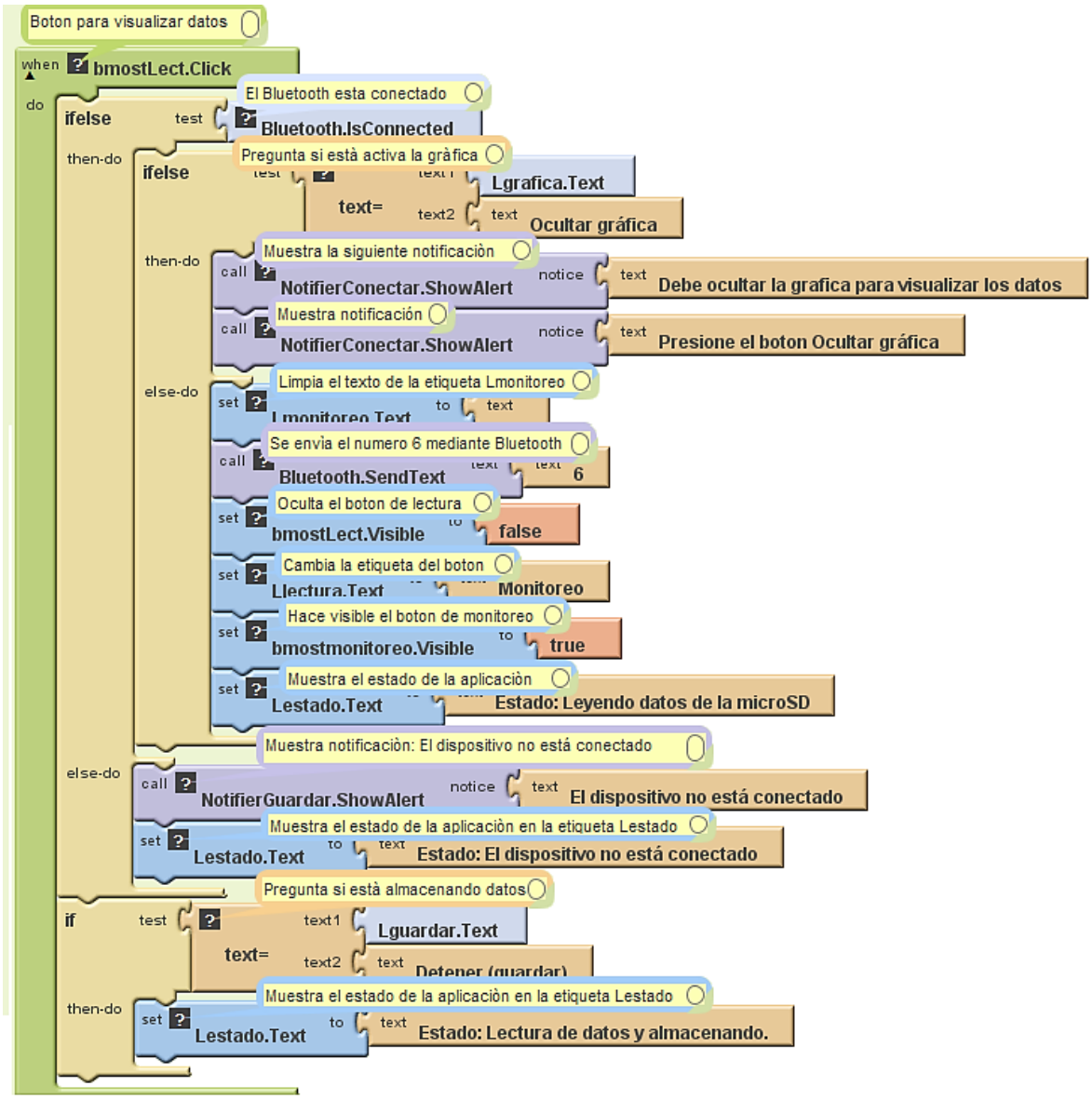

Bloque para despues de haber escrito el nombre del usuario
when NotifierConectar.AfterTextInput response name response
do
ifelse test text= text1 value response text2 text
Pregunta si el usuario no escribio nada en el cuadro de nombre
then-do
Muestra la notificaciòn: No ha escrito nada, escriba su nombre por favor
call NotifierConectar.ShowAlert notice text No ha escrito nada, escriba su nombre por favor
Como el usuario no ha escrito nada, le vuelve a pedir que escriba su nombre
call NotifierConectar.ShowTextDialog message Bienvenido!! Ingrese su nombre por favor title text Nombre del usuario cancelable false
else-do
Convierte el nombre del usuario en mayùsculas
set Nombre.Text to upcase text value response
Le da la bienvenida al usuario
call TextToSpeech1.Speak message text Bienvenido join Nombre.Text
Muestra notificaciòn
call NotifierConectar.ShowAlert notice text Presione el botón conectar para iniciar
set global f1 to true
NOTIFICACION SI / NO COMENZAR A GUARDAR
when NotifierGuardar.AfterChoosing choice name choice
do
ifelse test value choice = text Si
Pregunta si la respuesta es si
then-do
Muestra notificaciòn
call NotifierConectar.ShowAlert notice text Guardando datos en memoria microSD, no extraiga la tarjeta de memoria
Envía el numero cuatro por medio de Bluetooth
call Bluetooth.SendText 4
Oculta el botón de guardar datos
set bguadat.Visible to false
Hace visible el boton detener guardar
set bdetgua.Visible to true
Cambia el texto de la etiqueta Lguardar
set Lguardar.Text to text Detener (guardar)
Muestra el estado de la aplicaciòn
set Lestado.Text to text Estado: El dispositivo está conectado, almacenando datos
else-do
if test value choice = text No
Pregunta si la respuesta es No
then-do
Envía el numero cinco por medio de Bluetooth
call Bluetooth.SendText 5
Hace visible el boton de guardar datos
set bguadat.Visible to true
Oculta el boton de detener guardar
set bdetgua.Visible to false
Cambia el texto de la etiqueta Lguardar
set Lguardar.Text to text Guardar
Muestra el estado de la aplicaciòn
set Lestado.Text to Estado: El dispositivo está conectado, no está almacenando datos

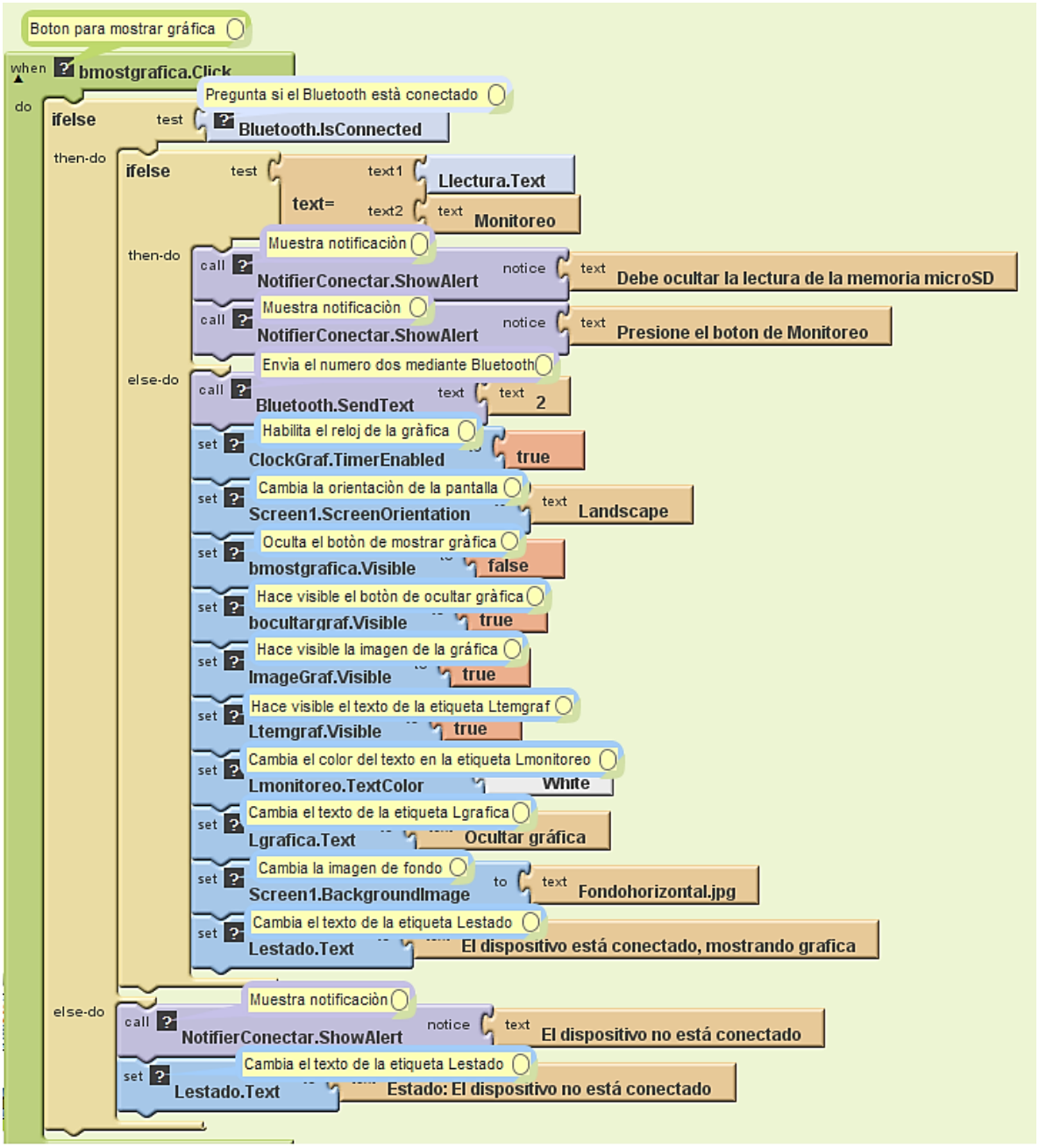

Reloj de la gráfica
when
ClockGraf.Timer
do
set
Lprueba.Text
to
call
length of list
list
global
Datosgrafica
ifelse
test
call
contains
text
Lmonitoreo.Text
piece
text
El texto en la etiqueta Lmonitoreo se pone un "0"
then-do
set
Lmonitoreo.Text
to
text
0
Tiempo de muestreo
else-do
set
ClockGr
Agrega los datos de la etiqueta Lmonitoreo a la lista Datos grafica
call
add items to list
list
global
Datosgrafica
item
Lmonitoreo.Text
item
agrega los tiempo de muestreo a la lista datatime
call
add items to list
list
global
datatime
item
call
make text
text
call
ClockGraf.Hour
instant
call
ClockGraf.Now
text
:
text
call
ClockGraf.Minute
instant
call
ClockGraf.Now
text
item
Pregunta si la longitud de la lista es mayor a cero.
if
test
call
length of list
list
global
Datosgrafica
>
number
0
then-do
set global
chdParam
to
call
make text
text
chd=t:
text
-1|
text
call
select list item
list
global
Datosgrafica
index
number
1
text
set global
tiempo
to
call
make text
text
chl=
text
call
select list item
list
global
datatime
index
number
1
text
El Indice de la gráfica toma el valor de dos
set global
indicedatgrafi
to
number
2
while
test
global
indicedatgrafi
<=
call
length of list
list
global
Datosgrafica
do
En la variable chdParam se almacena la misma variable y los datos almacenados en la lista que van separados por comas
set global
chdParam
to
call
make text
text
global
chdParam
text
,
text
call
select list item
list
global
Datosgrafica
index
global
indicedatgrafi
text
En la variable tiempose almacena la misma variable y los datos almacenados en la lista que van separados por |
set global
tiempo
to
call
make text
text
global
tiempo
text
|
text
call
select list item
list
global
datatime
index
global
indicedatgrafi
text
El Indice de la gráfica aumenta de uno en uno
set global
indicedatgrafi
to
global
indicedatgrafi
+
number
1
Se crea una URL adjuntando la variable chdParam,tiempo parara mostra la imagen de la gráfica
set
ImageGraf.Picture
to
call
make text
text
http://chart.googleapis.com/chart?cht=a,s,000000B4&chxr=1,-5,100&chxt=y,x&chs=350x220&cht=lxy&chco=FF9900&chdl=Temperatur
text
global
chdParam
text
&
text
global
tiempo
text

