Pulsera para Triage y Monitorización del Estado de Víctimas

GRADO EN INGENIERÍA ELECTRÓNICA, ROBÓTICA Y MECATRÓNICA

TRABAJO FIN DE GRADO

Autor:

Francisco Jesús Ruiz Ruiz

Tutores:

Jesús Manuel Gómez de Gabriel

Juan Manuel Gandarias Palacios

UNIVERSIDAD DE MÁLAGA

ESCUELA DE INGENIERÍAS INDUSTRIALES

Departamento de Ingeniería de Sistemas y Automática
ESCUELA DE INGENIERÍAS INDUSTRIALES

JUNIO DE 2018

Resumen

Ante un accidente con múltiples víctimas, el plan de actuación de los equipos de rescate repercute directamente en la salud de los afectados. Para determinar el plan de actuación, se recurre a una fase de triage en la que el personal sanitario examina superficialmente a los heridos y les asigna una prioridad para su extracción. En este trabajo se detalla el diseño e implementación de un prototipo funcional de pinza pasiva capaz de recoger y enviar información sobre la víctima en la que se coloca a una estación base para su representación. Se describe tanto el hardware y software del sistema como la parte mecánica en la que se ubica el sistema. Asimismo, se describe el sistema de comunicación empleado, basado en tecnología Wi-Fi, y la implementación de un servidor de pruebas en una placa *Raspberry Pi* para la depuración del sistema. Finalmente, se llevan a cabo una serie de experimentos para seleccionar el mejor prototipo y se comprueba la efectividad del sistema en cuanto a la validez de los datos proporcionados por los sensores.

Declaración de Autoría

Yo, Francisco Jesús Ruiz Ruiz, estudiante del Grado en Ingeniería Electrónica, Robótica y Mecatrónica en la Escuela de Ingenierías Industriales de la Universidad de Málaga, en relación con este Trabajo de Fin de Grado, titulado: "Pulsera para Triaje y Monitorización del Estado de Víctimas", declaro que asumo la originalidad de dicho trabajo, entendida en el sentido de que no se han utilizado fuentes sin citarlas debidamente.

FIRMADO: ..... FECHA: 14 de Junio de 2018.

Índice General

Índice de Figuras

Figura **Página**

Índice de Tablas

1

INTRODUCCIÓN

1.1 Visión General

1.1.1 Motivación y Justificación

En la actualidad, los accidentes con múltiples victimas (AMV) no suceden muy a menudo, pero cuando se dan es muy importante atender a los heridos con la mayor organización y brevedad posibles para evitar posibles secuelas o incluso pérdidas humanas. El término *triage* hace referencia al proceso de clasificación de pacientes de urgencia según su gravedad con el fin de optimizar los recursos médicos disponibles. Se trata de un proceso fundamental AMV, donde el equipo sanitario es escaso y la vida de los afectados depende de las labores de rescate [8].

El triaje se lleva a cabo asignando prioridades a los pacientes según su estado. La escala de prioridad depende del sistema de triage utilizado, siendo los sistemas de 5 niveles los más extendidos debido a su eficacia [9]. En estos sistemas, cada nivel de prioridad se identifica con un número, siendo el 1 el más prioritario, un color y un nombre que determina el tiempo máximo que la víctima puede pasar sin asistencia. En la Tabla 1.1 se puede observar la escala del sistema Manchester [7].

Número	Nombre	Color	Tiempo máximo (min)
1	Atención inmediata	Rojo	0
2	Muy urgente	Naranja	10
3	Urgente	Amarillo	60
4	Normal	Verde	120
5	No urgente	Azul o negro	240

Tabla 1.1: Niveles de prioridad del sistema MTS (Manchester Triage System). (Fuente: [7])

En la actualidad, no hay un protocolo de actuación definido ante un AMV. En [10] y [11] se proponen distintos modelos de actuación que siguen unas pautas comunes. De forma general, los modelos propuestos se pueden descomponer en las siguientes fases:

1. Valoración inicial. El objetivo principal de esta fase es saber qué ha pasado, el número de víctimas y los riesgos que presenta el accidente para prever los medios sanitarios que serán necesarios y delimitar el lugar del accidente.

2. Organización del escenario. Persigue dividir y delimitar la zona afectada a fin de evitar accidentes adicionales y riesgos innecesarios. Se establecerán perímetros de seguridad y se fijarán las vías de acceso de los equipos de rescate y extracción de los heridos.

3. Triage. El equipo médico realizará una evaluación rápida de cada una de las víctimas, asignando la prioridad según el sistema de triage empleado. La prioridad asignada a cada víctima se señaliza mediante el uso de tarjetas de triage, que se encontrarán sujetas a la muñeca o al tobillo. Esta fase debe ser repetida cada cierto tiempo por personal diferente.

4. Asistencia y evacuación de las víctimas. Las víctimas serán examinadas con detenimiento en el lugar del accidente en un puesto médico avanzado según la prioridad asociada en la fase anterior. Finalmente, serán evacuadas hasta el hospital más cercano.

Debido a que en la mayoría de los casos el personal médico es escaso en comparación con el número de víctimas, la continua monitorización del estado de las mismas se hace difícil. Esto quiere decir que si el estado de uno de los afectados empeora, no se le modificará el nivel de prioridad hasta el próximo turno de triage.

1.1.2 Objetivos

En este proyecto se pretende diseñar un nodo IoT que permita monitorizar de forma remota el estado de las víctimas para su posterior clasificación. Para tal fin se desarrollará un prototipo de pinza pasiva que podrá ser colocada por el equipo de rescate o por un robot de salvamento.

Dicha pinza constará de sensores para medir la frecuencia cardíaca y la saturación de oxígeno en la sangre, así como el movimiento del sujeto. La información recogida por los sensores será procesada por un microcontrolador, y, posteriormente, será notificada a los equipos de rescate. Para ello, se empleará un indicador luminoso, que cambiará de color según el estado de la víctima, y una pantalla OLED, en la que aparecerán la frecuencia cardíaca en pulsaciones por minuto y la saturación de oxígeno en la sangre en tanto por ciento.

Toda la información recogida por los sensores será enviada a una estación base mediante comunicación inalámbrica (Wi-Fi o LoRa). La comunicación será bidireccional, es decir, desde la estación base se podrá actuar sobre un indicador luminoso y un indicador acústico.

1.2 Trabajos Relacionados

En relación a los objetivos que persigue este proyecto, existen trabajos que tratan sobre nuevas tecnologías que ayudan a realizar el triage. En la literatura se pueden encontrar artículos que tratan sobre la adquisición de datos de las víctimas mediante sensores. En [12] se describe una solución para mejorar el triage mediante una red de sensores inalámbrica que recoge las constantes vitales de los pacientes y devuelve automáticamente un análisis clínico. Una solución similar es descrita en [13], donde se emplea una pulsera para albergar los sensores.

También hay artículos que tratan sobre el procesamiento de los datos recogidos para efectuar un diagnóstico, empleando para ello tecnologías innovadoras como el "Big Data". Es el caso de [14], donde se describen nuevos sistemas de priorización teniendo en cuenta el historial clínico de las víctimas. En [15] se detalla un sistema de filtrado y procesamiento de la información recogida por los sensores para hacer una primera estimación del estado del paciente.

Por otro lado, en [1] se detalla el uso de un dron no tripulado para la búsqueda de personas y la colocación de una pulsera mediante un manipulador paralelo tipo delta. En [16] se describe la creación de una camiseta inteligente que recoge información sobre la respiración del usuario.

1.3 Estructura de la Memoria

La memoria se encuentra estructurada de la siguiente manera. En el capítulo 2, se especifica el funcionamiento deseado del sistema y se introducen los criterios de diseño que se emplearán. En el capítulo 3 se describen los elementos hardware que componen el sistema, así como su principio de funcionamiento. En el capítulo 4 se describe el programa desarrollado para el control del sistema. En el cápitulo 5 se detalla el proceso de modelado e impresión de la pinza. Por último, en el capítulo 6, se discuten los resultados, se presentan las conclusiones obtenidas y se detallan las posibles líneas de desarrollo futuras.

2

Especificaciones de Diseño

2.1 Introducción

Como se indicó en 1.1.2, se diseñará un sistema hardware basado en un microcontrolador y varios sensores que permitan recoger información sobre la salud de una persona. Se desarrollará también la parte software del sistema atendiendo a una serie de criterios de temporización. Por último, se diseñará un prototipo para alojar al sistema que deberá cumplir con una serie de pautas.

Para simplificar el proyecto, se descompondrá en tres fases atendiendo a la parte del sistema que se trata. Así pues, el proyecto contará con las fases de diseño hardware, software y mecánico, siendo este el orden en el que se abordarán.

2.2 Requerimientos funcionales

A nivel de aplicación, el sistema debe recoger información acerca de las constantes vitales de una persona y enviar esta información, por conexión inalámbrica, a un servidor web para su representación en tiempo real. Más detalladamente, las variables que se recogerán son la frecuencia cardíaca, la saturación de oxígeno en la sangre y si la persona se encuentra en movimiento. Como las constantes vitales cambian lentamente, se admitirá una latencia máxima entre envío de mensajes de 2 segundos.

A nivel físico, el sistema se alojará en una pinza de dimensiones reducidas que pueda ser acomodada al brazo fácilmente. Esta pinza actuará de forma pasiva, no será necesaria la aplicación de energía externa para que la pinza se mantenga en su posición. El sistema será implementado por completo para su posterior puesta en funcionamiento y toma de resultados. Para tal fin, se hará uso de técnicas de impresión 3D.

2.3 Funcionamiento del Sistema

Debido a que la pinza podrá ser transportada por un manipulador o por un operario, su modo de uso debe ser autónomo. Por lo tanto, la pinza deberá permanecer abierta hasta que se produzca el contacto entre el brazo de la víctima y la pinza. La pinza quedará acomodada al brazo de la víctima y se desprenderá del manipulador, para que pueda volver al campamento base a cargar otra pinza. Una vez la pinza sea colocada, comenzará a transmitir información a la estación base, Figura 2.1.

Desde la estación base, en la interfaz de usuario, el personal del equipo de rescate podrá monitorizar las constantes vitales de la víctima y enviar ordenes al nodo IoT.

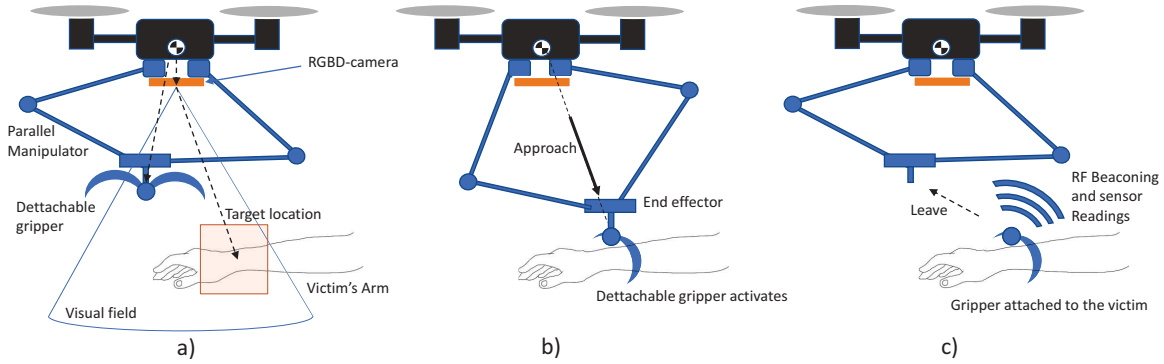


Figura 2.1: Proceso de colocación de la pinza mediante un dron. En la fase a) detecta la muñeca de la persona, en la fase b) acerca la pinza, que se cierra automáticamente, y la fase c) el dron se aleja y la pinza comienza a enviar información. (Fuente: [1])

2.4 Sistema de comunicaciones

Aunque el objetivo del proyecto no es el desarrollo de las comunicaciones, es necesaria la implementación de una pequeña interacción entre el sistema hardware objeto de este proyecto y un servidor minimalista alojado en una estación base. Por tanto, se creará un conjunto de mensajes reducidos que permita el envío de información desde el nodo IoT al servidor y el envío de órdenes desde el servidor al nodo. Serán necesarios, como mínimo, dos mensajes, uno para el envío de información y otro para el envío de órdenes.

