UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALE

DISEÑO Y DESARROLLO DE UN SISTEMA DE MEDICIÓN DE LA TEMPERATURA Y HUMEDAD DE UNA SALA

AUTOR: ÁNGELES RUBIO CUENCA

TUTOR: LUIS JOSÉ SAIZ ADALID

COTUTOR: JOAQUÍN GRACIA MORÁN

Curso Académico: 2020-21

Abstract

The work carried out in this project will be the design and development of a system able to measure the temperature and humidity of a room. To achieve these measures, the necessary electronics will be used in a motherboard with Arduino microcontroller, together with several components and sensors. They will allow us to interact with the environment and to have certain control over it. Likewise, the software part for the correct functioning of the system will be developed.

The purpose of the system is to maintain the temperature and humidity of the room between two values. To do this, measurements will be shown through a screen and, in addition, some actuators will be activated when the temperature and humidity values exceed or fall below the established limits. In case the temperature increases too much, the room will be locked, allowing the access only to authorized personnel with a key.

Key words: Arduino, programming, digital controller, measurement, electronics, assembly, sensor.

Resum

En aquest projecte es durà a terme el disseny i desenvolupament d'un sistema capaç de mesurar la temperatura i humitat d'una sala. Per això, s'utilitzarà l'electrònica necessària i una placa base amb microcontrolador Arduino, a més de diversos components i sensors, que ens permetran interactuar amb l'entorn i tindre'n un determinat control. Així mateix, es desenvoluparà la programació necessària perquè el sistema funcione correctament. La finalitat del sistema és mantidre la temperatura i la humitat de la sala entre uns valors, inferior i superior. Per això, els valors llegits pel sensor corresponent es mostraran per mitjà d'una pantalla. A més, s'activaran els dispositius necessaris els quals avisaran quan els valors de temperatura i humitat sobrepassen o es queden per davall dels límits establerts. En el cas que la temperatura n'arribara a uns valors excessius, l'accés a la sala des de l'exterior quedaria bloquejat, permetent-hi únicament l'entrada a personal autoritzat amb una clau per al seu accés.

Paraules clau: Arduino, programació, control, mesurament, electrònica, acoblament, sensor

Resumen

En este proyecto se realizará el diseño y desarrollo de un sistema capaz de medir la temperatura y humedad de una sala. Para ello, se utilizará la electrónica necesaria junto con una placa base con microcontrolador Arduino, además de diversos componentes y sensores, que nos permitirán interactuar con el entorno y tener un determinado control sobre él. Asimismo, se desarrollará la programación necesaria para el correcto funcionamiento del sistema.

La finalidad del sistema es la de mantener la temperatura y humedad de dicha sala entre unos valores, inferior y superior. Para ello, los valores leídos del sensor correspondiente se mostrarán a través de una pantalla. Además, se activarán los dispositivos necesarios que avisarán cuando los valores de temperatura y humedad sobrepasen o queden por debajo de los límites establecidos. En el caso de que la temperatura tomara unos valores excesivos, el acceso a la sala desde el exterior quedaría bloqueado, permitiendo el paso únicamente a personal autorizado con una llave para su acceso.

Palabras Clave: Arduino, programación, control, medición, electrónica, ensamblaje, sensor.

Índice general

Índice de figuras

Índice de tablas

Parte I
Memoria

CAPÍTULO 1. INTRODUCCIÓN GENERAL Y OBJETIVOS

Previo a la exposición del desarrollo del trabajo y conclusión de este, se realizará una introducción, dando una visión global de los antecedentes, la motivación que ha llevado a su desarrollo, y justificación, así como de los objetivos a lograr en el mismo.

1.1. Objetivos

El objetivo final del proyecto es el desarrollo de un prototipo capaz de medir continuamente la temperatura y la humedad relativa de una sala, y mostrar esta información a través de una pantalla. Dependiendo de los datos obtenidos, se generarán una serie de señales que avisarán en caso de funcionamiento anormal, evitando así daños mayores. Este prototipo, a su vez, incorporará un control de acceso para usuarios y personas autorizadas, dejando la sala inaccesible desde el exterior para personal no autorizado en el caso de que se de una situación de mal funcionamiento. Esto se controlará estableciendo unos límites mínimo y máximo para ambas variables medidas, las cuales deberán de mantenerse dentro de dicho rango de funcionamiento.