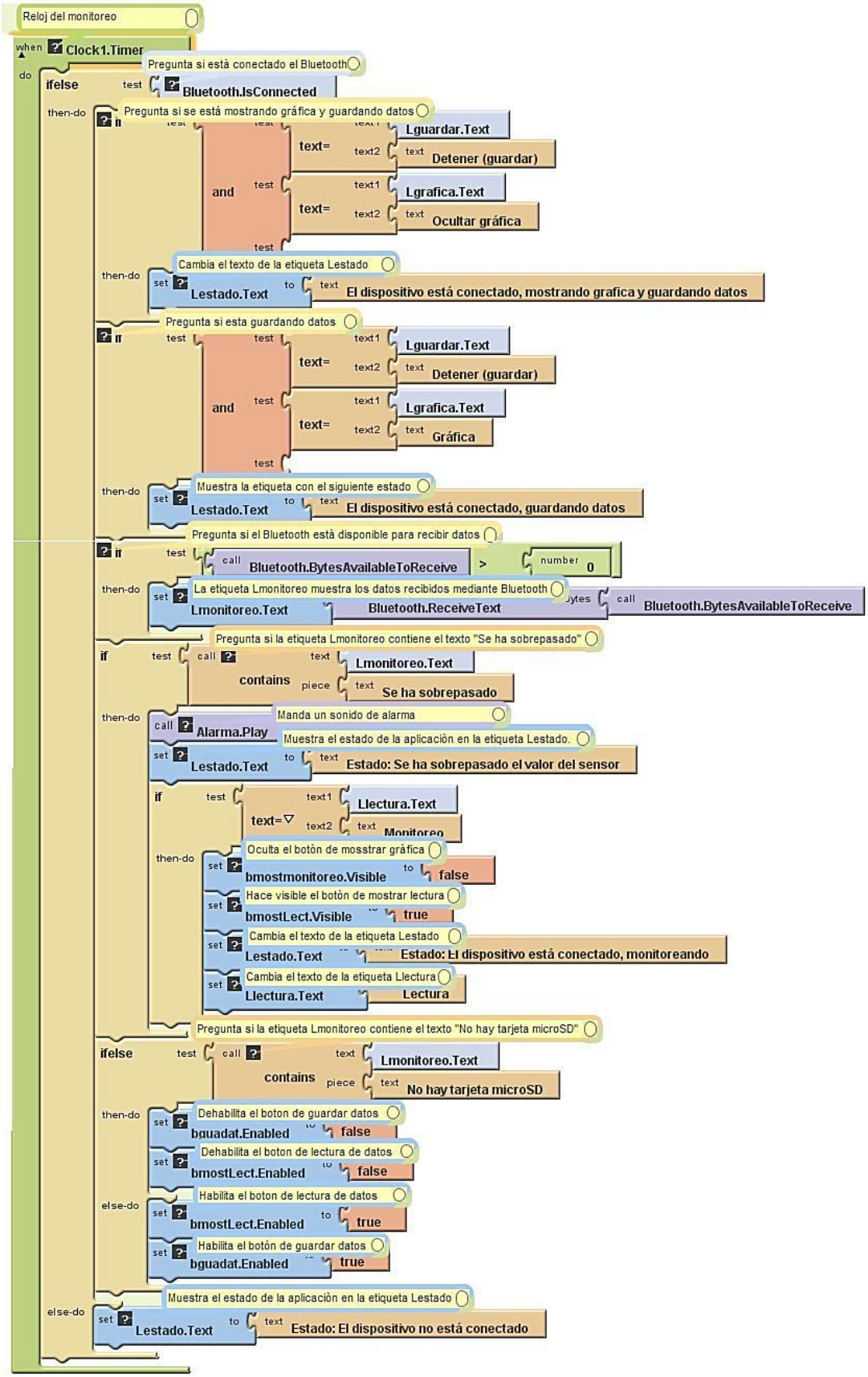

Al iniciar la pantalla principal, debe pedir el nombre del usuario.
when Screen1.Initialize
do
Muestra el mensaje para que el usuario ingrese su nombre
call NotifierConectar.ShowTextDialog
message
text
Bienvenido!! ingrese su nombre por favor
title
text
Nombre del usuario
cancelable
false

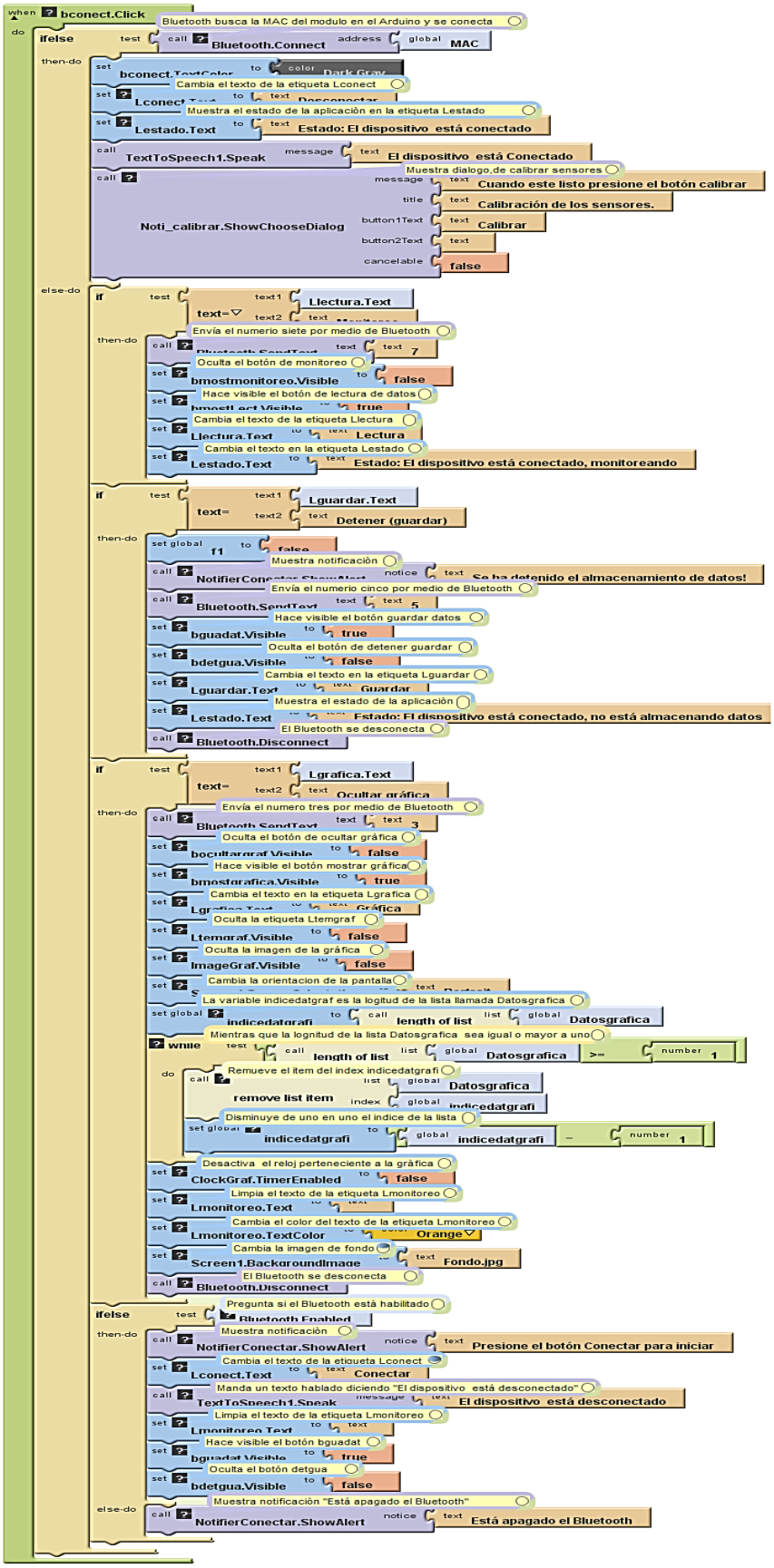

Variable para la direccion MAC del modulo Bluetooth de Arduino
def MAC as text 00:12:08:21:07:16
Variable chdParam que almacena los datos para la URL de la grafica
def chdParam as text
Lista para almacenar los datos de la temperatura
def Datosgrafica as call make a list
item
Variable para el ìndice de la lista Datosgrafica
def indicedatgrafi as number 1


Botón para ocultar gráfica
when bocultargraf.Click
do
ifelse
Pregunta si el Bluetooth está conectado
Bluetooth.IsConnected
then-do
Envía el numero tres por Bluetooth
call Bluetooth.SendText
text 3
La variable indicedatgraf es la logitud de la lista llamada Datosgrafica
indicedatgrafi
length of list
Datosgrafica
Mientras que la lognitud de la lista Datosgrafica sea igual o mayor a
while
call length of list
list global Datosgrafica
>=
number 1
do
Remueve el item del index indicedatgrafi
remove list item
Datosgrafica
index global indicedatgrafi
Disminuye de uno en uno el indice de la lista
indicedatgrafi
global indicedatgrafi
–
number 1
Oculta la etiqueta Ltemgraf
Ltemgraf.Visible
to false
Cambia el color del texto de la etiqueta Lmonitoreo
Lmonitoreo.TextColor
Orange
Desactiva el reloj perteneciente a la gràfica
ClockGraf.TimerEnabled
false
Cambia la orientacion de la pantalla
Screen1.ScreenOrientation
text Portrait
Hace visible el botón mostrar gráfica
bmostgrafica.Visible
true
Oculta el botón de ocultar gráfica
bocultargraf.Visible
false
Oculta la imagen de la gráfica
ImageGraf.Visible
false
Lprueba.Text
to call length of list
list global Datosgrafica
Cambia la imagen de fondo de pantalla
Screen1.BackgroundImage
to text Fondo.jpg
Cambia el texto de la etiqueta Lgrafica
Lgrafica.Text
Gráfica
else-do
Muestra notificación
call NotifierConectar.ShowAlert
notice text El dispositivo no está conectado
Cambia el texto de la etiqueta Lestado
Lestado.Text
Estado: El dispositivo no está conectado


Boton para visualizar datos
when bmostLect.Click
do
ifelse test
El Bluetooth esta conectado
Bluetooth.IsConnected
then-do
ifelse
Pregunta si està activa la gràfica
text=
Lgrafica.Text
text2 text Ocultar gráfica
then-do
Muestra la siguiente notificaciòn
call NotifierConectar.ShowAlert
notice text Debe ocultar la grafica para visualizar los datos
Muestra notificación
call NotifierConectar.ShowAlert
notice text Presione el boton Ocultar gráfica
else-do
Limpia el texto de la etiqueta Lmonitoreo
set Lmonitoreo.Text to text
Se envìa el numero 6 mediante Bluetooth
call Bluetooth.SendText text 6
Oculta el boton de lectura
set bmostLect.Visible to false
Cambia la etiqueta del boton
set Llectura.Text Monitoreo
Hace visible el boton de monitoreo
set bmostmonitoreo.Visible to true
Muestra el estado de la aplicaciòn
set Lestado.Text Estado: Leyendo datos de la microSD
Muestra notificaciòn: El dispositivo no está conectado
else-do
call NotifierGuardar.ShowAlert
notice text El dispositivo no está conectado
Muestra el estado de la aplicaciòn en la etiqueta Lestado
set Lestado.Text to text Estado: El dispositivo no está conectado
Pregunta si està almacenando datos
if test
text=
text1 Lguardar.Text
text2 text Detener (guardar)
then-do
Muestra el estado de la aplicaciòn en la etiqueta Lestado
set Lestado.Text to text Estado: Lectura de datos y almacenando.