Todas las comunicaciones entre nodo y estación base se harán mediante comunicación inalámbrica, empleando para ello tecnología Wi-Fi.

2.5 Conclusiones

En este proyecto se va a desarrollar un prototipo de pinza pasiva que pueda ser usado en labores de rescate. Debido a que este proyecto podría extenderse mucho, se ha tomado una serie de criterios de diseño que simplifiquen las distintas fases del proyecto y el funcionamiento del sistema. Como resultado se obtendrá una pinza de dimensiones reducidas que contendrá un sistema hardware capaz de comunicarse con una estación base a través de conexión Wi-Fi, implementando, para ello, una comunicación bidireccional.

3

Diseño Hardware

3.1 Selección de los Componentes

De acuerdo con los objetivos del proyecto, serán necesarios sensores para detectar movimiento, saturación de oxígeno y frecuencia cardíaca, así como un microcontrolador que procese los datos recogidos.

3.1.1 Detección de Movimiento

Los sensores más utilizados para la detección de movimiento son los acelerómetros, los giroscopios y los magnetómetros. El principio de funcionamiento de un acelerómetro se basa en la variación de la distancia de los electrodos de un condensador debido a los cambios de aceleración, cambiando la capacidad del mismo, figura 3.1. A partir de la aceleración obtenida en 3 ejes perpendiculares entre sí se puede determinar la orientación del vector gravedad, y por tanto, la orientación del dispositivo. Presentan insensibilidad ante aceleraciones constantes perpendiculares a la superficie terrestre.

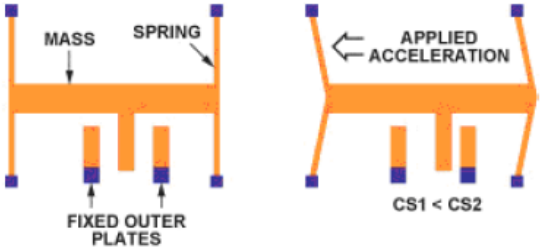


Figura 3.1: Principio de funcionamiento de un acelerómetro en una dimensión. (Fuente [2])

Un giroscopio mide la velocidad angular. Se pueden distinguir dos tipos según su principio de funcionamiento. El primero emplea una masa que puede rotar sobre un eje sostenido por uno o varios cardanes, dependiendo de los grados de libertad que se deseen, Figura 3.2.a). El segundo principio de funcionamiento se basa en un elemento que al rotar sobre un eje vibra afectado por la fuerza de Coriolis, determinando la velocidad angular, Figura 3.2.b). Para determinar la posición angular se integra la velocidad que devuelve el giroscopio, lo que introduce error de deriva o *drift* que aumenta con el tiempo, [3].

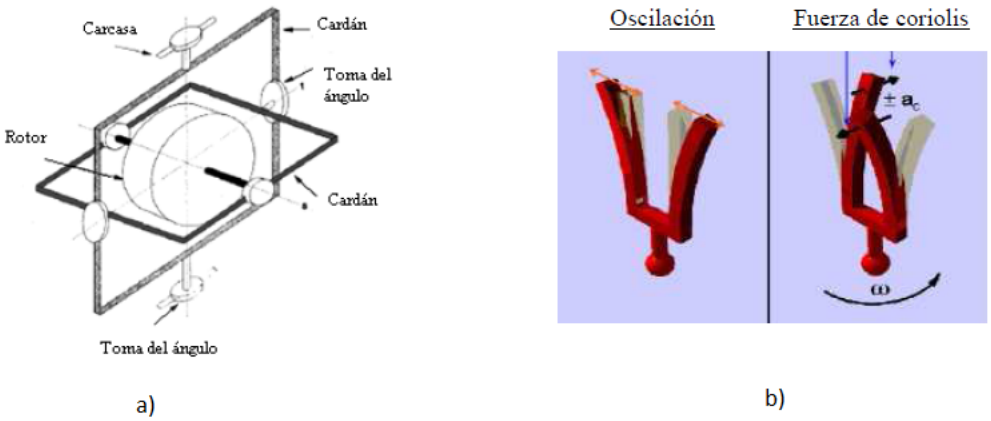


Figura 3.2: Principios de funcionamiento rotacional, a), y vibratorio, b), de un giroscopio. (Fuente: [3])

Un magnetómetro es un dispositivo capaz de medir la magnitud y dirección de un campo magnético. Se basan en el efecto Hall, o bien en resistencias que cambian su valor con un campo magnético circundante. Cuando se utiliza en aplicaciones basadas en la medida del campo magnético terrestre, los campos inducidos en la placa sobre la que se sitúa el sensor introducen errores en la medida.

Para corregir los errores de las medidas obtenidas por los sensores descritos con anterioridad se recurre a la fusión sensorial. La fusión sensorial utiliza la información proporcionada por varios sensores para complementar información que no es capaz de proporcionar un único sensor y mejorar la precisión de las medidas explotando la redundancia. Uniendo acelerómetro, giroscopio y magnetómetro se puede obtener una orientación precisa en presencia de aceleraciones lineales e interferencias magnéticas.

Por lo tanto, para detectar el movimiento, se ha empleado el módulo MPU-92/65, véase Figura 3.3, que integra el chip MPU 9250 de *Invensense* y toda la electrónica necesaria para su uso. El MPU 9250 es un dispositivo de detección de movimiento de 9 ejes que está compuesto por un acelerómetro, un giroscopio, un magnetómetro y un procesador de movimiento digital o *Digital Motion Processor* (DMP), todos ellos combinados en un chip de precio, tamaño y consumo reducidos. Proporciona fusión sensorial *on-chip* y un firmware de calibración automática. Incorpora

comunicación I^2C y SPI, con soporte para comunicación de alta velocidad. Permite la modificación de parámetros como el fondo de escala de los distintos sensores que integra o la frecuencia de muestreo. Pone a disposición del usuario el uso del DMP para realizar funciones de procesamiento, ahorrando tiempo en el microcontrolador principal. En [17] se detallan las características de este dispositivo.

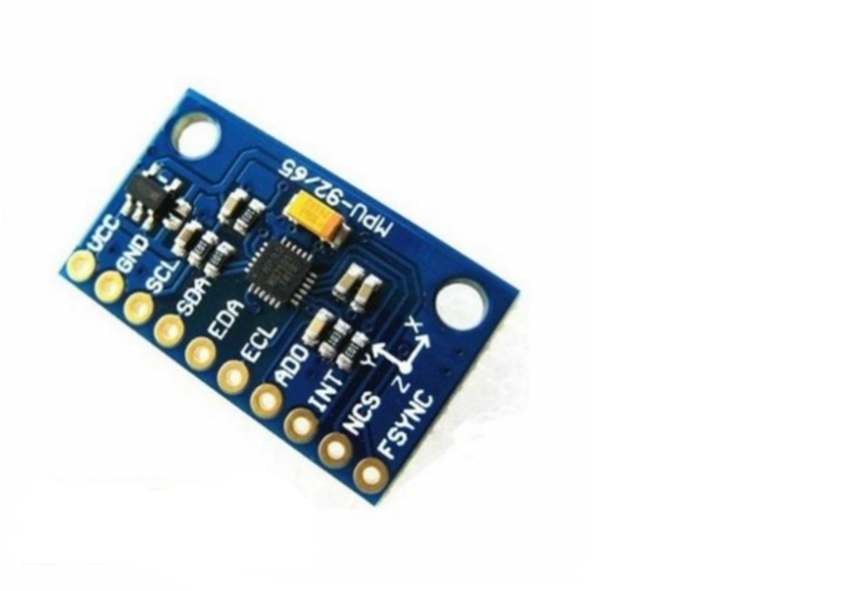


Figura 3.3: Módulo MPU 92/65. (Fuente: [4])

3.1.2 Detección de la Frecuencia Cardíaca

La frecuencia cardíaca o pulso se define como el número de contracciones que el corazón realiza en un periodo de tiempo, normalmente un minuto. El pulso cambia con la actividad física, el estado emocional, fiebre y hemorragias, estando comprendida la frecuencia cardíaca normal en reposo de un adulto entre 60 y 80 pulsaciones por minuto. Para medir el pulso se suelen emplear electrodos y principios ópticos.

Los electrodos tratan de recoger los impulsos eléctricos que llegan al corazón a través de la piel, siendo necesario el contacto directo. Dichos impulsos se caracterizan por ser una onda periódica, denominada PQRST, que corresponde con la secuencia de activación de las distintas cámaras del corazón. Comienza con la onda P, que corresponde con la depolarización y contracción de la aurícula. La sigue el complejo QRS, indicativo de la depolarización y contracción de los ventrículos. Por último, la onda T polariza los ventrículos de nuevo, haciendo que se relajen [18]. Son necesarios al menos dos electrodos y un buen sistema de amplificación y filtrado para determinar la frecuencia cardíaca.

La técnica basada en principios ópticos más extendida es la pulsioximetría. Utiliza las

características de absorción de los dos principales derivados de la hemoglobina, la oxihemoglobina (HbO_2) y la desoxihemoglobina (RHb), a través señales luminosas con una longitud de onda específica para determinar la frecuencia cardíaca y el grado de oxigenación de la sangre. Los dispositivos clásicos emplean dos diodos LED con emisiones en la zona del rojo (630 - 660 nm) y el infrarrojo (880 - 940 nm). En los sensores para pulsioximetría, la disposición de emisores y receptores se suele hacer en dos configuraciones diferentes, distinguiendo así entre los sensores de transmisión y los de reflexión, Figuras 3.4.a) y 3.4.b) respectivamente.

En los sensores de transmisión, el emisor y el receptor se encuentran enfrentados, quedando entre ellos el tejido sobre el que se va a medir. Esta configuración presenta el inconveniente de que el sensor se debe colocar en lugares determinados del cuerpo, como pueden ser los dedos o los lóbulos de las orejas. En los sensores por reflexión, emisor y receptor se tienen la misma orientación y localización, por lo que pueden ser colocados fácilmente en cualquier parte del cuerpo. La principal desventaja proviene de la señal recibida, que es más débil que la obtenida mediante transmisión [19].

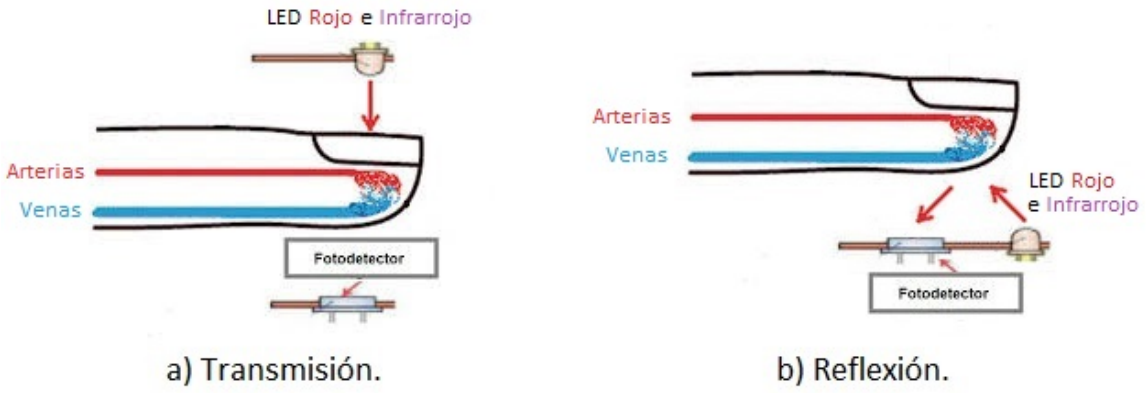


Figura 3.4: Tipos de pulsioxímetros.