En este caso se ha enfocado el proyecto a su aplicación en una sauna húmeda, también denominada baño turco. En resumen, la utilidad de este dispositivo es la de medir e informar de las condiciones ambientales de la sauna, detectar anomalías en los valores medidos y avisar de esta situación, además de controlar el acceso a esta.

Dentro de este objetivo, se pueden identificar otros subobjetivos, los cuales serán necesarios alcanzar previamente al objetivo final, ya que, son esenciales para asegurar el correcto desarrollo del proyecto.

Por una parte, se encuentra el montaje del hardware a partir de la placa Arduino y los diferentes sensores y actuadores, para lo cual es necesario conocimientos de electrónica y electricidad, siendo el primer objetivo que se debe conseguir. Una vez ensamblados los componentes, el montaje hardware servirá de apoyo para el desarrollo de la segunda parte, es decir, el desarrollo del software. Para esta segunda parte, se utilizará la interfaz que proporciona Arduino gratuitamente, programa que se descarga de su página oficial y el cual será necesario instalar y configurar correctamente para poder trabajar con la placa base Arduino Mega 2560.

Conforme se vaya avanzando en el desarrollo del código, también será necesario el desarrollo de un entorno de pruebas para ir comprobando el correcto avance del proyecto e ir subsanando errores, tanto de la parte de software como de hardware.

1.2. Introducción al problema. Antecedentes, motivación y justificación

1.2.1. Introducción y antecedentes

Un baño turco, también llamado Hammam en árabe, consiste en un baño de vapor que hidrata, limpia y suaviza el tejido epitelial del cuerpo. Se trata de un espacio que mantiene el vapor en su interior con una humedad relativa del 99%, lo que provoca la clásica “niebla” que caracteriza al baño turco. Además, se mantiene a temperaturas saludables que suelen oscilar entre los 25 y 50ºC, según la altura a la que se realice dicha medición (por ejemplo,

unos 25ºC en el suelo, 40ºC a metro y medio y a 50ºC a la altura de la cabeza). El calor circula a través de las cañerías y radiadores que se suelen situar en las paredes de dichos espacios (Lucía Reyes, 2005). Al exponerse a esas temperaturas y humedad, el cuerpo experimenta diversas reacciones que son beneficiosas. En primer lugar, es la transpiración y apertura de los poros de la piel lo que hace que se eliminen toxinas y se produzca una limpieza profunda de la epidermis. En segundo lugar, se estimula el riego sanguíneo y comienza la regeneración de las células. En tercer lugar, cabe destacar que el sistema cardiovascular también se beneficia de la concentración de calor, ya que el corazón bombea más fuerte para adaptarse y, por tanto, elimina más deprisa los productos de desecho del organismo (Breathnach, 2004).

Es importante considerar que el principal beneficio de los baños de vapor es su uso para eliminar toxinas por medio de la transpiración. El cuerpo se desintoxica de algunos metales pesados y de otras sustancias y elementos como alcohol, nicotina y sodio, y además mejora las condiciones de la piel, pues funciona como método de limpieza en profundidad (SADM-IPD, 2011).

El vapor del baño turco es caliente y húmedo, lo que favorece combatir las dificultades que se producen en las vías respiratorias, como en la nariz, la garganta y los bronquios. Estos, a través del vapor, reciben efectos beneficiosos. Además, se desarrolla una significativa relajación en el sistema nervioso.

El objetivo del presente proyecto es utilizar eficientemente los conceptos, postulados y leyes de la Ingeniería Industrial en el diseño de un dispositivo cuya función principal sea el mantener unos niveles de temperatura y humedad adecuados para su utilización en un baño turco, evitando posibles situaciones en las que las condiciones ambientales de la sauna húmeda alcancen unos valores indeseados. Además, el mismo prototipo será capaz de controlar el acceso de los usuarios a la sauna.