Boton para mostrar gráfica
when bmostgrafica.Click
do
ifelse test Pregunta si el Bluetooth està conectado Bluetooth.IsConnected
then-do
ifelse test text= text1 Llectura.Text text2 text Monitoreo
then-do
call NotifierConectar.ShowAlert Muestra notificaciòn notice text Debe ocultar la lectura de la memoria microSD
call NotifierConectar.ShowAlert Muestra notificaciòn notice text Presione el boton de Monitoreo
else-do
call Bluetooth.SendText Envìa el numero dos mediante Bluetooth text text 2
set ClockGraf.TimerEnabled Habilita el reloj de la gràfica to true
set Screen1.ScreenOrientation Cambia la orientaciòn de la pantalla to text Landscape
set bmostgrafica.Visible Oculta el botòn de mostrar gràfica to false
set bocultargraf.Visible Hace visible el botòn de ocultar gràfica to true
set ImageGraf.Visible Hace visible la imagen de la gráfica to true
set Ltemgraf.Visible Hace visible el texto de la etiqueta Ltemgraf to true
set Lmonitoreo.TextColor Cambia el color del texto en la etiqueta Lmonitoreo to White
set Lgrafica.Text Cambia el texto de la etiqueta Lgrafica to Ocultar gráfica
set Screen1.BackgroundImage Cambia la imagen de fondo to text Fondohorizontal.jpg
set Lestado.Text Cambia el texto de la etiqueta Lestado to El dispositivo está conectado, mostrando grafica
else-do
call NotifierConectar.ShowAlert Muestra notificaciòn notice text El dispositivo no está conectado
set Lestado.Text Cambia el texto de la etiqueta Lestado to Estado: El dispositivo no está conectado


Reloj del monitoreo
when Clock1.Timer
do
Pregunta si està conectado el Bluetooth
ifelse test Bluetooth.IsConnected
then-do
Pregunta si se está mostrando gráfica y guardando datos
if test and test text= text1 Lguardar.Text text2 Detener (guardar) test text= text1 Lgrafica.Text text2 Ocultar gráfica
then-do
Cambia el texto de la etiqueta Lestado
set Lestado.Text to text El dispositivo está conectado, mostrando grafica y guardando datos
Pregunta si esta guardando datos
if test and test text= text1 Lguardar.Text text2 Detener (guardar) test text= text1 Lgrafica.Text text2 Gráfica
then-do
Muestra la etiqueta con el siguiente estado
set Lestado.Text to text El dispositivo está conectado, guardando datos
Pregunta si el Bluetooth està disponible para recibir datos
if test call Bluetooth.BytesAvailableToReceive > number 0
then-do
La etiqueta Lmonitoreo muestra los datos recibidos mediante Bluetooth
set Lmonitoreo.Text Bluetooth.ReceiveText bytes call Bluetooth.BytesAvailableToReceive
Pregunta si la etiqueta Lmonitoreo contiene el texto "Se ha sobrepasado"
if test call contains text Lmonitoreo.Text piece text Se ha sobrepasado
then-do
call Alarma.Play
Manda un sonido de alarma
Muestra el estado de la aplicaciòn en la etiqueta Lestado.
set Lestado.Text to text Estado: Se ha sobrepasado el valor del sensor
if test text= text1 Llectura.Text text2 Monitoreo
then-do
Oculta el botòn de mosstrar gràfica
set bmostmonitoreo.Visible to false
Hace visible el botòn de mostrar lectura
set bmostLect.Visible true
Cambia el texto de la etiqueta Lestado
set Lestado.Text Estado: El dispositivo está conectado, monitoreando
Cambia el texto de la etiqueta Llectura
set Llectura.Text Lectura
Pregunta si la etiqueta Lmonitoreo contiene el texto "No hay tarjeta microSD"
ifelse test call contains text Lmonitoreo.Text piece text No hay tarjeta microSD
then-do
Dehabilita el boton de guardar datos
set bguadat.Enabled false
Dehabilita el boton de lectura de datos
set bmostLect.Enabled false
else-do
Habilita el boton de lectura de datos
set bmostLect.Enabled to true
Habilita el botòn de guardar datos
set bguadat.Enabled true
Muestra el estado de la aplicaciòn en la etiqueta Lestado
else-do
set Lestado.Text to text Estado: El dispositivo no está conectado


when bconect.Click
do
ifelse
test call Bluetooth.Connect address global MAC
Bluetooth busca la MAC del modulo en el Arduino y se conecta
then-do
set bconect.TextColor to color Dark Grey
Cambia el texto de la etiqueta Lconect
Muestra el estado de la aplicación en la etiqueta Lestado
set Lestado.Text to Estado: El dispositivo está conectado
call TextToSpeech1.Speak message El dispositivo está Conectado
Muestra dialogo,de calibrar sensores
call Noti_calibrar.ShowChooseDialog
message Cuando este listo presione el botón calibrar
title Calibración de los sensores.
button1Text Calibrar
button2Text
cancelable false
else-do
if test text= text1 Llectura.Text text2
Envía el numerio siete por medio de Bluetooth
then-do
call Bluetooth.SendText text 7
Oculta el botón de monitoreo
set bmostmonitoreo.Visible to false
Hace visible el botón de lectura de datos
Cambia el texto de la etiqueta Llectura
set Llectura.Text to Lectura
Cambia el texto en la etiqueta Lestado
set Lestado.Text to Estado: El dispositivo está conectado, monitoreando
if test text= text1 Lguardar.Text text2 Detener (guardar)
then-do
set global f1 to false
Muestra notificaciòn
call NotifierConectar.ShowAlert notice Se ha detenido el almacenamiento de datos!
Envía el numerio cinco por medio de Bluetooth
call Bluetooth.SendText text 5
Hace visible el botón guardar datos
set bguadat.Visible to true
Oculta el botón de detener guardar
set bdetgua.Visible to false
Cambia el texto en la etiqueta Lguardar
set Lguardar.Text to Guardar
Muestra el estado de la aplicación
set Lestado.Text to Estado: El dispositivo está conectado, no está almacenando datos
El Bluetooth se desconecta
call Bluetooth.Disconnect
if test text= text1 Lgrafica.Text text2 Ocultar gráfica
then-do
Envía el numero tres por medio de Bluetooth
call Bluetooth.SendText text 3
Oculta el botón de ocultar gráfica
set bocultargraf.Visible to false
Hace visible el botón mostrar gráfica
set bmostgrafica.Visible to true
Cambia el texto en la etiqueta Lgrafica
set Lgrafica.Text to Gráfica
Oculta la etiqueta Ltemgraf
set Ltemgraf.Visible to false
Oculta la imagen de la gráfica
set ImageGraf.Visible to false
Cambia la orientacion de la pantalla
La variable indicedatgraf es la logitud de la lista llamada Datosgrafica
set global indicedatgrafi to call length of list list global Datosgrafica
while test call length of list list global Datosgrafica >= number 1
Mientras que la logntud de la lista Datosgrafica sea igual o mayor a uno
do
Remueve el item del index indicedatgrafi
call remove list item list global Datosgrafica index global indicedatgrafi
Disminuye de uno en uno el indice de la lista
set global indicedatgrafi to global indicedatgrafi – number 1
Desactiva el reloj perteneciente a la gráfica
set ClockGraf.TimerEnabled to false
Limpia el texto de la etiqueta Lmonitoreo
set Lmonitoreo.Text to
Cambia el color del texto de la etiqueta Lmonitoreo
set Lmonitoreo.TextColor to Orange
Cambia la imagen de fondo
set Screen1.BackgroundImage to Fondo.jpg
El Bluetooth se desconecta
call Bluetooth.Disconnect
ifelse test Bluetooth Enabled
Pregunta si el Bluetooth está habilitado
then-do
Muestra notificaciòn
call NotifierConectar.ShowAlert notice Presione el botón Conectar para iniciar
Cambia el texto de la etiqueta Lconect
set Lconect.Text to Conectar
Manda un texto hablado diciendo "El dispositivo está desconectado"
call TextToSpeech1.Speak message El dispositivo está desconectado
Limpia el texto de la etiqueta Lmonitoreo
set Lmonitoreo.Text to
Hace visible el botón bguadat
set bguadat.Visible to true
Oculta el botón detgua
set bdetgua.Visible to false
else-do
Muestra notificaciòn "Está apagado el Bluetooth"
call NotifierConectar.ShowAlert notice Está apagado el Bluetooth

A3. CÓDIGO DE PROGRAMACIÓN Código comentado del microcontrolador ATMEGA328P

```
const int chipSelect = 8;                    //Puerto ocho para el shield microSd
#include "DHT.h"                             //Libreria para el sensor DHT11
#include <Wire.h>                            //Libreria DHT11
DHT dht;                                    //Elemento de libreria DHT11
#include "RTClib.h"                          //Libreria para RTC(Real Time Clock)
RTC_DS1307 rtc;                              //Libreria para RTC

#include <SdFat.h>                           //Libreria para guardar microSD
SdFat sd;                                    //Elemento de la libreria microSD
SdFile myFile;                               //Variable para almacenar valores

int s1=A0;                                  //Entrada analogica A0
int s2=A1;                                  //Entrada analogica A1
int s3=A2;                                  //Entrada analogica A2
int s4=A3;                                  //Entrada analogica A3
int val1,val2,val3,val4;                   //Variables para los cuatro sensores de fuerza
int  fsrVoltage,fsrVoltage2,fsrVoltage3,fsrVoltage4;//Cuatro variables para los voltajes de
los sensores FSR
int valsup_1,valinf_1,valsup_2,valinf_2,valsup_3,valinf_3,valsup_4,valinf_4;//Variables
para los limites superior e inferior de cada sensor
unsigned long fsrResistance,fsrResistance2,fsrResistance3,fsrResistance4; // El voltaje
convertido en resistencia
unsigned long
fsrConductance,fsrConductance2,fsrConductance3,fsrConductance4;//Variables para la
conductancia de cada sensor
int fsrForce,fsrForce2,fsrForce3,fsrForce4;//Variables para la fuerza de cada sensor

char incomingByte;          // caracter entrante
boolean f2=false;           // Bandera para almacenar datos
boolean f3 = false;         //Bandera para mostrar monitoreo
boolean f4 = false;         //Bandera para mostrar valores
boolean f6 = false;         //Bandera de no hay microsd
boolean f7 = false;         //Bandera para nombre de usuario
boolean flag = false;        //Bandera para calibracion
boolean f8 = false;         //Bandera valores de temperatura

boolean f10=false;           //Bandera para almacenar solo una vez el nombre de usuario

String content = "";         //Declara variable para concatenar lo leido en el serial
  char character;            //Declara variable que recibe los datos ingresados del serial

void setup() {
 Serial.begin(38400);                        //Inicia puerto serial a 38400
 rtc.begin();                                //Comienzo del RTC
 pinMode(10, OUTPUT);                       //Puerto 10 de salida para shield microSD
```

```
//Iniciando el sensor de temperatura
  #ifdef AVR                          //Parte de libreria del sensor de temperatura
  Wire.begin();                       //Inicia comunicacion con Arduino
  #else
  Wire1.begin();                    // Parte de la libreria del sensor
  #endif
  dht.setup(2);                       // Pin2 para el sensor DHT11
//*************************************
          }

void loop()
{
val1 = analogRead(s1);                                  //Lectura del pin A1
fsrVoltage = map(val1, 0, 1023, 0, 5000);       //Mapeo del valor analogico a 5V
fsrResistance = 5000 - fsrVoltage;              //fsrVoltaje está en millivolts 5V = 5000mV
fsrResistance *= 10000;                  //Se multiplica por la resistencia de 10K
fsrResistance /= fsrVoltage;                 //Se divide entre el voltaje del sensor FSR
fsrConductance = 1000000;                   //La conductancia se va a medir en microhms
fsrConductance /= fsrResistance;               //Se divide entre la resistencia del sensor FSR
if (fsrConductance <= 1000) {                //Si la fuerza en menor a 10N o 1Kg
fsrForce = fsrConductance / 80;              //La conductancia se dividen entre 80 para
obtener la fza aplicada
} else {                                 //Cuando la fuerza sobrepasa 10N o 1Kg
fsrForce = fsrConductance - 1000;            //Se le resta 1000 a la conductancia
fsrForce /= 30;                          //se divide entre 30 para obtener la fuerza aplicada en
Newton
}
delay(30);                               //Retardo de 30ms

val2 = analogRead(s2);                                  //Lectura del pin A2
fsrVoltage2 = map(val2, 0, 1023, 0, 5000);      //Mapeo del valor analogico a 5V
fsrResistance2 = 5000 - fsrVoltage2;                    //fsrVoltage2 está en millivolts 5V =
5000mV
fsrResistance2 *= 10000;                         //Se multiplica por la resistencia de 10K
fsrResistance2 /= fsrVoltage2;                //Se divide entre el voltaje del sensor FSR
fsrConductance2 = 1000000;                   //La conductancia2 se va a medir en microhms
fsrConductance2 /= fsrResistance2;             //Se divide entre la resistencia del sensor2 FSR
if (fsrConductance2 <= 1000) {               //Si la fuerza en menor a 10N o 1Kg
fsrForce2 = fsrConductance2 / 80;             //La conductancia se dividen entre 80 para
obtener la fza aplicada
} else {                                 //Si la fuerza sobrepasa 10N o 1Kg
fsrForce2 = fsrConductance2 - 1000;           //Se le resta 1000 a la conductancia
fsrForce2 /= 30;                          //se divide entre 30 para obtener la fuerza aplicada en
Newton
}
delay(30);                               //Retardo de 30ms
```

```
val3 = analogRead(s3);                          //Lectura del pin A3
fsrVoltage3 = map(val3, 0, 1023, 0, 5000);   //Mapeo del valor analogico a 5V
fsrResistance3 = 5000 - fsrVoltage3;          //fsrVoltage3 esta en millivolts 5V = 5000mV
fsrResistance3 *= 10000;                      //Se multiplica por la resistencia de 10K
fsrResistance3 /= fsrVoltage3;                //Se divide entre el voltaje del sensor FSR
fsrConductance3 = 1000000;                    //La conductancia3 esta en microhms
fsrConductance3 /= fsrResistance3;            //La conductancia3 se va a medir en microhms
if (fsrConductance3 <= 1000) {                //Si la fuerza en menor a 10N o 1Kg
fsrForce3 = fsrConductance3 / 80;            //La conductancia se dividen entre 80 para
obtener la fza aplicada
} else {                                      //Si la fuerza sobrepasa 10N o 1Kg
fsrForce3 = fsrConductance3 - 1000;           //Se le resta 1000 a la conductancia
fsrForce3 /= 30;                 //se divide entre 30 para obtener la fuerza aplicada en Newton
}
 delay(30);                           //Retardo de 30ms

val4 = analogRead(s4);                          //Lectura del pin A4
fsrVoltage4 = map(val4, 0, 1023, 0, 5000);   //Mapeo del valor analogico a 5V
fsrResistance4 = 5000 - fsrVoltage4;          //fsrVoltage4 está en millivolts 5V = 5000mV
fsrResistance4 *= 10000;                      //Se multiplica por la resistencia de 10K
fsrResistance4 /= fsrVoltage4;                //Se divide entre el voltaje del sensor FSR
fsrConductance4 = 1000000;                    //La conductancia4 está en microhms
fsrConductance4 /= fsrResistance4;            //La conductancia4 se va a medir en microhms
if (fsrConductance4 <= 1000) {                //Si la fuerza en menor a 10N o 1Kg
fsrForce4 = fsrConductance4 / 80;            //La conductancia se dividen entre 80 para
obtener la fza aplicada
} else {                                      //Si la fuerza sobrepasa 10N o 1Kg
fsrForce4 = fsrConductance4 - 1000;          //Se le resta 1000 a la conductancia
fsrForce4 /= 30;                 //se divide entre 30 para obtener la fuerza aplicada en Newton
}

/*Lineas para obtener los valores de temperatura y humedad para el sensor DHT11*/

delay(dht.getMinimumSamplingPeriod());    //Retardo entre lecturas de temperatura y
humedad.
int humidity = dht.getHumidity();             //Variable para almacenar los datos de
humedad
int temperature = dht.getTemperature();       //Variable para almacenar los datos de
temperatura

//*********************************************
DateTime now = rtc.now();                     //RTC para la hora actual

//Registrar en buffer2 todos los valores de los sensores
char buffer2[200];                          //Declara un arreglo de 200 caracteres
snprintf(buffer2, 200, "Fecha: %u/%u/%u \n\r  Hora: %u:%u:%u \n\r Humedad R: %u \n\r
Temp[C]: %u \n\r C Posterior[N]: %u \n\r C Medial[N]: %u \n\r C Anterior[N]: %u \n\r C
```