Por su versatilidad en la localización del sensor, se ha utilizado un pulsioxímetro de reflexión, más concretamente el MAX 30102 de *Maxim Integrated* (Figura 3.5). Dicho sensor consta de 2 diodos LED, uno rojo y uno infrarrojo, y los circuitos de control de los mismos y sus respectivos receptores. Posee comunicación mediante I^2C, lo que permite controlar parámetros como la intensidad que se suministra a los emisores, la resolución de las medidas (entre 15 y 18 bits) y el número de LEDs activos, entre otras opciones. Dispone de una memoria FIFO capaz de almacenar hasta 32 muestras cuando todos los LEDs están activos y a máxima resolución. En [20] se detallan las características eléctricas del dispositivo.

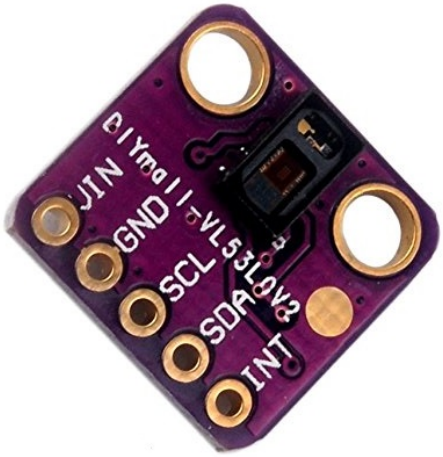


Figura 3.5: Pulsioxímetro MAX 30102. (Fuente: [5])

3.1.3 Microcontrolador

Un microcontrolador (MCU) es un circuito integrado programable, capaz de ejecutar las instrucciones almacenadas en su memoria. Está compuesto, principalmente, por la unidad central de procesamiento, la memoria y los periféricos de entrada y salida. El microcontrolador es el encargado de procesar la información recogida por los sensores y, a en función de ella, actuar sobre las salidas.

Para la tarea descrita en este proyecto, es necesario que el MCU a utilizar disponga de comunicación Wi-Fi y tamaño y consumo reducidos. Es por estos motivos por los que se ha seleccionado el ESP-32 de *Espressif Systems*, más concretamente el módulo Wi-Fi LoRa 32 de *Heltec Automation* (Figura 3.6).

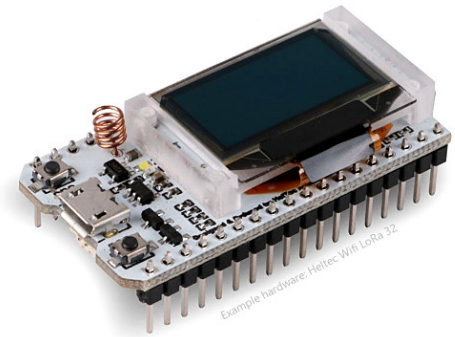


Figura 3.6: Módulo Wi-Fi LoRa 32 de *Heltec Automation*. (Fuente: [6])

Dicho controlador tiene dos núcleos de procesamiento de 32 bits, gobernados por un reloj interno de velocidad ajustable entre 80 y 160 MHz, 520 KB de memoria SRAM y 29 pines de entrada/salida. Dispone de soporte para comunicación tanto inalámbrica (Wi-Fi, Bluetooth) como

a través de conductores (I^2C, SPI, CAN, etc). El módulo de *Heltec* añade conexión mediante LoRa, una pantalla OLED, un terminal para conectar una batería y la circuitería necesaria para su carga.

3.2 Conexión del Sistema

Una vez seleccionados los componentes del sistema, se definirá su conexión. Para las comunicaciones entre los sensores y el microcontrolador se empleará el bus I^2C con una resistencia de pull-up de 2,7 $K\Omega$. Se emplearán un zumbador piezoeléctrico y un anillo de LEDs RGB conectados a los GPIO para la señalización de la víctima. El sistema se alimentará mediante una batería Li-Po de 3,7 V y 750 mAh. En la Figura 3.7 se muestra el esquema de conexión del sistema.

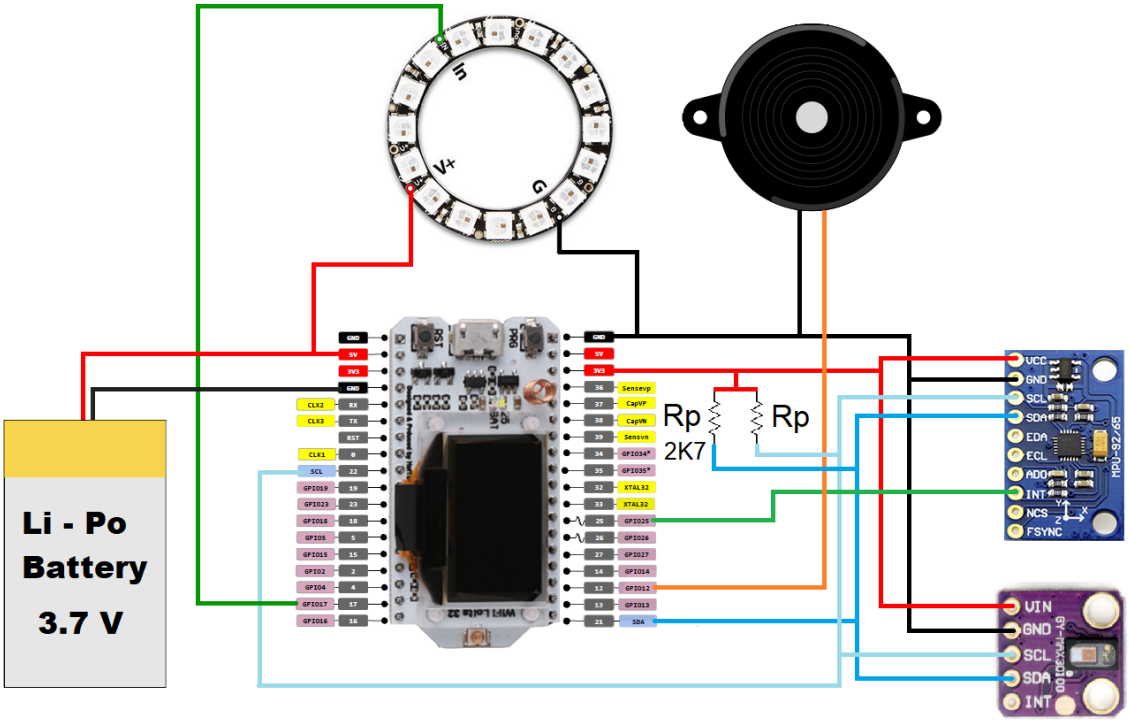


Figura 3.7: Esquema de conexión del circuito.

4

Diseño Software

4.1 Introducción

Como se indicó en 1.1.2, se va a desarrollar un nodo IoT. El término Internet de las cosas o *Internet of Things* (IoT), hace referencia a un nuevo enfoque tecnológico en el que todos los objetos que nos rodean están conectados a la red y pueden interactuar unos con otros. La conexión inteligente con las redes existentes (Wi-Fi y 4G entre otras) es una parte fundamental del IoT. Los pilares básicos de este nuevo paradigma son:

1. Conocimiento compartido del estado del usuario y otros dispositivos.

2. Arquitecturas Software y comunicación mediante redes generalizadas para procesar y transmitir la información hacia donde es relevante.

3. Herramientas para lograr un comportamiento autónomo e inteligente.

Por lo tanto, el IoT puede ser entendido como una red de objetos interconectados que no solo recogen información del entorno e interaccionan con el mundo físico, sino que también utilizan Internet para proveer servicios de transferencia de información, análisis y comunicaciones [21]. Existen multitud de protocolos de comunicación para las aplicaciones del IoT, siendo los más empleados los siguientes:

- **HTTP (HyperText Transfer Protocol).** Es un protocolo basado en el modelo cliente servidor. En este protocolo, el cliente envía un mensaje, con un formato definido, para solicitar información al servidor [22].

- **WebSocket.** Proporciona comunicación bidireccional entre un cliente y un servidor basándose para ello en una conexión fiable mediante TCP. Para ello emplea mensajes más simples que HTTP [23].

- **CoAp (Constrained Application protocol).** Se trata de un protocolo no fiable creado para dispositivos con recursos limitados. Emplea un sistema de comunicación máquina a máquina (M2M) bajo una filosofía de petición y respuesta.

- **MQTT (Message Queue Telemetry Transport).** Al igual que CoAp, está diseñado para dispositivos con recursos limitados, conectados a redes con bajo ancho de banda y no fiables. Tiene una filosofía de publicación/suscripción [24].

Además del nodo IoT, se programará un pequeño servidor que sirva para demostrar el funcionamiento del nodo objeto de este proyecto. Este servidor se alojará en un equipo con mayores recursos, y su función será la de mostrar los datos recibidos y generar órdenes simples.

4.2 Estación Base

4.2.1 Node-Red

La estación base estará compuesta por una placa *Raspberry Pi 3* en la que se ha programado un servidor minimalista. Para ello se ha usado *Node-Red*, una herramienta de programación basada en flujos que permite el desarrollo de aplicaciones on-line de una forma fácil y rápida. En esta herramienta, la programación se lleva a cabo mediante bloques que se unen a través de líneas de flujo. *Node-Red* dispone de una amplia librería de bloques con funciones muy diversas, siendo posible la programación de funciones más específicas mediante JavaScript y su difusión como archivos JSON [25].

JSON *(JavaScript Object Notation)* es un formato de intercambio de información ligero, basado en JavaScript, que puede ser generado y leído por multitud de lenguajes de programación (C, Java, Perl, Python, etc.). Este tipo de archivos puede ser construido en base a dos estructuras básicas [26]:

- Como un conjunto de pares nombre/valor llamados *objetos*. El campo *nombre* se emplea como un puntero que señala a su campo *valor*, que puede contener un número, una cadena de caracteres, una lista de valores (array) o incluso otro objeto. Los pares se separan por comas. Los objetos comienzan con una llave de apertura ({) y finalizan con una llave de cierre (}).

- Como una lista de valores (array). Al igual que ocurre en los objetos, las listas de valores pueden contener números, cadenas de caracteres, objetos e incluso otras listas de valores. Las listas de valores se definen entre corchetes ([]).

A continuación se muestra un ejemplo de objeto JSON:

$$\{\,"numero":1,\ "cadena":"hola",\ "lista":[1,2,3]\,\}$$

Node-Red y el formato JSON están íntimamente ligados, siendo los datos que viajan por las líneas de flujo objetos JSON. Así pues, cada bloque de función actúa sobre el campo del objeto indicado y devuelve a su salida el mensaje de entrada con sus campos modificados.

Node-Red dispone de una librería de funciones para crear una interfaz de usuario de una forma extremadamente simple. Esta librería recibe el nombre de *dashboard*. Contiene bloques de salida que permiten insertar gráficas, indicadores numéricos y texto. Con los bloques de entrada se pueden introducir controles de tipo binario (pulsadores y switches) e incluso recoger datos introducidos por el usuario. Todos los bloques son personalizables, dejando a elección del programador el orden en el que se muestran los elementos en la pantalla y las respuestas generadas por los controles.

4.2.2 Protocolo MQTT

Como se indicó anteriormente, MQTT es un protocolo de comunicación simple y ligero, diseñado para dispositivos con recursos limitados, como es el caso de nuestro sistema. Una red MQTT presenta una topología de estrella, donde los nodos conectados a la red se comunican con un servidor central o *broker*. Cada nodo puede crear hilos de comunicación o *topics*. Los diferentes dispositivos conectados a la red se suscriben y publican en los *topics*, de forma que solo los equipos suscritos a un *topic* reciben los mensajes publicados en ese *topic*. Los *topics* se ordenan siguiendo una estructura jerárquica, de forma que si un nodo se suscribe a un *topic* recibe los mensajes de todos los *subtopics* asociados al primero [24]. En la Figura 4.1 se muestra el esquema de la red MQTT del sistema objeto de este proyecto.