1.2.2. Motivación y justificación

Tras haber cursado todas las asignaturas del Grado de Ingeniería en Tecnologías Industriales, grado en la que se estudian en mayor o menor medida una gran variedad de ramas tecnológicas, he decidido centrar mi Trabajo de Fin de Grado en aquellas que más me han llamado la atención y con las que más he disfrutado. Estas son las relacionadas con informática y electrónica, en concreto la parte de programación, la cual se estudia en las asignaturas Informática 1, en el primer curso, y Tecnología Informática Industrial en el cuarto curso. De esta manera, el proyecto también sirve para ampliar mis conocimientos en este campo.

Por otro lado, también quise aplicarlo a algo cotidiano, que resultara familiar y que supusiera, en cierto modo, una necesidad real que había localizado en mi entorno.

Gran parte de mi vida he sido usuaria de piscinas climatizadas que disponían de sauna finlandesa y/o baño turco, en las que suele haber un medidor analógico de temperatura y humedad, localizado dentro o fuera de la sauna, lo que no facilita el tener un control constante y preciso del clima de esta, y, por lo tanto, resulta difícil identificar cuándo se produce un funcionamiento anormal.

Por otra parte, en varios de los complejos deportivos en los que disponen de sauna, no todos los usuarios están autorizados a usarla, por el modo de suscripción. En este caso, el

control se hace mediante un ticket que se pide previamente en la recepción. Lo cual no es práctico por varias razones: el usuario tiene que acordarse de pedir el ticket antes de entrar a las instalaciones, la mayoría de los usuarios que usan la sauna realizan previamente otro deporte, normalmente natación, y este ticket acaba mojándose o perdiéndose. Además, es necesario personal responsable de este modo de control.

Lo que se pretende conseguir con este TFG es tener un control más preciso del clima de la sauna, saber inmediatamente si se da un mal funcionamiento para así poder actuar en consecuencia y, además, controlar el acceso sin necesidad de personal, de una manera mucho más cómoda para los usuarios.

1.3. Estructura del trabajo

En la escritura de la memoria de este trabajo que se ha llevado a cabo, se han analizado consecutivamente todos los pasos que se han seguido en su desarrollo y se han plasmado sobre el papel, intentando hacerlo de una manera clara y concisa para que cualquier persona sea capaz de entender el trabajo aquí realizado.

Se ha comenzado con un capítulo introductorio (**Capítulo 1**) que proporciona la base para entender el resto del proyecto. En este capítulo, se han establecido los objetivos y la motivación que ha llevado a su desarrollo, es decir, contiene la idea principal sobre la que gira el resto del trabajo y que es necesaria para entender el contenido del resto de los apartados.

El **Capítulo 2** se ha reservado para explicar el proceso de análisis del problema, es decir, desde la identificación de la necesidad y su estudio, hasta la propuesta de soluciones y la elección de la más idónea.

A partir del **Capítulo 3** se comienza con la puesta en práctica de la solución seleccionada, comenzando por los materiales que se van a utilizar, en este caso todos los componentes electrónicos y de conexión, que en su conjunto representan el hardware. También se describe la parte no física del proyecto (software), que también forma parte de los materiales, en este caso intangibles, y la explicación del entorno de pruebas en el que se han realizado los ensayos experimentales necesarios para comprobar la correcta funcionalidad del prototipo y de sus partes.

La explicación del desarrollo del proyecto en sí, es decir, su “construcción”, una vez se sabe qué se quiere hacer y con qué, comienza en el **Capítulo 4**. En este apartado, se diferenciará entre el desarrollo de la parte del hardware y el del software, aunque uno sirve de apoyo para el desarrollo del otro y viceversa, lo que se verá en el **Capítulo 6**, donde se expondrán las pruebas experimentales que se han llevado a cabo para la comprobación del correcto desarrollo del prototipo.