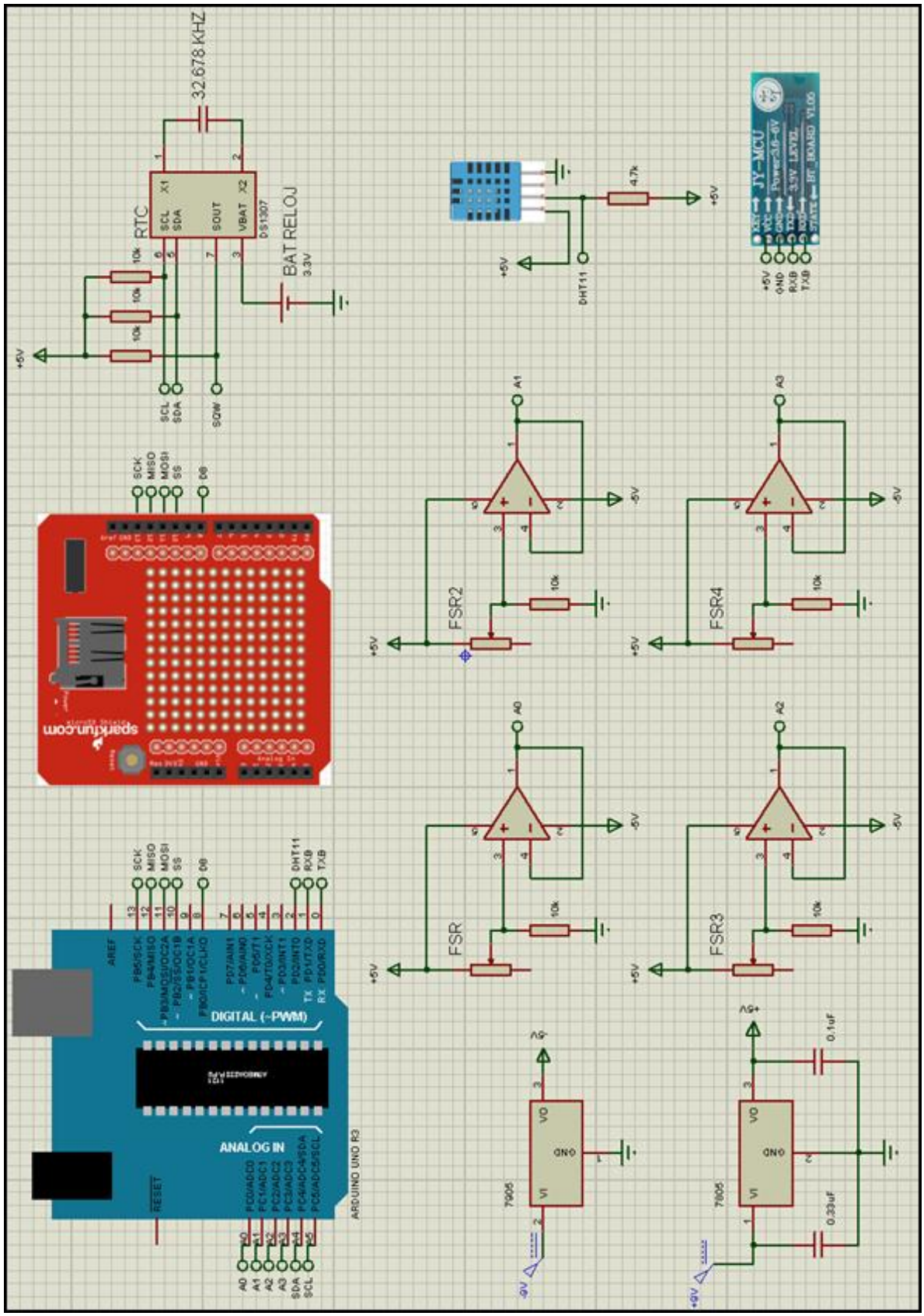
```
Lateral[N]: %u \n\r ",now.day(),now.month(),now.year(), now.hour(),now.minute(),
now.second(), humidity, temperature, fsrForce, fsrForce2, fsrForce3, fsrForce4);

if(f8==true)                    //Pregunta si la bandera para graficar temperatura està activada
 {Serial.print(temperature);          //Manda valores de temperatura
  delay(1000);                        //Retardo de 1 segundo
 }
if (f3==true){                       //Pregunta si la bandera de monitoreo està activada
  Serial.println(buffer2);           //Manda a pantalla el monitoreo de los sensores
    if(f6==true){                    //Si la bandera de no hay tarjeta microSD està activada
      Serial.println("No hay tarjeta microSD");    //Avisa que no hay tarjeta insertada
             }
      delay(100);                    //Retardo de 100ms
           }
if(f7==true){                        //Pregunta si la bandera para capturar nombre del
usuario esta activa
while(Serial.available()) {          //Mientras que el serial este disponible
     character = Serial.read();      //La variable caracter lee el puerto serial
     content.concat(character);      //La variable concaten, concatena el valor leido
  }
  if (content != "") {               //Pregunta si la variable concaten es diferente a ""
  if (!myFile.open("test.txt", O_RDWR | O_CREAT | O_AT_END)) //Pregunta si falla al
abrir el archivo
  { sd.errorHalt("Error al guardar,verifique dispositivo");  //Manda mensaje de error
  }
  myFile.print("Nombre de usuario: ");   //Escribe en el archivo "Nombre del usuario"
  myFile.println(content);       //Escribe en el archivo el nombre que ha ingresado el usuario
  myFile.close();          //Cierra el archivo
  f10=true;                //Activa bandera para almacenar solo una vez el nombre del usuario
  f7=false;                //Desactiva la bandera para capturar nombre del usuario
  }
}
if (!sd.begin(chipSelect, SPI_HALF_SPEED))   //Verifica que este insertada la microSD
  { f6=true;                                 //Activa bandera de no hay microSD
  }
else
 {f6=false;                          //Desactiva bandera de no hay microSD
   if (f2==true)                     //Activa bandera de Almacenando datos
   {
    if (!myFile.open("test.txt", O_RDWR | O_CREAT | O_AT_END)) {//Pregunta si hay
falla al abrir el archivo
    sd.errorHalt("Error al guardar,verifique dispositivo");     //Manda mensaje de error
   }
  myFile.println(buffer2);             //Escribe la variable buffer2 en el archivo
  myFile.close();                            //Cierra el archivo
  }
 }
```

```
int caracter = Serial.read();          //Recibe un caracter para seleccionar algun caso
delay(100);                            //Retardo de 100ms
switch(caracter){                        //Inicio de la sentencia switch para seleccionar casos

case '1':                                          //Caso para calibrar sensores
//Guardando valores de sensores FSR
  valsup_1=fsrForce+1;                     //Se establece rango superior e inferior para
cada uno de los cuatro sensores de fuerza
  valinf_1=fsrForce-1;
  valsup_2=fsrForce2+5;
  valinf_2=fsrForce2-5;
  valsup_3=fsrForce3+5;
  valinf_3=fsrForce3-5;
  valsup_4=fsrForce4+5;
  valinf_4=fsrForce4-5;
//*****************************************
//Serial.println("Se ha calibrado los sensores");
  flag = true;                             //Activa bandera de calibracion
  f3=true;                                 //Activa bandera de monitoreo
  //f9=false;                                //Desactiva bandera de lectura de datos
  f8 = false;                              //Desactiva bandera para enviar valores de temperatura
  f2 = false;                              //Desactiva bandera de almacenar datos
  break;                                  //Termina el caso

case '2':                                 //Caso para mostrar gràfica de temperatura
flag=false;                               //Desactiva bandera de calibracion
f3=false;                                 //Desactiva bandera de monitoreo
delay(100);                               //Retardo de 100ms
f8 = true;                                //Activa bandera para enviar valores de temperatura
break;                                   //Termina el caso

case '3':                                 //Caso para ocultar gràfica
flag=true;                                //Activa bandera de calibracion
f3=true;                                  //Activa Bandera de monitoreo
f8 = false;                               //Desactiva bandera para enviar valores de temperatura
break;                                   //Termina el caso

case '4':                                  //Caso para almacenamiento de datos
if(f10==false)                             //Pregunta si la bandera de almacenar solo una vez el
nombre de usuario está desactivada
{f7=true;}                                 //Activa bandera para recibir el nombre del usuario
else
{f7=false;}                                //Desactiva bandera para recibir el nombre del usuario
delay(500);                                //Retardo de 500ms
f2 = true;                                //Activa bandera de almacenamiento de datos en microSD
```

```
break;                              //Termina el caso

case '5':                                   //Caso para detener el almacenamiento de datos
f2 = false;                          //Desactiva la bandera de almacenar datos
break;                              //Termina el caso

case '6':                           //Caso para mostrar datos almacenados
flag=false;                         //Desactiva bandera de calibración
//f9=true;                            //Activa bandera de lectura de datos
f3=false;
if (sd.exists("test.txt")) {                //Pregunta si existe el archivo
 if (myFile.open("test.txt", O_READ)) {         //Pregunta abre el archivo test.txt
   }
Serial.println("test.txt:");                 //Manda el nombre del archivo
int data;                       //Crea una variable llamada datos donde almacena los datos leidos
while ((data = myFile.read()) >= 0)           //mientras existan datos que leer sean mayor a
cero
Serial.write(data);                  //Manda los datos via serial
myFile.close();                     //Cierra el archivo
  }
  else {
    Serial.println("El archivo test.txt no existe."); //Manda mensaje el archivo no existe
          }
break;                               //Termina el caso

case '7':                           //Caso para desactivar lectura de datos
flag=true;                          //Activa bandera calibración
//f9=false;                           //Desactiva bandera de lectura de datos
f3=true;                            //Activa bandera de monitoreo
break;                               //Termina caso
              }                     //Termina la sentencia Switch case

if (flag == true)                        //Pregunta si la bandera de calibracion està activada
 {
   if(humidity>=60)
  {Serial.println("Humedad relativa de 50%"); //Manda mensaje de precauciòn
    }
   if(temperature>=30)              //Pregunta si la temperatura es mayor o igual a 35 grados
    {Serial.println("Precaucion: temperatura elevada!"); //Manda mensaje de precauciòn
    }
  // Comparando  los valores maximos y minimos permitidos de los sesores
   if (fsrForce==0)                             //Pregunta si la fza del sensor uno es cero
      {
        //Serial.println("No hay fza en S1");
      }
   else
```

```
  {if (fsrForce >= valsup_1 )                    //Pregunta si la fza del sensor 1 es mayor al
valor superior calibrado
       { Serial.println("ha sobrepasado C Posterior"); //muestra mensaje
            }
    if (fsrForce <= valinf_1)                    //Pregunta si la fza del sensor 1 es menor al
valor menor calibrado
    {Serial.println("ha disminuido C Posterior");      //muestra mensaje
    }
    if (fsrForce2 >= valsup_2 )                   //Pregunta si la fza del sensor 2 es mayor al
valor superior calibrado
       { Serial.println("ha sobrepasado C Anterior");  //muestra mensaje
            }
    if (fsrForce2 <= valinf_2)                   //Pregunta si la fza del sensor 1 es menor al
valor menor calibrado
    {Serial.println("ha disminuido C Anterior");      //muestra mensaje
    }
    if (fsrForce3 >= valsup_3 )                   //Pregunta si la fza del sensor 3 es mayor al
valor superior calibrado
       { Serial.println("ha sobrepasado C Medial");  //muestra mensaje
            }
    if (fsrForce3 <= valinf_3)                   //Pregunta si la fza del sensor 1 es menor al
valor menor calibrado
    {Serial.println("ha disminuido C Medial");       //muestra mensaje
    }
    if (fsrForce4 >= valsup_4 )                   //Pregunta si la fza del sensor 4 es mayor al
valor superior calibrado
       { Serial.println("ha sobrepasado C Lateral");  //muestra mensaje
            }
    if (fsrForce4 <= valinf_4)                   //Pregunta si la fza del sensor 1 es menor al
valor menor calibrado
    {Serial.println("ha disminuido C Lateral");      //muestra mensaje
    }
  }

 }
}
```