Este protocolo ofrece tres tipos de calidad de servicio (QoS). Un valor de QoS 0 asegura que el mensaje se publicará en el topic, pero no asegura su recepción en los demás nodos. Un QoS 1 indica el mensaje se publicará en el topic, asegurando también su recepción en los nodos suscritos, pero no impide que el mensaje sea recibido más de una vez. Un QoS 2 asegura que el mensaje se publicará correctamente en el topic y que los nodos suscritos lo recibirán una sola vez, siendo esta opción la más fiable.

Los mensajes en el protocolo MQTT son objetos JSON compuestos por los siguientes campos:

- **Identificador (*packetId*).** Contiene un número, asignado por el broker, que identifica al mensaje.
- ***Topic* destino (*topic*).** Almacena el nombre del *topic* en el que se publica el mensaje.
- **Calidad de servicio (QoS).** Contiene un valor numérico que indica al servidor el tipo de QoS seleccionado.

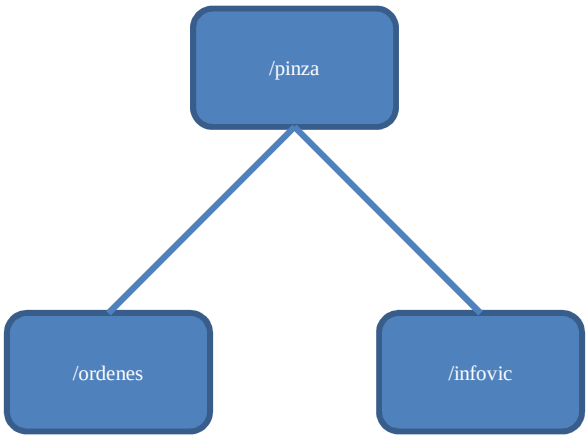


Figura 4.1: Esquema de la red MQTT.

- **Almacenamiento del mensaje (*retainFlag*).** Indica al broker que debe almacenar el mensaje. Permanecerá en el *topic* hasta que se mande otro mensaje, por lo que los nuevos nodos que se conecten a dicho *topic* lo recibirán.

- **Datos (*payload*).** Este campo almacena los datos útiles del mensaje, es decir, los datos que el usuario envía.

Al tratarse de un objeto JSON, el campo *payload* del mensaje puede contener, como se indicó en 4.2.1, un valor numérico, una cadena de caracteres, una lista u otro objeto.

4.2.3 Código de la Estación Base

Antes de comenzar con la programación de la estación base, es necesario instalar un servidor MQTT. En esta aplicación se ha usado *Mosquitto*, un servidor open-source, sencillo e intuitivo. Una vez instalado, se procede a la programación de la estación base.

El código puede ser diferenciado en dos partes. La primera parte, consiste en la representación de los datos recibidos, un programa corto en el que solo es necesario conocer el *topic* por el que se reciben los datos (/pinza/infovic) y la estructura de los mismos. Para no hacer uso de una gran variedad de mensajes, se han introducido todos los datos en un objeto JSON. En la Figura 4.2 se puede observar un ejemplo de mensaje de entrada al servidor. Ante la entrada de un mensaje nuevo, el programa descompone el campo *payload* del mensaje en los distintos campos y los representa en el *dashboard*. En la Figura 4.3 se observa el flujo diseñado para la recepción de mensajes en la estación base. En la Figura 4.4 se muestra el *dashboard*. Se puede observar los indicadores de frecuencia cardíaca, saturación de oxígeno y movimiento, así como el panel de control.

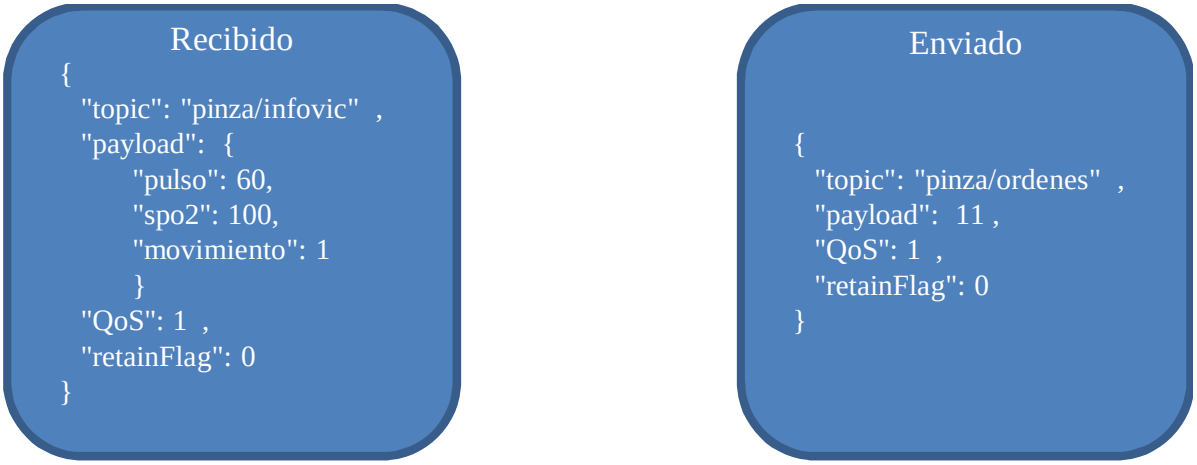


Figura 4.2: Estructura de los mensajes recibidos y enviados desde la estación base.

La segunda parte del código de la estación base hace referencia al envío de mensajes desde la base al nodo sensor. Los mensajes que se mandan desde la estación base al nodo son más simples, ya que solo contienen la información del estado del zumbador y un diodo LED. En este caso, el campo *payload* adquiere un valor numérico, en el que el dígito en la posición de las decenas corresponde al estado del zumbador +1 y el de las unidades al estado del LED +1. De esta forma, el valor enviado siempre estará compuesto por dos cifras, facilitando la labor de procesamiento de la información en el programa del nodo sensor. El campo *topic* adquiere el valor /pinza/ordenes, que corresponde al *topic* al que el nodo se suscribe. En la Figura 4.2 se puede observar un ejemplo de los mensajes enviados desde la estación base. En la Figura 4.3 se muestra el flujo diseñado para el envío de los mensajes.

4.3 Nodo Sensor

El software se ha escrito en lenguaje C en el entorno de desarrollo integrado o IDE (*Integrated Development Environment*) de Arduino. En este entorno de desarrollo los programas se dividen en dos partes, diferenciadas por pertenecer a dos funciones, *setup()* y *loop()*. La primera parte del código, función *setup*, solo se ejecuta una vez. En ella se escriben las instrucciones de inicialización necesarias. Por otro lado, la función *loop* compone el bucle principal del programa, ejecutándose de forma cíclica tras la primera ejecución de la función *setup*.

Antes de empezar a escribir el fichero principal del programa, se han escrito algunas librerías básicas. La primera de ellas es la librería de comunicación mediante el bus I^2C, archivo I2C.h adjunto en el CD del proyecto. Esta librería, basada en la librería *Wire* de Arduino, contiene funciones básicas de lectura/escritura de uno o varios registros en una única consulta.

Se ha escrito también una librería para el MPU-92/65, que contiene el mapa de registros completo y funciones para la lectura de los datos. Para detectar movimiento se ha hecho uso del

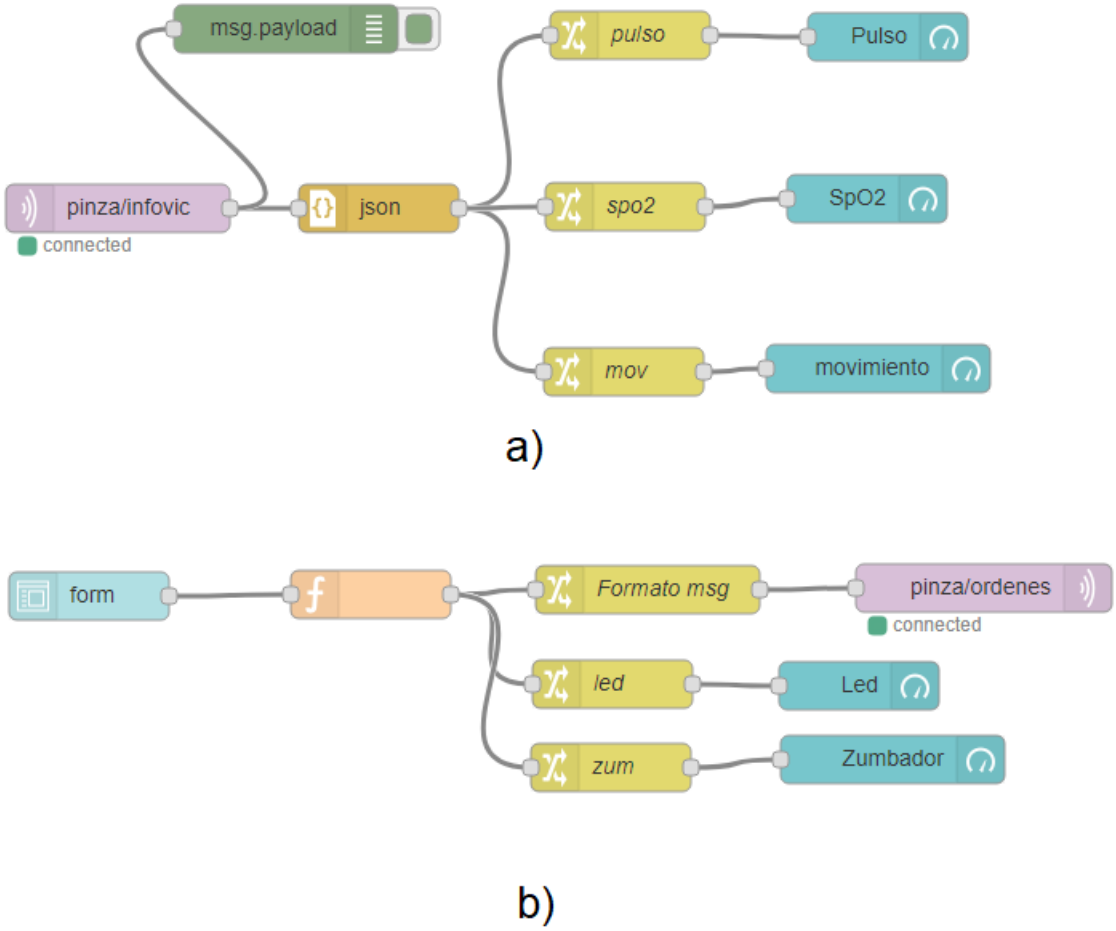


Figura 4.3: Flujo *Node-Red* de recepción, a), y envío, b), de mensajes.

DMP, más concretamente, del modo *Wake On Motion*. Este modo activa el pin de interrupción del módulo cuando hay una variación en los datos almacenados en los registros del acelerómetro superior a un umbral preestablecido, permitiendo la detección de movimientos sin ocupar tiempo de ejecución en el controlador principal del sistema. Más adelante se explica cómo se ha implementado la rutina de procesamiento de interrupciones en el ESP. Para el control del pulsioxímetro MAX 30102 se ha utilizado una librería escrita por *Sparkfun* [27] que contiene todas las funciones necesarias para su uso.

Atendiendo a la tarea que realiza cada conjunto de instrucciones, el código principal se puede descomponer en tres secciones diferentes, Figura 4.5.