En el **Capítulo 5** se va a entrar en detalle en aquellos problemas que han ido surgiendo y la solución que se ha propuesto, así como el desarrollo de programas y entornos de prueba de apoyo que han sido necesarios para algunas partes específicas del proyecto, pero que, en sí, estas no están incluidas en el proyecto principal.

Para terminar, se cerrará la memoria con el **Capítulo 7**, donde se expondrán las conclusiones tras finalizar el proyecto.

CAPÍTULO 2. ANÁLISIS DEL PROBLEMA

En este capítulo se explican las diferentes fases por las que se ha pasado durante el proceso de análisis del problema, desde que se identifica la necesidad hasta que se encuentra una posible solución. Dicha solución es la desarrollada en este proyecto.

2.1. Identificación y descripción de la necesidad

Como ya se ha introducido en el capítulo 1, la necesidad a satisfacer mediante el desarrollo de este proyecto se identifica de la propia experiencia personal en el uso de este tipo de instalaciones a lo largo de varios años y, a raíz de conversar con usuarios y propietarios y a la observación, se han detectado varios puntos a mejorar que supondrían una gestión más eficiente y cómoda. Las necesidades que se quieren cubrir son las de tener controlada las condiciones ambientales de un baño turco, es decir, la temperatura y la humedad relativa, e identificar cuando se produce un mal funcionamiento de la instalación para poner sobre aviso tanto a los usuarios como al personal responsable. Por otro lado, también se quiere controlar el acceso a dicha estancia para que solo puedan hacer uso de ella determinados usuarios.

Las necesidades a satisfacer se pueden resumir en tres puntos:

- Obtener información actualizada de la temperatura y de la humedad.
- Identificar cuando alguna de estas dos variables se sale del rango óptimo de funcionamiento.
- Controlar el acceso a la sauna.

2.1. Proceso de búsqueda de soluciones y toma de decisiones

Este proceso se ha llevado a cabo mediante el estudio del problema y las diferentes situaciones que pueden derivar de este. Y se han propuesto soluciones acordes con los medios y la tecnología disponible, cedidos por la Escuela Técnica Superior de Ingeniería Informática. Se dispone de un kit con una placa base Arduino Mega 2560 y diferentes componentes de entre los cuales se elegirán aquellos que mejor se ajusten para conseguir los objetivos marcados.

Se comienza por las dos variables que habrá que controlar: la temperatura y la humedad. Se sabe que el baño turco debe de tener unas condiciones específicas de humedad relativa del 99% y de temperatura de entre 25ºC y 50ºC. Es indiscutible que se necesitará un sensor o sensores capaces de medir estas variables, cuya lectura dará información actualizada de las condiciones climáticas en todo momento. En este caso se dispone de un sensor de temperatura/humedad DHT22 y un sensor de temperatura LM35. Teniendo en cuenta los requerimientos necesarios, se considera el DHT22 como opción válida, ya que nos permite obtener información tanto de la temperatura como de la humedad relativa.

Para la visualización de los datos obtenidos se utilizará una pantalla LCD de 16x2, idónea para lo que se necesita.

Para las señales de emergencia visuales y auditivas, es decir, aquellas cuya función será informar de que alguna o ambas variables toman valores fuera del rango óptimo de

funcionamiento, se han seleccionado leds y zumbadores activos.

A la hora de controlar el acceso se estuvo dudando entre un sensor de tecnología NFC o un teclado matricial, decantándose finalmente por el teclado, puesto que, para esta aplicación en concreto, desempeña mejor la función, ya que no requiere de ningún objeto que haya que guardar y se pueda extraviar o dañar, como ocurre con el sensor NFC, que requiere de una llave o tarjeta.

Para poder dar paso a la sala no es suficiente únicamente con el teclado, puesto que este nos permitirá introducir un código válido, pero, para poder abrir la puerta será necesario actuar sobre la cerradura. En este caso se simulará con un servo.

La placa base con microcontrolador Arduino Mega 2560 constituirá el núcleo del prototipo, ya que será la encargada del control de todos los sensores y actuadores, conformando en su conjunto el prototipo que cumplirá con los objetivos marcados.