A4.DIAGRAMA DEL CIRCUITO


32.678 KHZ
RTC
DS1307
BAT RELOJ
3.3V
SCL
SDA
SQW
SOUT
VBAT
X1
X2
10k
4.7k
+5V
DHT11
JY-MCU
FSR
FSR2
FSR3
FSR4
A0
A1
A2
A3
7905
0.1uF
0.33uF
-5V
-9V
SCK
MISO
MOSI
SS
sparkfun.com
DIGITAL (~PWM)
ANALOG IN
ARDUINO UNO R3
RESET
AREF
RX
TX

A5.ESPECIFICACIONES

Módulo Bluetooth HC-06

HC Serial Bluetooth Products

User Instructional Manual

1 Introduction

HC serial Bluetooth products consist of Bluetooth serial interface module and Bluetooth adapter, such as:

(1) Bluetooth serial interface module:

Industrial level: HC-03, HC-04(HC-04-M, HC-04-S)

Civil level: HC-05, HC-06(HC-06-M, HC-06-S)

HC-05-D, HC-06-D (with baseboard, for test and evaluation)

(2) Bluetooth adapter:

HC-M4

HC-M6

This document mainly introduces Bluetooth serial module. Bluetooth serial module is used for converting serial port to Bluetooth. These modules have two modes: master and slaver device. The device named after even number is defined to be master or slaver when out of factory and can't be changed to the other mode. But for the device named after odd number, users can set the work mode (master or slaver) of the device by AT commands.

HC-04 specifically includes:

Master device: HC-04-M, M=master

Slave device: HC-04-S, S=slaver

The default situation of HC-04 is slave mode. If you need master mode, please state it clearly or place an order for HC-O4-M directly.The naming rule of HC-06 is same.

When HC-03 and HC-05 are out of factory, one part of parameters are set for activating the device. The work mode is not set, since user can set the mode of HC-03, HC-05 as they want.

The main function of Bluetooth serial module is replacing the serial port line, such as:

1. There are two MCUs want to communicate with each other. One connects to Bluetooth master device while the other one connects to slave device. Their connection can be built once the pair is made. This Bluetooth connection is equivalently liked to a serial port line connection including RXD, TXD

signals. And they can use the Bluetooth serial module to communicate with each other.

2. When MCU has Bluetooth salve module, it can communicate with Bluetooth adapter of computers and smart phones. Then there is a virtual communicable serial port line between MCU and computer or smart phone.

3. The Bluetooth devices in the market mostly are salve devices, such as Bluetooth printer, Bluetooth GPS. So, we can use master module to make pair and communicate with them.

Bluetooth Serial module's operation doesn't need drive, and can communicate with the other Bluetooth device who has the serial. But communication between two Bluetooth modules requires at least two conditions:

(1) The communication must be between master and slave.

(2) The password must be correct.

However, the two conditions are not sufficient conditions. There are also some other conditions basing on different device model. Detailed information is provided in the following chapters.

In the following chapters, we will repeatedly refer to Linvor's (Formerly known as Guangzhou HC Information Technology Co., Ltd.) material and photos.

2 Selection of the Module

The Bluetooth serial module named even number is compatible with each other; The salve module is also compatible with each other. In other word, the function of HC-04 and HC-06, HC-03 and HC-05 are mutually compatible with each other. HC-04 and HC-06 are former version that user can't reset the work mode (master or slave). And only a few AT commands and functions can be used, like reset the name of Bluetooth (only the slaver), reset the password, reset the baud rate and check the version number. The command set of HC-03 and HC-05 are more flexible than HC-04 and HC-06's. Generally, the Bluetooth of HC-03/HC-05 is recommended for the user.

Here are the main factory parameters of HC-05 and HC-06. Pay attention to the differences:

HC-05	HC-06
Master and slave mode can be switched	Master and slave mode can't be switched
Bluetooth name: HC-05	Bluetooth name: linvor
Password:1234	Password:1234

Master role: have no function to remember the last paired salve device. It can be made paired to any slave device. In other words, just set AT+CMODE=1 when out of factory. If you want HC-05 to remember the last paired slave device address like HC-06, you can set AT+CMODE=0 after paired with the other device. Please refer the command set of HC-05 for the details.	Master role: have paired memory to remember last slave device and only make pair with that device unless KEY (PIN26) is triggered by high level. The default connected PIN26 is low level.
Pairing: The master device can not only make pair with the specified Bluetooth address, like cell-phone, computer adapter, slave device, but also can search and make pair with the slave device automatically. Typical method: On some specific conditions, master device and slave device can make pair with each other automatically. (This is the default method.)	Pairing: Master device search and make pair with the slave device automatically. Typical method: On some specific conditions, master and slave device can make pair with each other automatically.
Multi-device communication: There is only point to point communication for modules, but the adapter can communicate with multi-modules.	Multi-device communication: There is only point to point communication for modules, but the adapter can communicate with multi-modules.
AT Mode 1: After power on, it can enter the AT mode by triggering PIN34 with high level. Then the baud rate for setting AT command is equal to the baud rate in communication, for example: 9600. AT mode 2: First set the PIN34 as high level, or while on powering the module set the PIN34 to be high level, the Baud rate used here is 38400 bps. Notice: All AT commands can be operated only	AT Mode: Before paired, it is at the AT mode. After paired it's at transparent communication.

when the PIN34 is at high level. Only part of the AT commands can be used if PIN34 doesn't keep the high level after entering to the AT mode. Through this kind of designing, set permissions for the module is left to the user's external control circuit, that makes the application of HC-05 is very flexible.	
During the process of communication, the module can enter to AT mode by setting PIN34 to be high level. By releasing PIN34, the module can go back to communication mode in which user can inquire some information dynamically. For example, to inquire the pairing is finished or not.	During the communication mode, the module can't enter to the AT mode.
Default communication baud rate: 9600, 4800-1.3M are settable.	Default communication baud rate: 9600, 1200-1.3M are settable.
KEY: PIN34, for entering to the AT mode.	KEY: PIN26, for master abandons memory.
LED1: PIN31, indicator of Bluetooth mode. Slow flicker (1Hz) represents entering to the AT mode2, while fast flicker(2Hz) represents entering to the AT mode1 or during the communication pairing. Double flicker per second represents pairing is finished, the module is communicable. LED2: PIN32, before pairing is at low level, after the pairing is at high level. The using method of master and slaver's indicator is the same. Notice: The PIN of LED1 and LED2 are connected with LED+.	LED: The flicker frequency of slave device is 102ms. If master device already has the memory of slave device, the flicker frequency during the pairing is 110ms/s. If not, or master has emptied the memory, then the flicker frequency is 750m/s. After pairing, no matter it's a master or slave device, the LED PIN is at high level. Notice: The LED PIN connects to LED+ PIN.
Consumption: During the pairing, the current is	Consumption: During the pairing, the current is

fluctuant in the range of 30-40mA. The mean current is about 25mA. After paring, no matter processing communication or not, the current is 8mA. There is no sleep mode. This parameter is same for all the Bluetooth modules.	fluctuant in the range of 30-40 m. The mean current is about 25mA. After paring, no matter processing communication or not, the current is 8mA. There is no sleep mode. This parameter is same for all the Bluetooth modules.
Reset: PIN11, active if it's input low level. It can be suspended in using.	Reset: PIN11, active if it's input low level. It can be suspended in using.
Level: Civil	Level: Civil

The table above that includes main parameters of two serial modules is a reference for user selection.

HC-03/HC-05 serial product is recommended.

3. Information of Package

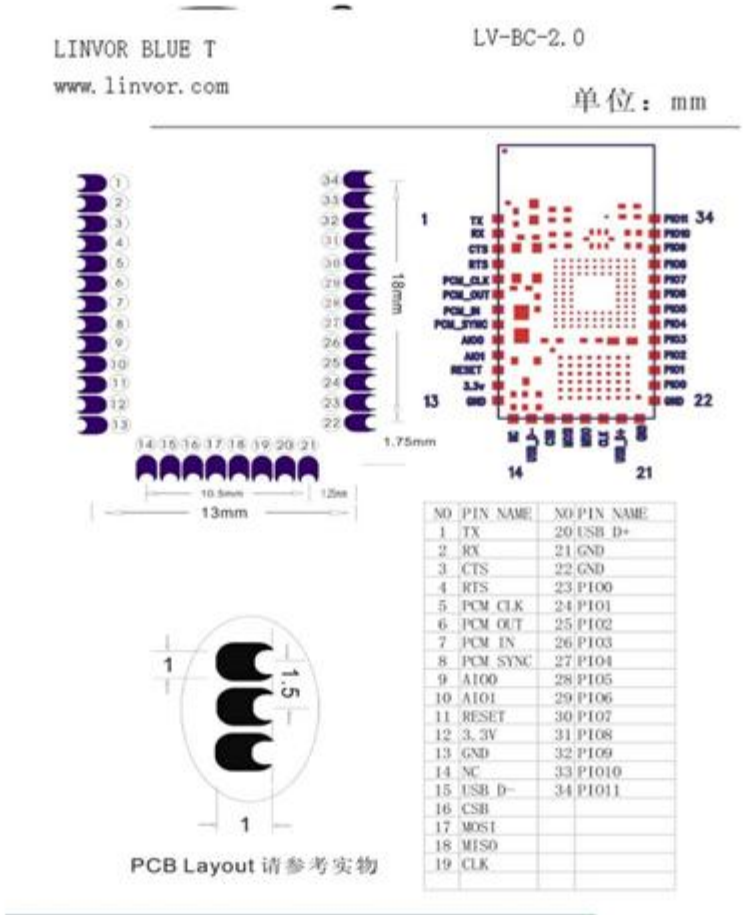
The PIN definitions of HC-03, HC-04, HC-05 and HC-06 are kind of different, but the package size is the same: 28mm * 15mm * 2.35mm.

The following figure 1 is a picture of HC-06 and its main PINs. Figure 2 is a picture of HC-05 and its main PINs. Figure 3 is a comparative picture with one coin. Figure 4 is their package size information. When user designs the circuit, you can visit the website of Guangzhou HC Information Technology Co., Ltd. (www.wavesen.com) to download the package library of protle version.

Figure 1 HC-06

Figure 2 HC-05

Figure 3 Comparative picture with one coin



NO	PIN NAME	NO	PIN NAME
1	TX	20	USB D+
2	RX	21	GND
3	CTS	22	GND
4	RTS	23	PIO0
5	PCM_CLK	24	PIO1
6	PCM_OUT	25	PIO2
7	PCM_IN	26	PIO3
8	PCM_SYNC	27	PIO4
9	AIO0	28	PIO5
10	AIO1	29	PIO6
11	RESET	30	PIO7
12	3.3V	31	PIO8
13	GND	32	PIO9
14	NC	33	PIO10
15	USB D-	34	PIO11
16	CSB		
17	MOSI		
18	MISO		
19	CLK		

Figure 4 Package size information

A6.ESPECIFICACIONES

Sensor FSR 406

INTERLINK ELECTRONICS
Sensor Technologies

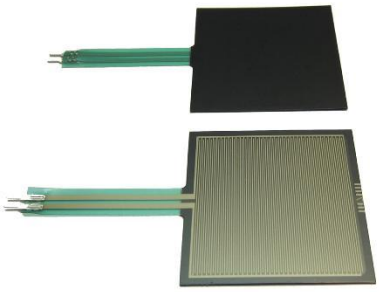
FSR 406 Data Sheet

FSR 400 Series Square Force Sensing Resistor

Description

Interlink Electronics FSR™ 400 series is part of the single zone Force Sensing Resistor™ family. Force Sensing Resistors, or FSRs, are robust polymer thick film (PTF) devices that exhibit a decrease in resistance with increase in force applied to the surface of the sensor. This force sensitivity is optimized for use in human touch control of electronic devices such as automotive electronics, medical systems, and in industrial and robotics applications.