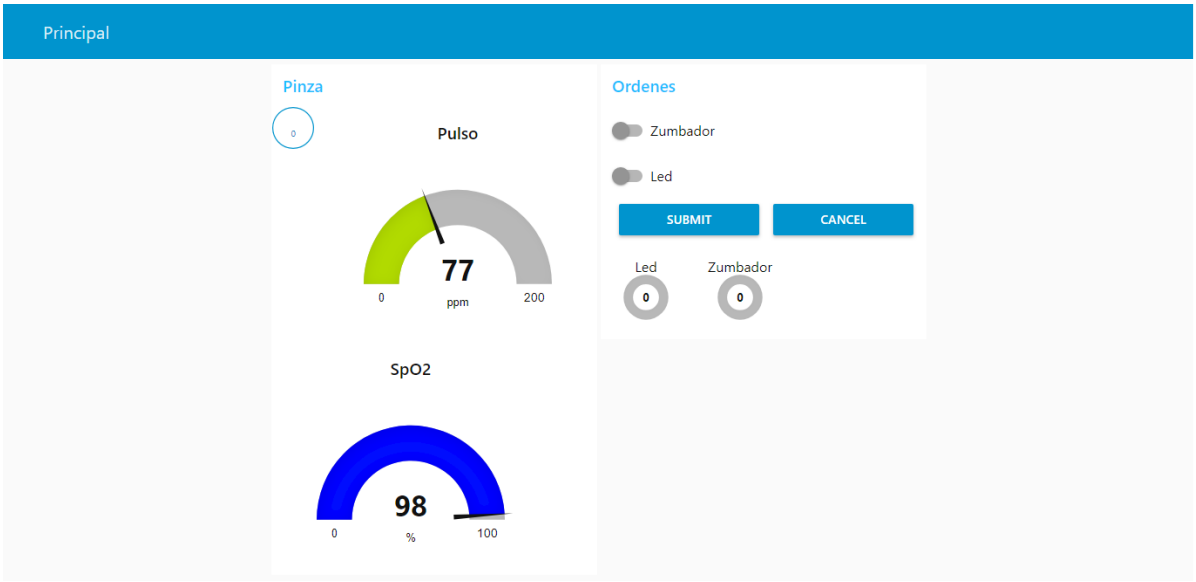


Figura 4.4: Interfaz de usuario de *Node-Red*, el *Dashboard*.

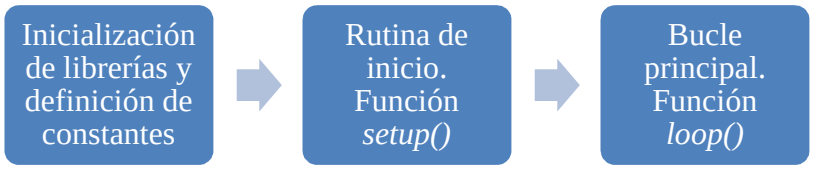


Figura 4.5: Esquema de la estructura general del programa.

4.3.1 Inicialización de librerías y constantes

En la primera parte del código, se declaran las constantes y librerías necesarias para el correcto funcionamiento del programa. Las librerías necesarias son:

- **WiFi.h**. Contiene todas las funciones relacionadas con la comunicación inalámbrica mediante Wi-Fi.
- **PubSubClient.h**. Contiene funciones de comunicación mediante el protocolo MQTT.
- **I2C.h**. Contiene funciones básicas para la comunicación en I^2C. Su código fuente se encuentra en el archivo del mismo nombre adjunto en el CD del proyecto.
- **MAX30105.h**. Contiene todas las funciones necesarias para la utilización del sensor MAX30102.

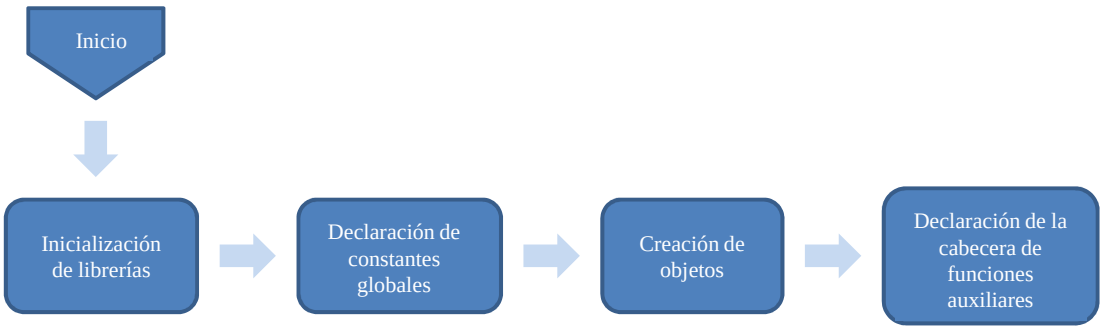


Figura 4.6: Esquema general del orden de ejecución de la sección de inicialización.

- **spo2_algorithm.h**. Contiene las funciones para el cálculo de la frecuencia cardíaca y la saturación de oxígeno.
- **MPU9250.h**. Contiene el mapa de registros completo y algunas funciones básicas del módulo MPU 92/65. Su código fuente se encuentra en el archivo del mismo nombre adjunto en el CD del proyecto..
- **imagenes.h**. Contiene las imágenes que se muestran en la pantalla OLED en formato XBM. Su código se encuentra en el archivo del mismo nombre adjunto en el CD del proyecto.
- **SSD1306.h**. Contiene todas las funciones referentes a la pantalla OLED.
- **Adafruit_NeoPixel.h**. Contiene funciones para el control del anillo de LEDs RGB.

Tras la definición de las librerías que se van a usar, se declaran constantes globales a todo el programa, como son el umbral del modo *Wake On Motion*, los pines que serán usados y los datos de la red Wi-Fi y el servidor MQTT. Posteriormente, se crean e inicializan los objetos necesarios (manejador de interrupciones, objetos para el control de los sensores y comunicaciones) y la cabecera de las funciones auxiliares escritas. En la Figura 4.6 se muestra el esquema de funcionamiento de la primera parte del código.

4.3.2 Rutina de Inicio. Función *Setup()*

La segunda sección del código corresponde a la rutina de inicio, escrita dentro de la función *setup()*, cuyo código fuente se encuentra en el archivo adjunto en el CD. En la Figura 4.7 se muestra un esquema general del orden de ejecución de dicha rutina. En los párrafos siguientes se describe un comportamiento más detallado de esta sección.

Antes de comenzar con la configuración de los sensores, se inicializan las comunicaciones, empezando por iniciar la interfaz I^2C que se va a usar (el ESP 32 dispone de dos interfaces I^2C), seguido del establecimiento de las comunicaciones inalámbricas (Wi-Fi y MQTT). Una vez

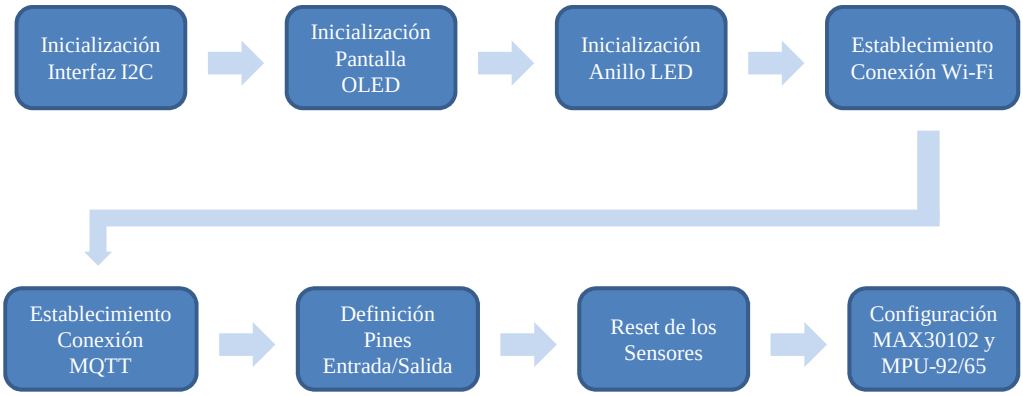


Figura 4.7: Esquema general del orden de ejecución de las instrucciones de la rutina de inicio.

establecidas las comunicaciones, se sigue con la definición de los pines de entrada/salida y la asociación de interrupciones.

Así pues, se establecen como salidas el pin correspondiente al zumbador, y como entrada, el pin asociado a la interrupción del módulo MPU-92/65. Se le a asociado a este último pin una interrupción de flanco de bajada o tipo *FALLING*, ya que, según se especifica en la hoja de características del MPU, el pin de interrupción cambia de estado *HIGH* (+3,3 V) a estado *LOW* (0 V) cuando tiene lugar una interrupción.

Posteriormente, se lleva a cabo un proceso de reset en los sensores para descartar cualquier posible configuración anterior. Acto seguido, se procede a la configuración de los módulos MAX30102 y MPU-93/65. En la configuración del MPU, solo es necesario indicar el fondo de escala, establecida a 16 g, ya que se usará el modo *Wake On Motion*, configurado mediante la función *WOM_Config*. La sensibilidad de este modo ha sido ajustado experimentalmente.

En la configuración del modulo MAX30102 interviene un conjunto de parámetros más amplio, teniendo que establecerse un equilibrio entre velocidad y precisión. Los valores con los que se ha configurado el sensor son los siguientes:

- **Intensidad suministrada a los LEDs.** Otro parámetro que el MAX30102 permite configurar es la intensidad suministrada a los LEDs en un rango que va desde 0 A (LED apagado) hasta los 50 mA, codificado como un entero de 8 bits. Este valor repercute directamente en el valor de la señal devuelta. Se ha determinado de manera experimental.

- **Promediado de lecturas.** Se puede indicar al MAX30102 que el valor devuelto en la lectura sea la media de un número previamente definido de lecturas, tal como se muestra en la expresión 4.1. Esto ayuda a la hora de eliminar el ruido que se pueda introducir en la

señal, haciendo el sistema más preciso pero también más lento. Puede devolver el promedio de hasta 32 medidas en una sola lectura. Con el fin de no hacer el sistema demasiado lento, se ha establecido a 8.

$$Lectura = \frac{Medida_1 + \cdots + Medida_n}{n} \tag{4.1}$$

- **LEDs activos.** El MAX30102 dispone de dos diodos LED que emiten en diferentes longitudes de onda (660 nm en el caso del LED rojo y 880 en el caso del infrarrojo). Dichos LEDs pueden ser encendidos y apagados según crea conveniente el usuario. Para medir la frecuencia cardíaca solo es necesario el LED rojo. Como se explicó en 3.1.2, son necesarios al menos dos longitudes de onda diferentes para medir la saturación de oxígeno en sangre, por lo que se hará uso de ambos LEDs.

- **Frecuencia de muestreo.** Se parte de que la frecuencia cardíaca viene dada por una señal periódica. Si se acota la frecuencia cardíaca máxima a 120 pulsaciones por minuto (2 pulsaciones por segundo), se puede decir que se trata de una señal que cambia con una frecuencia no superior a 2 Hz. De acuerdo con el teorema de muestreo de Nyquist-Shannon, expresión 4.2, la frecuencia de muestreo de una señal periódica debe ser mayor que el doble de la frecuencia de la señal [28], que en este caso es 4 Hz. La frecuencia de muestreo más baja configurable es de 50 muestras por segundo, valor que cumple con un amplio margen el teorema de muestreo. Con el fin de hacer el sistema más rápido, la frecuencia de muestreo asignada al MAX30102 será de 100 muestras por segundo.

$$F_s \geq 2F_{max} \tag{4.2}$$

- **Ancho de pulso.** Este parámetro hace referencia al tiempo de exposición o apertura, es decir, al tiempo que un emisor y su receptor correspondiente estarán activos. Una vez más aparece esta relación entre precisión y velocidad, un tiempo de exposición pequeño hace al sistema más rápido pero menos preciso y al contrario. El ancho de pulso está fuertemente relacionado con la frecuencia de muestreo, siendo imposibles algunas configuraciones (por ejemplo, no se puede establecer una frecuencia de 1000 muestras por segundo con un ancho de pulso de 411 μs). Con el fin de obtener la mayor resolución posible se establecerá el tiempo de exposición mínimo, 411 μs.

- **Rango.** El rango determina la diferencia entre los valores mayor y menor posibles. Se especifica mediante el valor del fondo de escala en nA. Por criterios de diseño se ha impuesto a 4096 nA. Este parámetro solo interviene cuando se consigue una resolución en la medida de 18 bits. La resolución de las medidas puede variar entre 15 y 18 bits dependiendo de la configuración que se haya escogido.