CAPÍTULO 3. MATERIALES Y MÉTODOS

Definidos los objetivos del proyecto y seleccionados los materiales que conformarán el prototipo, se continua con la descripción de los componentes de las diferentes partes que han constituido proceso de desarrollo e implementación. Se pueden distinguir tres partes diferenciadas.

En primer lugar, el hardware, es decir, la parte física del dispositivo, que es el ensamblado de los componentes electrónicos. En segundo lugar, el software, es decir, el desarrollo del programa que controlará el funcionamiento del dispositivo. En tercer y último lugar, el entorno de pruebas que nos ayudará a comprobar el correcto desarrollo de las dos partes anteriores.

3.1. Hardware

La parte principal del prototipo es la placa base con microcontrolador Arduino Mega 2560, que será la encargada de obtener la información de las entradas, procesarlas y controlar las salidas. Para el desarrollo del proyecto se utilizará la alimentación de 5V que proporciona el ordenador. Con la conexión al ordenador mediante el cable USB, será suficiente para completar todo el desarrollo del proyecto, puesto que las pruebas experimentales se pueden hacer in situ y, al tratarse de un prototipo, será suficiente.

Aunque una vez el dispositivo esté completamente montado y el programa se encuentre finalizado, se podría conectar a una batería externa para que el dispositivo sea completamente autónomo. Una pila alcalina de 9V es una opción válida para alimentar la placa Arduino Mega, ya que su rango de voltaje de alimentación recomendado es de 7 a 12V.