The standard 406 sensor is a square sensor 43.69mm in size. Custom sensors can be manufactured in sizes ranging from 5mm to over 600mm.



Features and Benefits

- Actuation Force as low as 0.1N and sensitivity range to 10N.
- Easily customizable to a wide range of sizes
- Highly Repeatable Force Reading; As low as 2% of initial reading with repeatable actuation system
- Cost effective
- Ultra thin; 0.45mm
- Robust; up to 10M actuations
- Simple and easy to integrate

Industry Segments

- Game controllers
- Musical instruments
- Medical device controls
- Remote controls
- Navigation Electronics
- Industrial HMI
- Automotive Panels
- Consumer Electronics

Figure 1 - Typical Force Curve

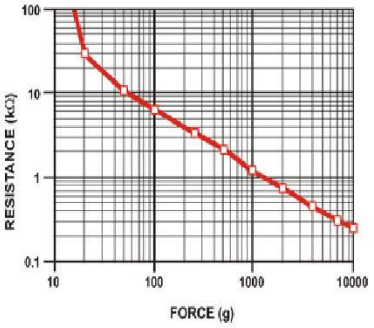
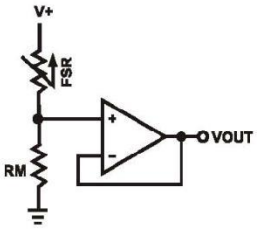


Figure 2 - Typical Schematic



Interlink Electronics - Sensor Technologies

INTERLINK ELECTRONICS
Sensor Technologies

FSR 406
P/N: 30-73258

Applications

Detect & qualify press
Sense whether a touch is accidental or intended by reading force

Use force for UI feedback
Detect more or less user force to make a more intuitive interface

Enhance tool safety
Differentiate a grip from a touch as a safety lock

Find centroid of force
Use multiple sensors to determine centroid of force

Detect presence, position, or motion
Of a person or patient in a bed, chair, or medical device

Detect liquid blockage
Detect tube or pump occlusion or blockage by measuring back pressure

Detect tube positioning

Many other force measurement applications

Device Characteristics

Feature	Condition	Value*	Notes
Actuation Force		0.1 Newtons	
Force Sensitivity Range		0.1 - 10.0[2] Newtons	
Force Repeatability[3]	(Single part)	± 2%	
Force Resolution[3]		continuous	
Force Repeatability[3]	(Part to Part)	±6%	
Non-Actuated Resistance		10M W	
Size		43.69 x 43.69mm	
Thickness Range		0.2 - 1.25 mm	
Stand-Off Resistance		>10M ohms	Unloaded, unbent
Switch Travel	(Typical)	0.05 mm	Depends on design
Hysteresis[3]		+10%	$(R_{F+} - R_{F-})/R_{F+}$.
Device Rise Time		<3 microseconds	measured w/steel ball
Long Term Drift		<5% per $\log_{10}$(time)	35 days test, 1kg load
Temp Operating Range	(Recommended)	-30 - +70 °C	
Number of Actuations	(Life time)	10 Million tested	Without failure

* Specifications are derived from measurements taken at 1000 grams, and are given as one standard deviation / mean, unless otherwise noted.

1. Max Actuation force can be modified in custom sensors.
2. Force Range can be increased in custom sensors. Interlink Electronics have designed and manufactured sensors with operating force larger than 50Kg.
3. Force sensitivity dependent on mechanics, and resolution depends on measurement electronics.

INTERLINK ELECTRONICS
Sensor Technologies

FSR 406
P/N: 30-73258

Contact Us

United States
Corporate Offices
Interlink Electronics, Inc.
546 Flynn Road
Camarillo, CA 93012, USA
Phone: +1-805-484-8855
Fax: +1-805-484-9457
Web: www.interlinkelectronics.com
Sales and support:
fsr@interlinkelectronics.com

Japan
Japan Sales Office
Phone: +81-45-263-6500
Fax: +81-45-263-6501
Web: www.interlinkelec.co.jp

Korea
Korea Sales Office
Phone: +82 10 8776 1972

Application Information

FSRs are two-wire devices with a resistance that depends on applied force.

For specific application needs please contact Interlink Electronics support team. An integration guide is also available.

For a simple force-to-voltage conversion, the FSR device is tied to a measuring resistor in a voltage divider configuration (see Figure 3). The output is described by the equation:

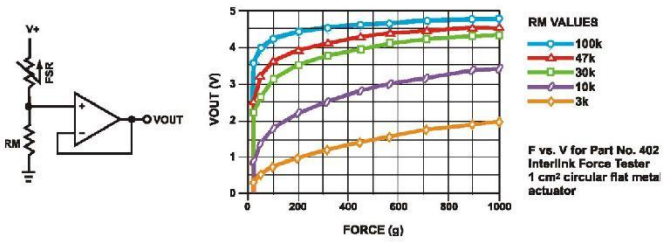
$$V_{OUT} = \frac{R_M V +}{(R_M + R_{FSR})}$$

In the shown configuration, the output voltage increases with increasing force. If R_{FSR} and R_M are swapped, the output swing will decrease with increasing force.

The measuring resistor, R_M, is chosen to maximize the desired force sensitivity range and to limit current. Depending on the impedance requirements of the measuring circuit, the voltage divider could be followed by an op-amp.

A family of force vs. V_{OUT} curves is shown on the graph below for a standard FSR in a voltage divider configuration with various R_M resistors. A (V+) of +5V was used for these examples.

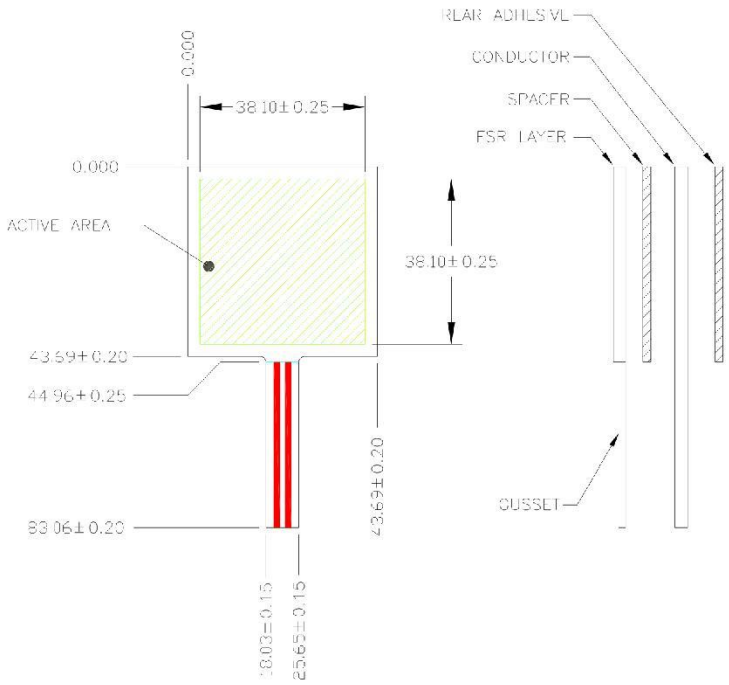
Figure 3



INTERLINK ELECTRONICS
Sensor Technologies

FSR 406
P/N: 30-73258

Mechanical Data



Part No. 406

- Active Area: 38.1mm x 38.1mm
- Nominal thickness: 0.54 mm

P/N: 94-00009 Rev. A

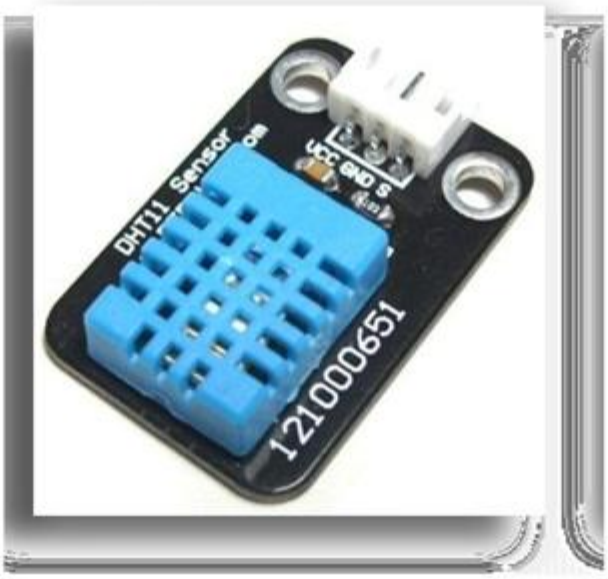
Interlink Electronics - Sensor Technologies

www.interlinkelectronics.com

A7.ESPECIFICACIONES

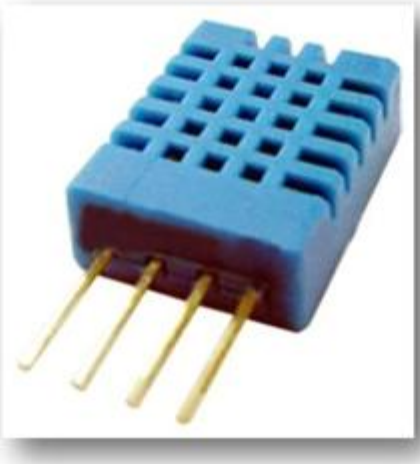
Sensor DHT11

DHT 11 Humidity & Temperature Sensor



1. Introduction

This DFRobot DHT11 Temperature & Humidity Sensor features a temperature & humidity sensor complex with a calibrated digital signal output. By using the exclusive digital-signal-acquisition technique and temperature & humidity sensing technology, it ensures high reliability and excellent long-term stability. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high-performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness.



Each DHT11 element is strictly calibrated in the laboratory that is extremely accurate on humidity calibration. The calibration coefficients are stored as programmes in the OTP memory, which are used by the sensor's internal signal detecting process. The single-wire serial interface makes system integration quick and easy. Its small size, low power consumption and up-to-20 meter signal transmission making it the best choice for various applications, including those most demanding ones. The component is 4-pin single row pin package. It is convenient to connect and special packages can be provided according to users' request.

2. Technical Specifications:

Overview:

Item	Measurement Range	Humidity Accuracy	Temperature Accuracy	Resolution	Package
DHT11	20-90%RH 0-50 ℃	±5%RH	±2℃	1	4 Pin Single Row

Detailed Specifications:

Parameters	Conditions	Minimum	Typical	Maximum
Humidity				
Resolution		1%RH	1%RH	1%RH
			8 Bit	
Repeatability			±1%RH	
Accuracy	25℃		±4%RH	
	0-50℃			±5%RH
Interchangeability	Fully Interchangeable			
Measurement Range	0℃	30%RH		90%RH
	25℃	20%RH		90%RH
	50℃	20%RH		80%RH
Response Time (Seconds)	1/e(63%)25℃, 1m/s Air	6 S	10 S	15 S
Hysteresis			±1%RH	
Long-Term Stability	Typical		±1%RH/year	
Temperature				
Resolution		1℃	1℃	1℃
		8 Bit	8 Bit	8 Bit
Repeatability			±1℃	
Accuracy		±1℃		±2℃
Measurement Range		0℃		50℃
Response Time (Seconds)	1/e(63%)	6 S		30 S

3. Typical Application (Figure 1)

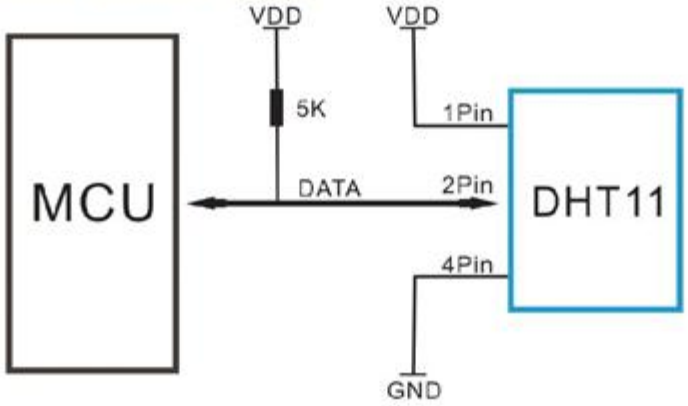


Figure 1 Typical Application

Note: 3Pin – Null; MCU = Micro-computer Unite or single chip Computer

When the connecting cable is shorter than 20 metres, a 5K pull-up resistor is recommended; when the connecting cable is longer than 20 metres, choose a appropriate pull-up resistor as needed.

4. Power and Pin

DHT11's power supply is 3-5.5V DC. When power is supplied to the sensor, do not send any instruction to the sensor in within one second in order to pass the unstable status. One capacitor valued 100nF can be added between VDD and GND for power filtering.

5. Communication Process: Serial Interface (Single-Wire Two-Way)

Single-bus data format is used for communication and synchronization between MCU and DHT11 sensor. One communication process is about 4ms.