La configuración de los sensores marca el fin de la rutina de inicio, dando paso al bucle principal del programa.

4.3.3 Bucle Principal. Función *Loop()*

La tercera sección del código corresponde al bucle principal, escrito dentro de la función *loop*. En la figura 4.8 se muestra de forma general el funcionamiento de bucle.

De forma más detallada, antes del comienzo del bucle se declaran las variables de control. Acto seguido, ya dentro de la función *loop* representada en el esquema por el rectángulo azul oscuro, se recogen 100 muestras de los receptores rojo e infrarrojo y se almacenan en un array. Después se ejecuta la función *maxim_heart_rate_and_oxygen_saturation()*, perteneciente a la librería *spo2_algorithm.h*, que recibe como parámetros los array de muestras y devuelve el valor de la frecuencia cardíaca, la saturación de oxígeno y una variable que indica si los valores son válidos. En el caso de que sean válidos, se almacenan en un array y en el caso contrario se descartan.

Se inicia ahora un bucle infinito, representado por el rectángulo azul claro, en el que primero se eliminan las 25 primeras muestras del array y se desplazan las restantes. Posteriormente, se calculan de nuevo los valores de frecuencia cardíaca y saturación de oxígeno con el mismo procedimiento que se indicó anteriormente, se comprueba la validez de los resultados y se almacenan o desechan según la misma. Luego, se realiza la media de los valores almacenados en memoria, teniendo en cuenta que se almacenan 4 datos como máximo. El resultado de esta operación es el valor final que se muestra en la pantalla y se manda a la estación base. Partiendo de los resultados anteriores, se comienza el proceso de triage.

Primero se evalúa el pulso para comprobar si éste es demasiado bajo o nulo. En este caso, la víctima presenta una prioridad no urgente (color negro o LEDs apagado). En caso contrario, se atiende al nivel de oxígeno en sangre teniendo en cuenta que en un adulto sano los niveles normales superan el 95%. En caso de que el nivel de SPO2 sea superior al 95%, la víctima presenta una prioridad no urgente (color verde). Si el nivel de SPO2 se encuentra entre el 90% y el 95%, la víctima presentará una prioridad urgente (color amarillo) en el caso de que su pulso sea normal (inferior a 85 ppm) o muy urgente (color rojo) en caso contrario. En el caso de que el nivel de SPO2 sea menor del 90%, la víctima presentará prioridad muy urgente (color rojo).

Tras valorar la gravedad de la victima, se comprueba si ha ocurrido alguna interrupción del MPU, y en caso afirmativo, se actualiza la variable que lo indica. Una vez que se han recogido todos los datos, se compone el mensaje MQTT. El mensaje se envía por el topic *"pinza/infovic"*, siendo su payload una cadena de caracteres con formato JSON formada por los campos *pulso*, *spo2* y *movimiento*. Acto seguido se comprueba si hay mensajes pendientes de ser procesados. Después de procesar los mensajes recibidos, en caso de que los hubiera, se reinicia el bucle, descartando de nuevo las 25 primeras muestras del array.

4.3.4 Funciones Auxiliares

Se han escrito una serie de funciones auxiliares, cuyo código fuente se puede ver en el archivo adjunto en el CD. Corresponden al tratamiento de interrupciones, tanto las producidas por el

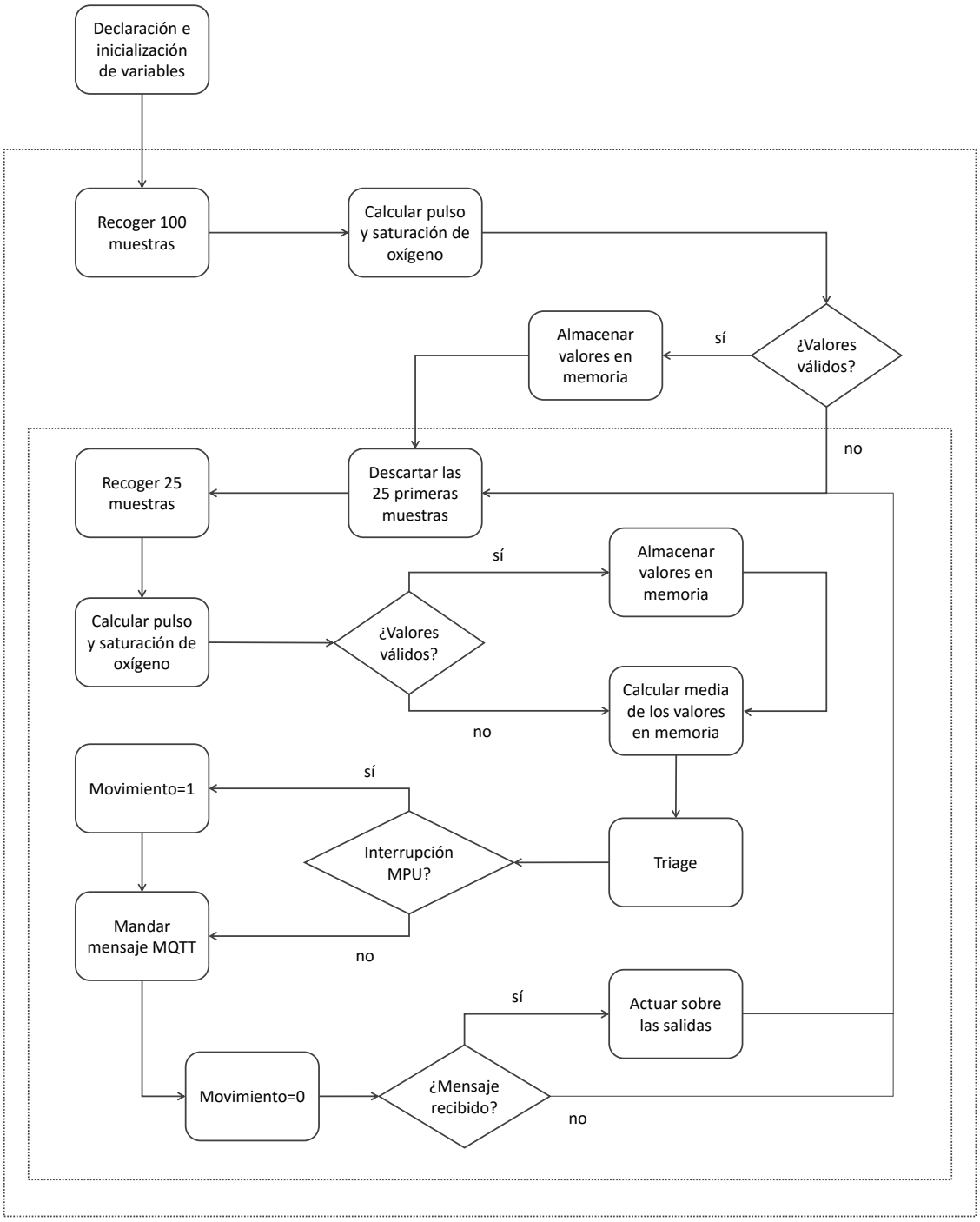


Figura 4.8: Esquema general del orden de ejecución del bucle principal.

MPU como las producidas por mensajes de entrada, y al cambio de color de los LEDs RGB.

La función para el tratamiento de interrupciones debidas al modo *Wake On Motion*, consiste en el cambio del valor de una variable para indicar de esta forma al programa principal que hay movimiento. La función de interrupción MQTT, debida a la recepción de un mensaje, extrae la información útil del mensaje y la almacena en una variable global. También cambia el valor de una variable que actúa como indicadora del evento. La función del cambio de color consta de un bucle de 16 iteraciones, una por cada LED, que asigna el color que recibe como parámetro a cada LED del anillo.

5

Diseño Mecánico

Esta fase comienza con el diseño del mecanismo que provoca la apertura y el cierre de la pinza. Como se indica en el título de este proyecto, se trata de una pinza pasiva, es decir, no es necesaria la aplicación de energía externa para que desempeñe su función. Teniendo esto en cuenta, se ha diseñado un mecanismo basado en un muelle de tracción que conmuta la posición de la pinza entre dos estados estables: abierta y cerrada. Esto se consigue haciendo que el estado de máxima elongación del muelle se encuentre en el punto de rotación de las partes móviles de la pinza. En la Figura 5.1 se muestra el esquema de funcionamiento, donde se puede ver que en los estados a) y c) el muelle hace que la pinza permanezca abierta y cerrada respectivamente. El estado b), representa la transición del estado a) al c) y viceversa.

El diseño de la pinza se ha realizado en el programa *SolidWorks* y, posteriormente, se ha impreso un prototipo con una impresora 3D. Se han seguido dos filosofías diferentes para el diseño del prototipo.

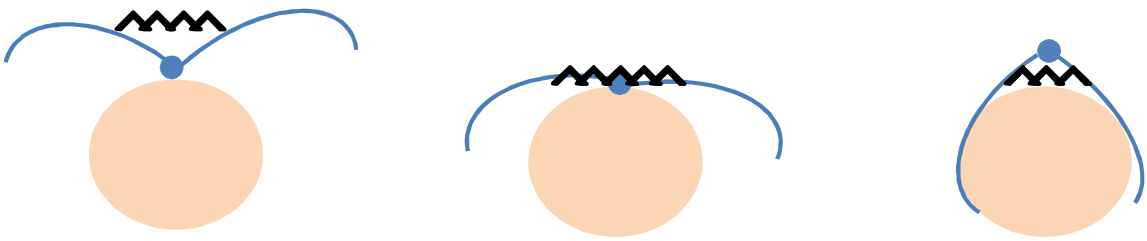


Figura 5.1: Esquema de funcionamiento del mecanismo de la pinza.

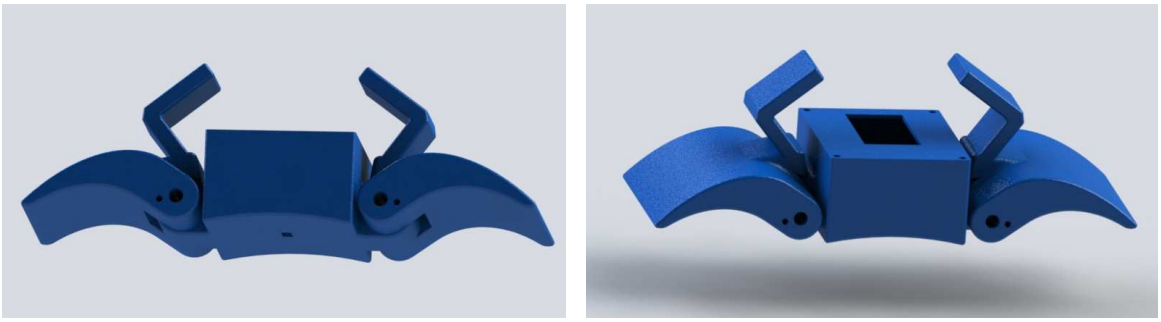


Figura 5.2: Primer prototipo de la pinza, modelo de 3 piezas.

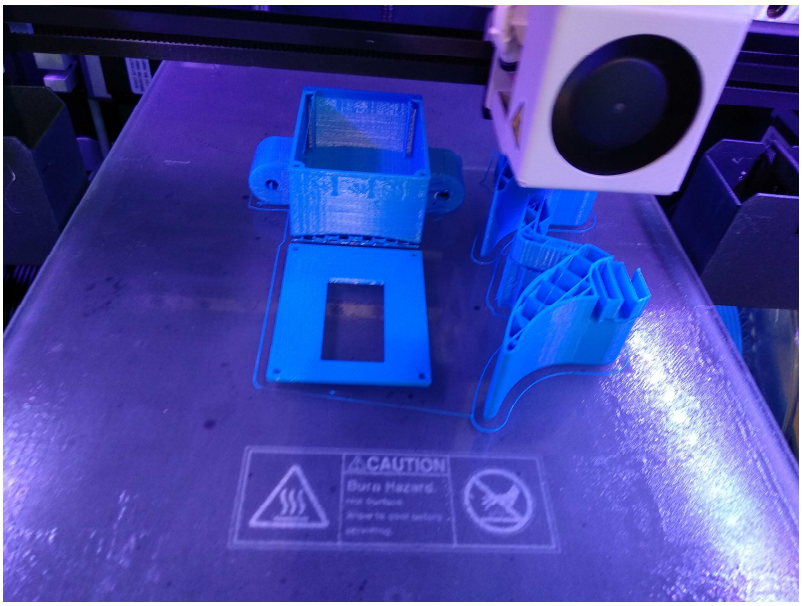


Figura 5.3: Proceso de impresión 3D del primer prototipo.