Data consists of decimal and integral parts. A complete data transmission is **40bit**, and the sensor sends **higher data bit** first.
Data format: 8bit integral RH data + 8bit decimal RH data + 8bit integral T data + 8bit decimal T data + 8bit check sum. If the data transmission is right, the check-sum should be the last 8bit of "8bit integral RH data + 8bit decimal RH data + 8bit integral T data + 8bit decimal T data".

5.1 Overall Communication Process (Figure 2, below)

When MCU sends a start signal, DHT11 changes from the low-power-consumption mode to the running-mode, waiting for MCU completing the start signal. Once it is completed, DHT11 sends a response signal of 40-bit data that include the relative humidity and temperature information to MCU. Users can choose to collect (read) some data. Without the start signal from MCU, DHT11 will not give the response signal to MCU. Once data is collected, DHT11 will change to the low-power-consumption mode until it receives a start signal from MCU again.

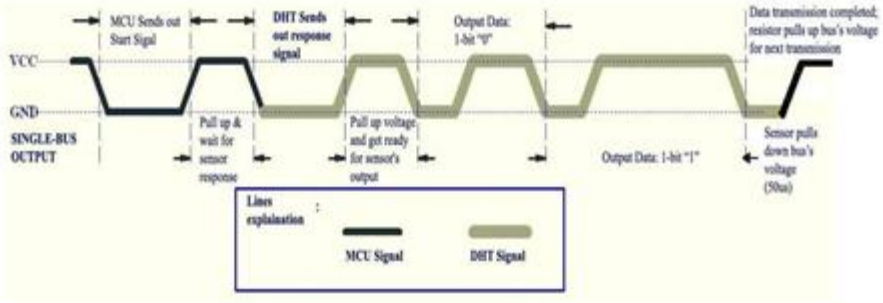


Figure 2 Overall Communication Process

5.2 MCU Sends out Start Signal to DHT (Figure 3, below)

Data Single-bus free status is at high voltage level. When the communication between MCU and DHT11 begins, the programme of MCU will set Data Single-bus voltage level from high to low and this process must take at least 18ms to ensure DHT's detection of MCU's signal, then MCU will pull up voltage and wait 20-40us for DHT's response.

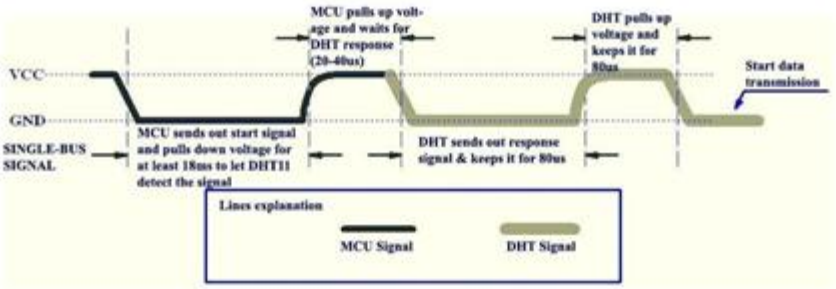


Figure 3 MCU Sends out Start Signal & DHT Responses

A8.ESPECIFICACIONES

Real time clock DS1307

maxim integrated™

DS1307
64 x 8, Serial, I²C Real-Time Clock

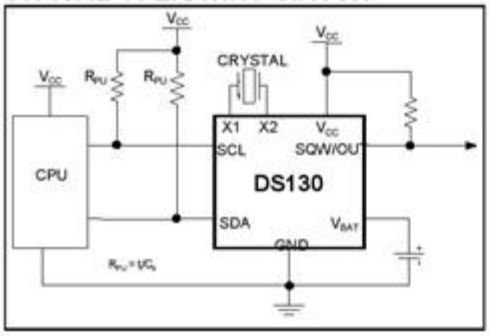
GENERAL DESCRIPTION

The DS1307 serial real-time clock (RTC) is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially through an I²C, bidirectional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power-sense circuit that detects power failures and automatically switches to the backup supply. Timekeeping operation continues while the part operates from the backup supply.

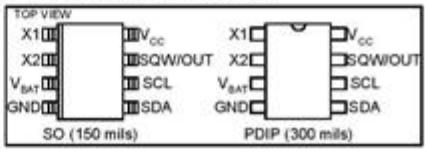
FEATURES

- Real-Time Clock (RTC) Counts Seconds, Minutes, Hours, Date of the Month, Month, Day of the week, and Year with Leap-Year Compensation Valid Up to 2100
- 56-Byte, Battery-Backed, General-Purpose RAM with Unlimited Writes
- I²C Serial Interface
- Programmable Square-Wave Output Signal
- Automatic Power-Fail Detect and Switch Circuitry
- Consumes Less than 500nA in Battery-Backup Mode with Oscillator Running
- Optional Industrial Temperature Range: -40°C to +85°C
- Available in 8-Pin Plastic DIP or SO
- Underwriters Laboratories (UL) Recognized

TYPICAL OPERATING CIRCUIT



PIN CONFIGURATIONS



ORDERING INFORMATION

PART	TEMP RANGE	VOLTAGE (V)	PIN-PACKAGE	TOP MARK*
DS1307+	0°C to +70°C	5.0	8 PDIP (300 mils)	DS1307
DS1307N+	-40°C to +85°C	5.0	8 PDIP (300 mils)	DS1307N
DS1307Z+	0°C to +70°C	5.0	8 SO (150 mils)	DS1307
DS1307ZN+	-40°C to +85°C	5.0	8 SO (150 mils)	DS1307N
DS1307Z+T&R	0°C to +70°C	5.0	8 SO (150 mils) Tape and Reel	DS1307
DS1307ZN+T&R	-40°C to +85°C	5.0	8 SO (150 mils) Tape and Reel	DS1307N

+Denotes a lead-free/RoHS-compliant package.
**A "+" anywhere on the top mark indicates a lead-free package. An "N" anywhere on the top mark indicates an industrial temperature range device.*

For pricing, delivery, and ordering information, please contact Maxim Direct at 1-888-629-4642, or visit Maxim's website at www.maximintegrated.com.

REV: 100208

ABSOLUTE MAXIMUM RATINGS

Voltage Range on Any Pin Relative to Ground ..... -0.5V to +7.0V
Operating Temperature Range (Noncondensing)
Commercial ..... 0°C to +70°C
Industrial ..... -40°C to +85°C
Storage Temperature Range ..... -55°C to +125°C
Soldering Temperature (DIP, leads) ..... +260°C for 10 seconds
Soldering Temperature (surface mount) ..... Refer to the JPC/JEDEC J-STD-020 Specification.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to the absolute maximum rating conditions for extended periods may affect device reliability.

RECOMMENDED DC OPERATING CONDITIONS

(T_A = 0°C to +70°C, T_A = -40°C to +85°C.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Supply Voltage	V_{CC}		4.5	5.0	5.5	V
Logic 1 Input	V_{IH}		2.2		V_{CC} + 0.3	V
Logic 0 Input	V_{IL}		-0.3		+0.8	V
V_{BAT} Battery Voltage	V_{BAT}		2.0	3	3.5	V

DC ELECTRICAL CHARACTERISTICS

(V_{CC} = 4.5V to 5.5V; T_A = 0°C to +70°C, T_A = -40°C to +85°C.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Input Leakage (SCL)	I_{LI}		-1		1	µA
I/O Leakage (SDA, SQW/OUT)	I_{LO}		-1		1	µA
Logic 0 Output (I_{OL} = 5mA)	V_{OL}				0.4	V
Active Supply Current (f_{SCL} = 100kHz)	I_{CCA}				1.5	mA
Standby Current	I_{CCS}	(Note 3)			200	µA
V_{BAT} Leakage Current	I_{BATLKG}			5	50	nA
Power-Fail Voltage (V_{BAT} = 3.0V)	V_{PF}		1.216 x V_{BAT}	1.25 x V_{BAT}	1.284 x V_{BAT}	V

DC ELECTRICAL CHARACTERISTICS

(V_{CC} = 0V, V_{BAT} = 3.0V; T_A = 0°C to +70°C, T_A = -40°C to +85°C.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
V_{BAT} Current (OSC ON); SQW/OUT OFF	I_{BAT1}			300	500	nA
V_{BAT} Current (OSC ON); SQW/OUT ON (32kHz)	I_{BAT2}			480	800	nA
V_{BAT} Data-Retention Current (Oscillator Off)	I_{BATDR}			10	100	nA

WARNING: Negative undershoots below -0.3V while the part is in battery-backed mode may cause loss of data.

AC ELECTRICAL CHARACTERISTICS

(V_{CC} = 4.5V to 5.5V; T_A = 0°C to +70°C, T_A = -40°C to +85°C.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
SCL Clock Frequency	f_{SCL}		0		100	kHz
Bus Free Time Between a STOP and START Condition	t_{BUF}		4.7			µs
Hold Time (Repeated) START Condition	$t_{HD:STA}$	(Note 4)	4.0			µs
LOW Period of SCL Clock	t_{LOW}		4.7			µs
HIGH Period of SCL Clock	t_{HIGH}		4.0			µs
Setup Time for a Repeated START Condition	$t_{SU:STA}$		4.7			µs
Data Hold Time	$t_{HD:DAT}$		0			µs
Data Setup Time	$t_{SU:DAT}$	(Notes 5, 6)	250			ns
Rise Time of Both SDA and SCL Signals	t_R				1000	ns
Fall Time of Both SDA and SCL Signals	t_F				300	ns
Setup Time for STOP Condition	$t_{SU:STO}$		4.7			µs

CAPACITANCE

(T_A = +25°C)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Pin Capacitance (SDA, SCL)	$C_{I/O}$				10	pF
Capacitance Load for Each Bus Line	C_B	(Note 7)			400	pF

Note 1: All voltages are referenced to ground.

Note 2: Limits at -40°C are guaranteed by design and are not production tested.

Note 3: I_{CCS} specified with V_{CC} = 5.0V and SDA, SCL = 5.0V.

Note 4: After this period, the first clock pulse is generated.

Note 5: A device must internally provide a hold time of at least 300ns for the SDA signal (referred to the $V_{IH(MIN)}$ of the SCL signal) to bridge the undefined region of the falling edge of SCL.

Note 6: The maximum $t_{HD:DAT}$ only has to be met if the device does not stretch the LOW period (t_{LOW}) of the SCL signal.

Note 7: C_B—total capacitance of one bus line in pF.

TIMING DIAGRAM

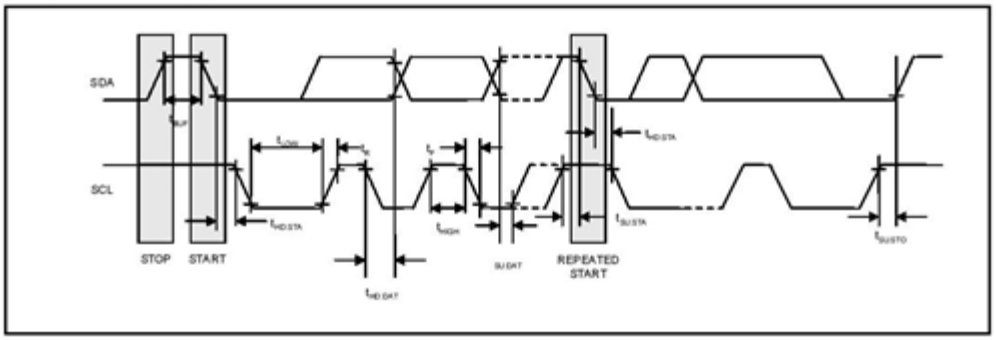
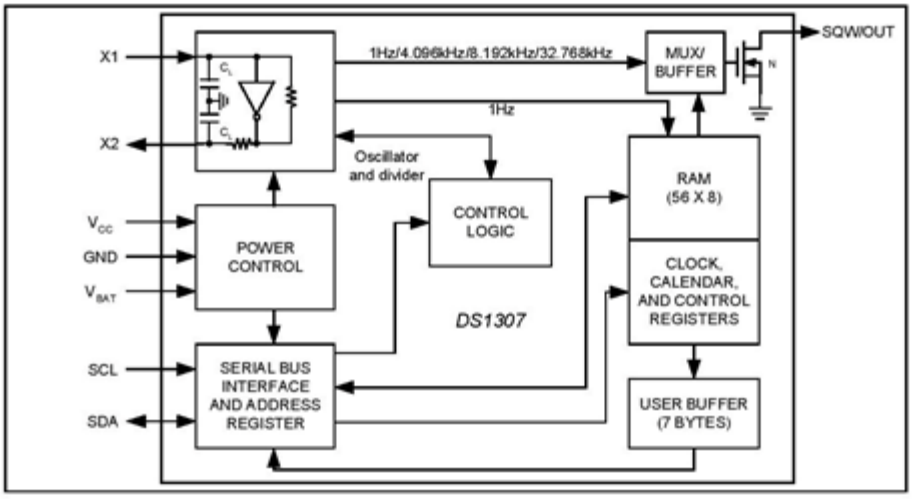