5.1 Primer Prototipo. Diseño Centralizado

En el primer prototipo se diseñó una caja que contiene todos los elementos hardware del sistema, asegurando, así, un diseño lo más compacto posible. Partiendo de esa caja primitiva, se diseñaron las partes móviles de la pinza y se ubicaron los ejes de rotación, obteniéndose un prototipo formado por tres piezas, la caja central, la paleta izquierda y la paleta derecha. En la Figura 5.2 se muestra el modelo diseñado en *SolidWorks*.El prototipo se materializó haciendo uso de una impresora 3D (como se observa en la Figura 5.3).

Las dimensiones de la caja primitiva son aquellas que permiten el almacenamiento de todos los componentes, dejando cierto margen al cableado, obteniéndose una caja de 44 x 27 x 57 (todas las medidas están representadas en milímetros, mm). Presenta cierta curvatura en la parte inferior para que se adapte mejor a la forma de la muñeca. Dentro de la caja central, los componentes se ubican a diferentes alturas. Así, el módulo MAX30102 se ubica en la parte

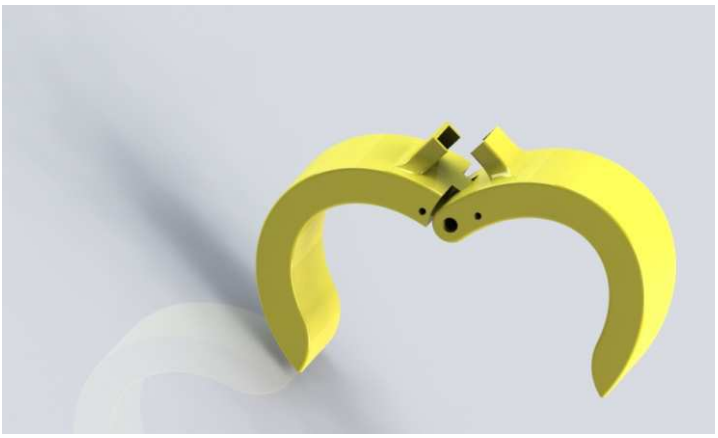


Figura 5.4: Segundo prototipo de la pinza, modelo de dos piezas inicial.

más baja de la caja, utilizando un orificio para ponerlo en contacto con la piel de la víctima. El ESP32 se coloca en la parte más alta de la caja, de forma que la pantalla pueda ser observada sin dificultad a través de la abertura de la tapadera. Por último, la batería y el MPU-92/60 se ubican en las capas intermedias. El punto fuerte de este diseño es que todos los componentes están centralizados, facilitando el proceso de cableado. Al ser un diseño de 3 piezas, se corre el riesgo de que una de las paletas de la pinza no se cierre, con lo que el diseño no cumpliría su función.

5.2 Segundo Prototipo. Diseño Funcional

Para prevenir el efecto descrito en 5.1, se ha diseñado un segundo prototipo, en el que se ha buscado primero la funcionalidad. Como resultado, se ha obtenido un prototipo formado por dos piezas, Figura 5.4.

Al ser las dimensiones de la pieza reducidas, 10 mm de espesor y 30 mm de profundidad, el problema que surge ahora es la localización de los componentes. Se han creado diversos compartimentos para alojar los distintos sensores a lo largo de la pinza, intentando no crear un prototipo demasiado voluminoso. El resultado es un diseño más compacto que en el primer prototipo, que no presenta el error del que se hablaba al principio de este punto. En la figura 5.5 se puede observar el modelo final de la pinza.

Contiene un compartimento para cada sensor, excepto en el caso del MPU, que se encuentra junto al MAX30102. En la tapadera del compartimento del ESP32 se ha abierto un orificio para mejorar el acceso a la pantalla OLED. Sobre la tapadera del compartimento de la batería se ha colocado el anillo de LEDs. Para ello, se ha extruido una estructura en forma de cruz cuya longitud es la del diámetro interior del anillo. Posteriormente, se ha diseñado una carcasa que

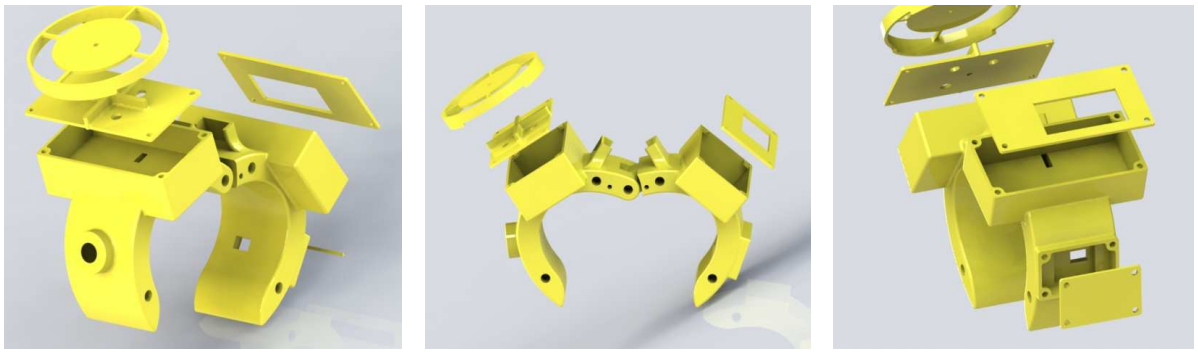


Figura 5.5: Segundo prototipo de la pinza, modelo de dos piezas final.

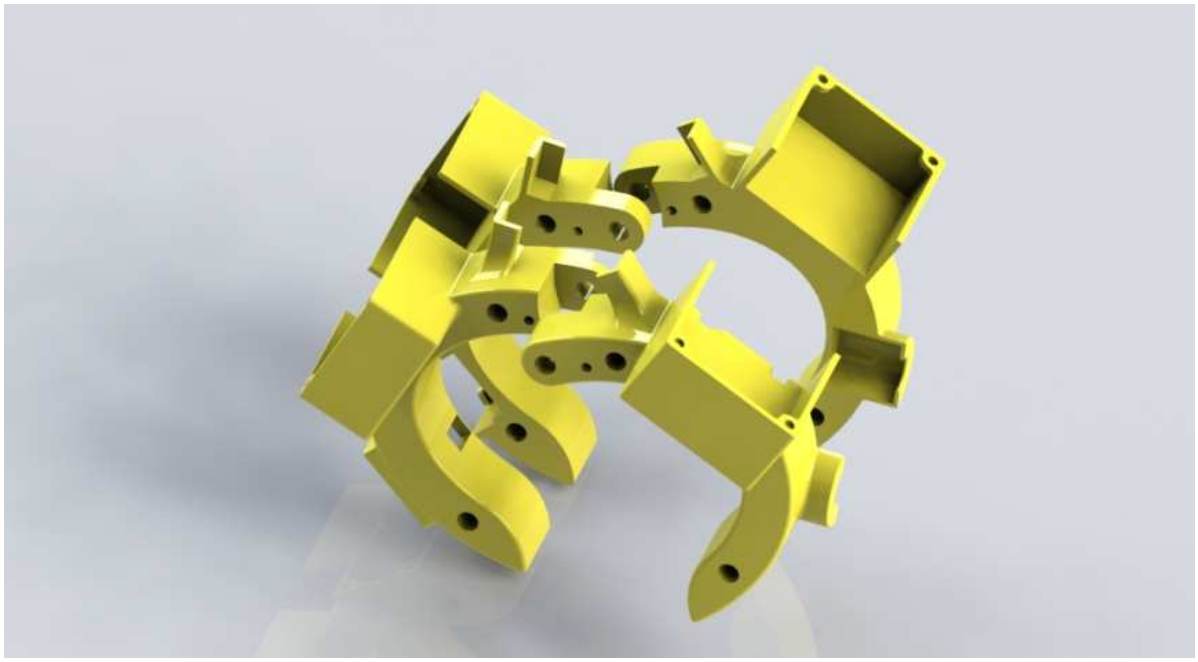


Figura 5.6: División de las partes de la pinza en dos para facilitar el proceso de impresión.

sujete el anillo y no dificulte la visión del indicador luminoso. Se ha practicado una obertura el compartimento del pulsioxímetro para que este se encuentre en contacto con la piel. Además, sobre este sensor, se ha ubicado el acelerómetro.

Como se puede observar en la Figura 5.5, se trata de piezas complejas y difíciles de imprimir debido a que no hay una superficie plana lo bastante grande como para comenzar la impresión desde ella. Por este motivo, se ha dividido cada pieza de la pinza en dos partes tal como se muestra en la Figura 5.6. Se ha aprovechado esta división para tallar unos raíles en el interior de la pieza para el paso de los cables, de forma que queden ocultos dentro del modelo, Figura 5.7. Al igual que con el primer prototipo, el modelo se materializó por medio de una impresora 3D, Figura 5.8.

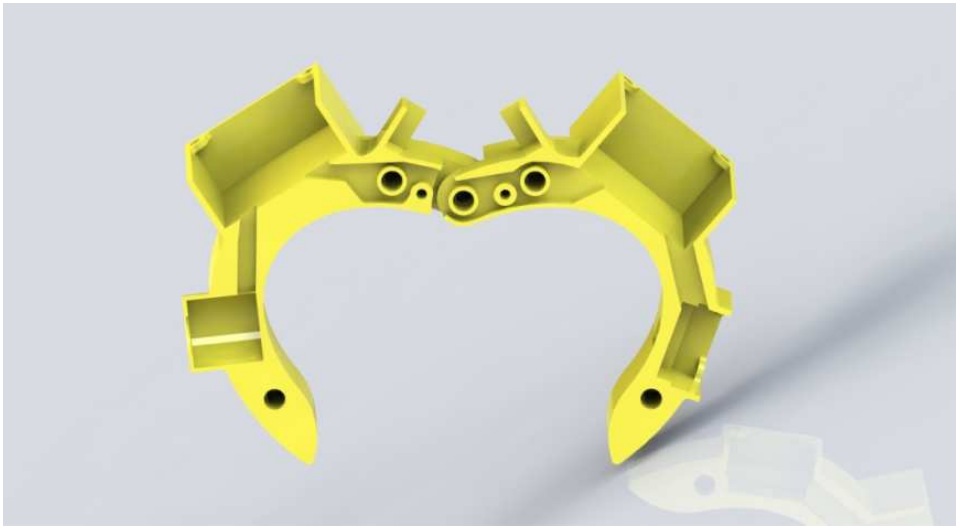


Figura 5.7: Surcos para el paso de los cables.

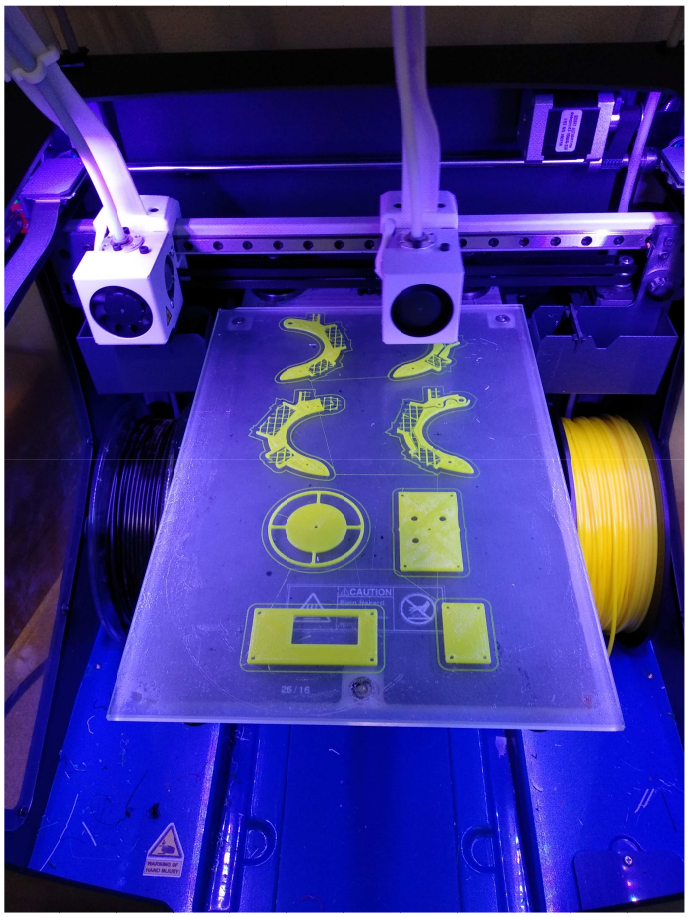


Figura 5.8: Proceso de impresión 3D del segundo prototipo.

6

Experimentos y Conclusiones

6.1 Experimentos

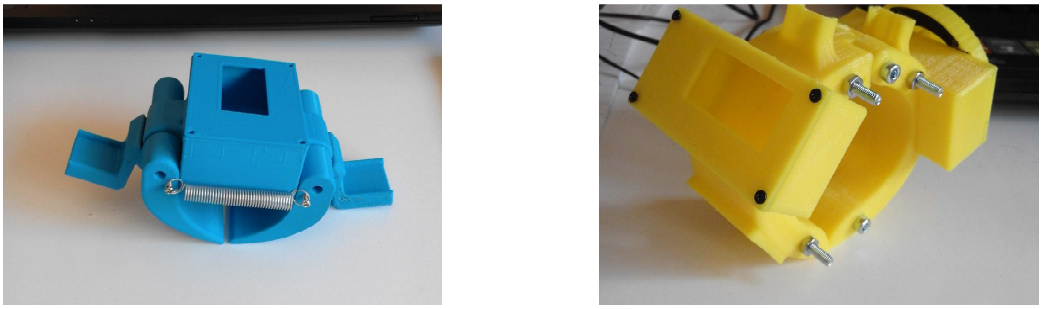
En este capítulo se describen los experimentos realizados y se presentan los resultados obtenidos. En una primera aproximación se montó el circuito sobre una protoboard para determinar el funcionamiento del sensor MAX30102 en distintos puntos del brazo y la mano. Se compararon los valores obtenidos con un oxímetro de pulso recomendado para uso clínico. Los resultados más realistas se obtuvieron en el dedo, seguido de la parte baja de la muñeca, y ,por último, la parte alta de la muñeca. Los resultados obtenidos eran muy fluctuantes aún en presencia del filtro de la media.

Acto seguido, se montaron ambos prototipos y se procedio para comprobar su funcionamiento, Figura 6.1. El segundo prototipo obtuvo mejor desempeño en aspectos mecánicos, siendo el primero demasiado pequeño y más voluminoso. Por estos motivos, se descartó la implementación del sistema electrónico en el primer prototipo, pasando directamente al montaje en el segundo.

Una vez completado el montaje, se procedió a comprobar el funcionamiento del sistema completo. Los resultados obtenidos en la primera prueba fueron incluso mejores que los que se obtuvieron con el montaje en protoboard. En pruebas sucesivas, esta mejoría no se mantenía, Figura 6.2. Esto se debe a la iluminación ambiente, dependiendo de la posición de la pinza en la muñeca los valores se acercan o se alejan de los reales.

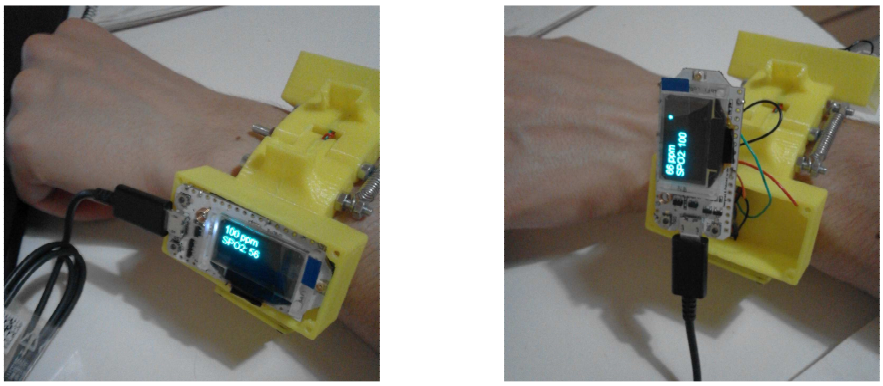
6.2 Conclusiones

En este proyecto se ha hecho un gran aporte a la robótica de rescate. Se ha desarrollado un prototipo de pinza capaz de medir la frecuencia cardíaca y la saturación de oxígeno en sangre y enviarlo en tiempo real a un servidor MQTT a través de Wi-Fi.



a) b)

Figura 6.1: Montaje del primer prototipo, a), y del segundo, b).



a) b)

Figura 6.2: Experimentos realizados con el sistema implementado en el segundo prototipo. En a) se muestra la primera prueba, en b) se muestra la segunda.

Se ha diseñado el sistema a nivel electrónico, mecánico y de software, siendo plenamente funcional los diseños mecánico y electrónico. El diseño software no ha sido completamente funcional por las limitaciones que introducen los sensores, más concretamente el pulsioxímetro MAX30102.

Aunque el sistema no proporciona medidas fiables, se ha demostrado que es posible la implementación de un sistema de monitorización en tiempo real con un tamaño reducido que facilite la toma de decisiones ante una situación de desastre.

6.3 Trabajo Futuro

Dado que el propósito de este proyecto es muy amplio, se proponen varias líneas de investigación futuras que mejoren y amplíen el desempeño de la pinza. Así pues, se puede mejorar el proyecto en cuanto a las comunicaciones, la información recogida de la víctima y el diseño de la pinza.

En cuanto al tema de las comunicaciones, se ha usado conexión mediante Wi-Fi, que presenta un alcance limitado. Se propone como línea de investigación futura la integración de nuevas tecnologías en el campo de las comunicaciones inalámbricas, como es el caso de LoRa. De esta forma, el alcance se vería aumentado considerablemente, pasando de un alcance de varios metros, que proporciona la tecnología Wi-Fi, a un alcance de varios kilómetros, que proporciona la tecnología LoRa. También sería interesante la programación de una interfaz de usuario más elaborada, con su correspondiente aumento de la funcionalidad de la misma.

En cuanto a la información recogida, se podría mejorar el pulsioxímetro del sistema e incorporar sistemas de geolocalización GPS para conocer la ubicación aproximada de las víctimas y trazar planes de acción más detallados, evitando riesgos innecesarios. Sería interesante añadir también sensores para medir la temperatura corporal e incluso un sistema que permitiese hablar con la víctima. Con la incorporación de los nuevos sensores, el algoritmo de triage contaría con más información, lo que conlleva una clasificación más fiable.

En cuanto al diseño, se podría hacer de un material elástico que se amoldara a la forma de la muñeca, evitando que la pinza se desprenda y tomando medidas más fiables. Se podría hacer de un material resistente a golpes que asegurara el funcionamiento del sistema aún en presencia de entornos hostiles. También se podría hacer accesible el puerto de carga de la batería y los pulsadores del ESP32.

Referencias

[1] DPI2015-65186-R. Human wrist localization. 2018.

[2] Fernando Vidal Verdú. Instrumentación Electrónica. Tema 4: Sensores Digitales. pages 125–140.

[3] David Fernando Pozo Espín. Diseño y construcción de una plataforma didáctica para medir ángulos de inclinación usando sensores inerciales como acelerómetro y giroscopio.

[4] I2C IIC SPI MPU9250 sensor módulo MPU 9250 MPU 9250 9 actitud Gyro acelerador + módulo del sensor del magnetómetro MPU9250 en Circuitos integrados de Componentes y sistemas electrónicos en AliExpress.com | Alibaba Group.

[5] DIYmall GY-MAX30102 Heart Rate Click Sensor Breakout Sensors Module for Arduino | eBay.

[6] WIFI LoRa 32 - HelTec Automation.

[7] W. Navarre (Spain). Departamento de Salud., M. Gómez Muñoz, E. Bragulat, and A. Álvarez. *Anales del sistema sanitario de Navarra.*, volume 33. Gobierno de Navarra, Departamento de Salud, 2010.

[8] Tatiana Cuartas Álvarez, Rafael Castro Delgado, and Pedro Arcos González. Aplicabilidad de los sistemas de triaje prehospitalarios en los incidentes con multiples victimas: De la teora a la practica. *Emergencias*, 26(2):147–154, 2014.

[9] J. Gómez Jiménez. Clasificación de pacientes en los servicios de urgencias y emergencias: Hacia un modelo de triaje estructurado de urgencias y emergencias. *emergencias Correspondencia: J. Gómez Jiménez*, 15:165–174, 2003.

[10] E E Pesqueira. Protocolo de campo para el coordinador sanitario de accidentes de múltiples víctimas.

[11] H. Leonardo Ristori. Respuesta prehospitalaria al evento con múltiples víctimas. *Revista Médica Clínica Las Condes*, 22(5):556–565, 9 2011.

[12] Joao Ricardo Borges dos Santos, Gabriel Blard, Arnaldo Silva Rodrigues Oliveira, and Nuno Borges de Carvalho. Wireless Sensor Tag and Network for Improved Clinical Triage. In *2015 Euromicro Conference on Digital System Design*, pages 399–406. IEEE, 8 2015.

[13] David Rodriguez, Stephan Heuer, Alexandre Guerra, Wilhelm Stork, Benedikt Weber, and Markus Eichler. Towards automatic sensor-based triage for individual remote monitoring during mass casualty incidents. In *2014 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 544–551. IEEE, 11 2014.

[14] O. H. Salman, A. A. Zaidan, B. B. Zaidan, Naserkalid, and M. Hashim. Novel Methodology for Triage and Prioritizing Using "Big Data" Patients with Chronic Heart Diseases Through Telemedicine Environmental. *International Journal of Information Technology & Decision Making*, 16(05):1211–1245, 9 2017.

[15] Clinical decision support system based triage decision making. 11 2014.

[16] D. Lo Presti, C. Massaroni, D. Formica, F. Giurazza, E. Schena, P. Saccomandi, M. A. Caponero, and M. Muto. Respiratory and cardiac rates monitoring during MR examination by a sensorized smart textile. In *2017 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pages 1–6. IEEE, 5 2017.

[17] San Jose. MPU9250 Datasheet. *Product Specification*, 1(408):1–4, 2016.

[18] Method and apparatus for filtering electrocardiogram (ECG) signals to remove bad cycle information and for use of physiologic signals determined from said filtered ECG signals. 3 1999.

[19] Sonnia M López Silva, M. L. Dotor, J. P. Silveira, and Luis Antonio Herrera. Fotopletismografía por reflexión con LEDs infrarrojos para evaluar órganos y tejidos intra-abdominales: estudio inicial en cerdos.

[20] Maxim Integrated. MAX 30102 DataSheet.

[21] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions.

[22] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. Technical report, 6 1999.

[23] Victoria Pimentel and Bradford G. Nickerson. Communicating and Displaying Real-Time Data with WebSocket. *IEEE Internet Computing*, 16(4):45–53, 7 2012.

[24] Documentation | MQTT.

[25] Node-RED : About.

[26] JSON.

[27] Sparkfun. SparkFun MAX301x Library.

[28] Los tres teoremas: Fourier -Nyquist -Shannon.