Figure 1. Block Diagram



PIN DESCRIPTION

PIN	NAME	FUNCTION
1	X1	Connections for Standard 32.768kHz Quartz Crystal. The internal oscillator circuitry is designed for operation with a crystal having a specified load capacitance (C_L) of 12.5pF. X1 is the input to the oscillator and can optionally be connected to an external 32.768kHz oscillator. The output of the internal oscillator, X2, is floated if an external oscillator is connected to X1. **Note:** For more information on crystal selection and crystal layout considerations, refer to *Application Note 58: Crystal Considerations with Dallas Real-Time Clocks.*
2	X2	
3	V_{BAT}	Backup Supply Input for Any Standard 3V Lithium Cell or Other Energy Source. Battery voltage must be held between the minimum and maximum limits for proper operation. Diodes in series between the battery and the V_{BAT} pin may prevent proper operation. If a backup supply is not required, V_{BAT} must be grounded. The nominal power-fail trip point (V_{PF}) voltage at which access to the RTC and user RAM is denied is set by the internal circuitry as 1.25 x V_{BAT} nominal. A lithium battery with 48mAh or greater will back up the DS1307 for more than 10 years in the absence of power at +25°C. UL recognized to ensure against reverse charging current when used with a lithium battery. Go to: www.maxim-ic.com/qa/info/ul/.
4	GND	Ground
5	SDA	Serial Data Input/Output. SDA is the data input/output for the I²C serial interface. The SDA pin is open drain and requires an external pullup resistor. The pullup voltage can be up to 5.5V regardless of the voltage on V_{CC}.
6	SCL	Serial Clock Input. SCL is the clock input for the I²C interface and is used to synchronize data movement on the serial interface. The pullup voltage can be up to 5.5V regardless of the voltage on V_{CC}.
7	SQW/OUT	Square Wave/Output Driver. When enabled, the SQWE bit set to 1, the SQW/OUT pin outputs one of four square-wave frequencies (1Hz, 4kHz, 8kHz, 32kHz). The SQW/OUT pin is open drain and requires an external pullup resistor. SQW/OUT operates with either V_{CC} or V_{BAT} applied. The pullup voltage can be up to 5.5V regardless of the voltage on V_{CC}. If not used, this pin can be left floating.
8	V_{CC}	Primary Power Supply. When voltage is applied within normal limits, the device is fully accessible and data can be written and read. When a backup supply is connected to the device and V_{CC} is below V_{TP}, read and writes are inhibited. However, the timekeeping function continues unaffected by the lower input voltage.

DETAILED DESCRIPTION

The DS1307 is a low-power clock/calendar with 56 bytes of battery-backed SRAM. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The date at the end of the month is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The DS1307 operates as a slave device on the I²C bus. Access is obtained by implementing a START condition and providing a device identification code followed by a register address. Subsequent registers can be accessed sequentially until a STOP condition is executed. When V_{CC} falls below 1.25 x V_{BAT}, the device terminates an access in progress and resets the device address counter. Inputs to the device will not be recognized at this time to prevent erroneous data from being written to the device from an out-of-tolerance system. When V_{CC} falls below V_{BAT}, the device switches into a low-current battery-backup mode. Upon power-up, the device switches from battery to V_{CC} when V_{CC} is greater than V_{BAT} +0.2V and recognizes inputs when V_{CC} is greater than 1.25 x V_{BAT}. The block diagram in Figure 1 shows the main elements of the serial RTC.

A9.ESPECIFICACIONES

Microcontrolador ATMEGA328P

Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32K Bytes of In-System Self-Programmable Flash progam memory (ATmega48PA/88PA/168PA/328P)
  - 256/512/512/1K Bytes EEPROM (ATmega48PA/88PA/168PA/328P)
  - 512/1K/1K/2K Bytes Internal SRAM (ATmega48PA/88PA/168PA/328P)
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
    - Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
    - Temperature Measurement
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Byte-oriented 2-wire Serial Interface (Philips I^2C compatible)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 23 Programmable I/O Lines
  - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
  - 1.8 - 5.5V for ATmega48PA/88PA/168PA/328P
- Temperature Range:
  - -40°C to 85°C
- Speed Grade:
  - 0 - 20 MHz @ 1.8 - 5.5V
- Low Power Consumption at 1 MHz, 1.8V, 25°C for ATmega48PA/88PA/168PA/328P:
  - Active Mode: 0.2 mA
  - Power-down Mode: 0.1 µA
  - Power-save Mode: 0.75 µA (Including 32 kHz RTC)

8-bit AVR® Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash

ATmega48PA
ATmega88PA
ATmega168PA
ATmega328P

Rev. 8161D–AVR–10/09

1.1 Pin Descriptions

1.1.1 VCC

Digital supply voltage.

1.1.2 GND

Ground.

1.1.3 Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7..6 is used as TOSC2..1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

The various special features of Port B are elaborated in "Alternate Functions of Port B" on page 82 and "System Clock and Clock Options" on page 26.

1.1.4 Port C (PC5:0)

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5..0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

1.1.5 PC6/$\overline{\text{RESET}}$

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. The minimum pulse length is given in Table 28-3 on page 318. Shorter pulses are not guaranteed to generate a Reset.

The various special features of Port C are elaborated in "Alternate Functions of Port C" on page 85.

1.1.6 Port D (PD7:0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

The various special features of Port D are elaborated in "Alternate Functions of Port D" on page 88.

1.1.7 AV_{CC}

AV_{CC} is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter. Note that PC6..4 use digital supply voltage, V_{CC}.

1.1.8 AREF

AREF is the analog reference pin for the A/D Converter.

1.1.9 ADC7:6 (TQFP and QFN/MLF Package Only)

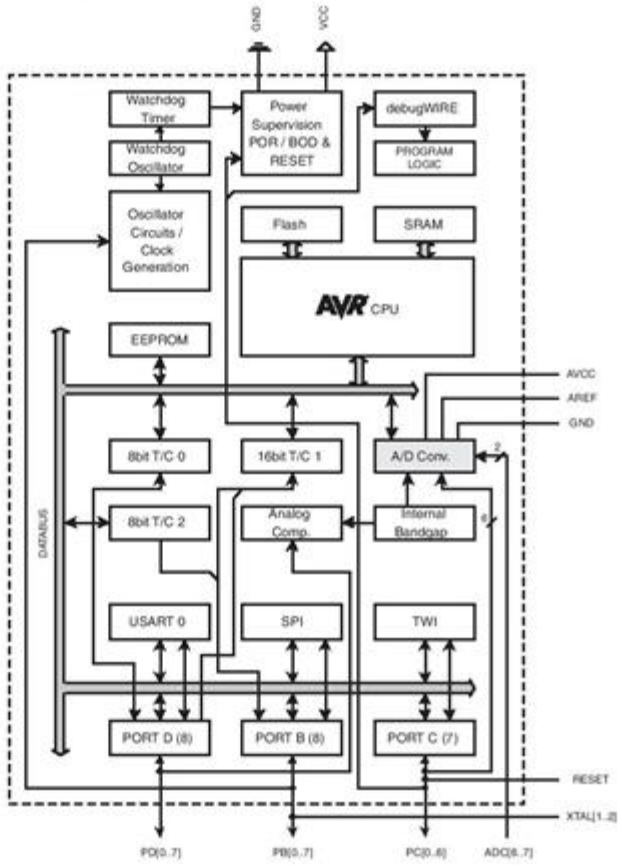
In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

2. Overview

The ATmega48PA/88PA/168PA/328P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega48PA/88PA/168PA/328P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting

Bibliografía

1. **Hanna L.García Guerra.** Tesis de Mastría. *Diseño de un socket autoajustable para prótesis de miembro inferior.* Cuidad de México : s.n., 2009.
2. **José M. Pérez Santana.***Manual de fisioterapia.* s.l. : Mad, S.L., Noviembre 2004.
3. **MedicineNet.** MedicineNet.com. [En línea] http://www.medterms.com/script/main/art.asp?articlekey=12537.
4. **Ron Seymour.***Prosthetics and Orthotics: Lower Limb and Spinal.* USA : LIPPICOTT, 2002.
5. **J.S Cuccurullo.***Physical medicine and rehabilitation board review.* 2002.
6. **Donna Falvo.***Medical and psychosocial aspects of chronic illness and disability.* 2005.
7. *Identifying and Managing Skin Issues With Lower-Limb Prosthetic Use.* **Jason Highsmith.** Num 1, Enero/Febrero del 2011, Vol. 21.
8. **Jesús Alós Villacrosa.***Amputaciones del miembro inferior en cirugía vascular.* Barcelon : Glosa, 2008.
9. **Secretaría de Salud.** [En línea] www.salud.gob.mx.
10. **Guadalupe Camacho.** ¡Cuidado! Diabetes, primera causa de amputación. *Excelsior.* 26/07/2012.
11. **Secretaría de Salud.** Secretaría de Salud. Direccion de información en Salud. *Subsistema Automatizado de Egresos Hospitalarios.* 2004-2006.
12. *Encuesta Nacional de Salud* . **IMSS, Revista Medica.** México : s.n., 2000.
13. **Personal de la NLLIC.***Estadísticas de amputaciones según la causa, pérdida de extremidades en los Estados Unidos.* Estados Unidos : Centro de Información de la Asociación Nacional de Miembros Perdidos, 2006.
14. **Consejo de salubridad general.** Resumen de evidencia y recomendaciones. Guia clinica. [aut. libro] Gobierno del Distrito Federal. México, Distrito Federal : s.n.
15. **Mashable.** [En línea] [Citado el: 20 de Octubre de 2013.] http://mashable.com/.
16. **Comscore Inc.** Analytics for a Digital World. [En línea] www.comscore.com/.
17. **Unocero.** [En línea] 31 de Mayo de 2013. http://www.unocero.com/2013/05/31/ios-la-plataforma-movil-mas-usada-en-mexico/.
18. **Wahid Haq.** blog at WordPress.com. [En línea] http://amarkhatun.wordpress.com/2013/04/27/prosthetic-hand-controlled-with-.
19. **Orthocare innovations.** [En línea] http://www.orthocareinnovations.com/orthocare.micro/index.html.
20. *Prosthetic Primer: Skin Care Determines Prosthetic Comfort.* **Levy, S. William.** Num 1, s.l. : BioMechanics, Enero/Febrero 2000, Vol. 10.
21. **Oriol Sallent Roig.***Principio de comunicaciones móviles.* Barcelona : s.n., 2003.
22. **Interlink Electronics.***FSR Integration Guide.*
23. **Kinetronica.** [En línea] Octubre de 2013. http://www.kinetroni.com/.
24. **Ottobocks.** [En línea] 2014. [Citado el: 15 de Octubre de 2012.] http://professionals.ottobockus.com.
25. **Tecnologia123.** [En línea] http://www.tecnologia123.com/tag/symbian-anna-os/.
26. **Apple.** itunes. [En línea] 11 de Noviembre de 2013. https://itunes.apple.com/mx/app/biosim/id622776961?mt=8.
27. **The Johns Hopkins Hospital.** [En línea] http://www.hopkinsmedicine.org/healthlibrary/test_procedures/cardiovascular/am.

28. **Xataka Móvil.** [En línea] http://www.xatakamovil.com/.
29. **Anandtech.** [En línea] http://www.anandtech.com/show/6302/apple-ios-6-review-maps-investigated-and-more.
30. **Susan Hargreaves.** Mobile Phone News. [En línea] http://www.mofonu.com/2013/05/20/.
31. **One digital.** Apps User. [En línea] http://appsuser.net/.
32. **Pere J. Riu Costa.***Sistemas de instrumentación.* s.l. : UPC, 1995.
33. **W. Bolton.***Mediciones y pruebas eléctricas y electrónicas.* s.l. : Marcombo, 1995.
34. **José Ma. Drake Moyano.***Instrumentación electrónica de comunicaciones.* Santander : Deptp de electrónica y computadores, 2005.
35. **Eduardo SantaMaria.***Electronica digital y microprocesadores.*
36. **Jacob Fraden.***Handbook of modern sensors.* s.l. : Springer.
37. **Interlink Electronics.***FSR 406 Data Sheet.*
38. **Texas Instruments.***LM348N Data Sheet.*
39. *DHT11 DataSheet.*
40. **Atmel.***ATmega328P Datasheet.*
41. **Massachusetts Institute of Technology.** MIT App Inventor. [En línea] http://appinventor.mit.edu/explore/. 41
42. **Arduino.** Arduino.cc. [En línea] http://arduino.cc/en/Main/arduinoBoardUno.
43. **Ralph Roberts.***Google app inventor beginners guide.* s.l. : Packt publishing.
44. **Jason Tyler.***App Inventor for Android.* USA : Jhon Wiley & Sons, 2011.
45. **Jorg H. Kloss.***Android apps with app inventor.* USA : Addison-Wesley.