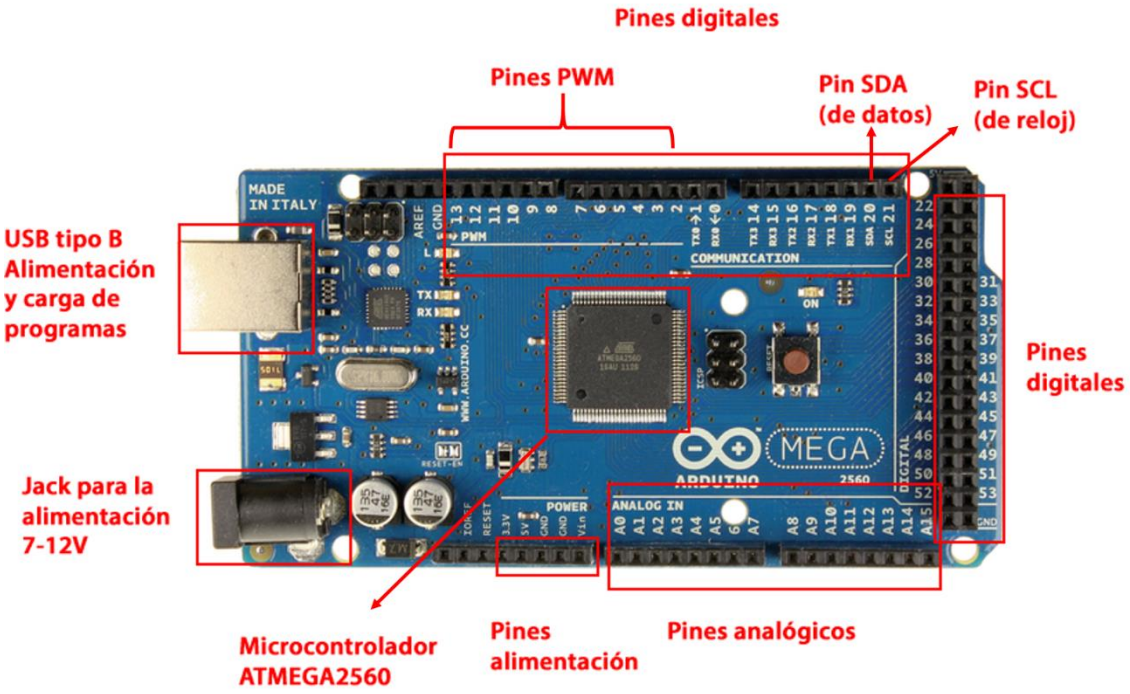


Figura 1. Placa base con microcontrolador Arduino Mega 2560

Para medir la temperatura y humedad se utilizará el sensor digital DHT22, o también llamado AM2302. Este sensor está compuesto por un microprocesador, que utiliza un sensor capacitivo de humedad capaz de medir la humedad relativa del aire. También puede obtener la temperatura ambiente, para lo cual incluye un termistor que mide el aire circundante y calcula la temperatura de este. Este dispositivo obtiene los datos actualizados cada dos segundos enviando la señal a la placa base a través del pin de datos. La ventaja del DHT22 es que el propio dispositivo hace las conversiones a digital de las mediciones de ambos parámetros analógicos.

El sensor que se va a utilizar viene integrado en una PCB, que cuenta con una resistencia pull-up y un condensador de filtrado. La resistencia integrada, R103, de 10 KΩ, está conectada entre el pin de datos y el de alimentación. Esta resistencia no es necesaria para cables de poca longitud. Sin embargo, si se utiliza un cable de más de 20 metros, sí es imprescindible. También hay que tener en cuenta que, si el sensor se alimenta con 3,3V el cable no debe de exceder los 20 metros, ya que las caídas de tensión pueden producir errores en las medidas. En el desarrollo de este prototipo no habrá problema, puesto que se usará la alimentación de 5V y un cable de 2 m de longitud. Si se utilizase un sensor sin PCB con resistencia integrada, sería necesario soldar una entre ambos pines si las circunstancias lo requieren.

Las características técnicas de un DHT22 son:

- Alimentación: 3.3Vdc ≤ Vcc ≤ 6Vdc
- Rango de medición de temperatura: -40°C a +80 °C
- Precisión de medición de temperatura: error de ±0.5 °C

- Resolución Temperatura: 0.1°C
- Rango de medición de humedad: De 0 a 100% RH
- Precisión de medición de humedad: 2% RH
- Resolución Humedad: 0.1%RH
- Tiempo de actualización de datos: 2s

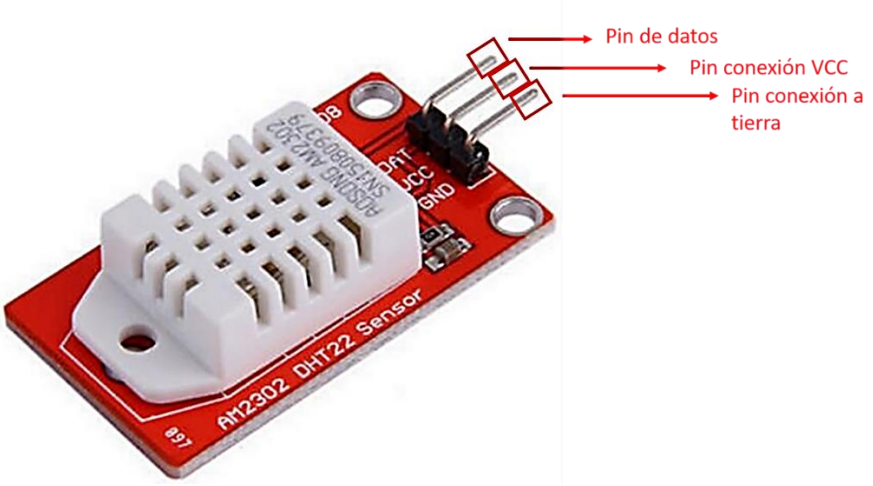


Figura 2. Sensor de temperatura y humedad relativa DHT22

Para el control de acceso al baño turco, se ha usado un teclado matricial 4x4, mediante el cual se introducirán los códigos de los abonados y el código de acceso del personal autorizado. Este teclado está formado por una matriz de pulsadores dispuestos en 4 filas y 4 columnas, de manera que se reduce el número de pines necesarios para su conexión. Para 16 teclas se necesitan 8 pines digitales del microcontrolador, en lugar de 16 pines que serían necesarios para la conexión de 16 teclas independientes. Para poder leer qué tecla ha sido pulsada se debe de utilizar una técnica de barrido y no solo leer un pin del microcontrolador.

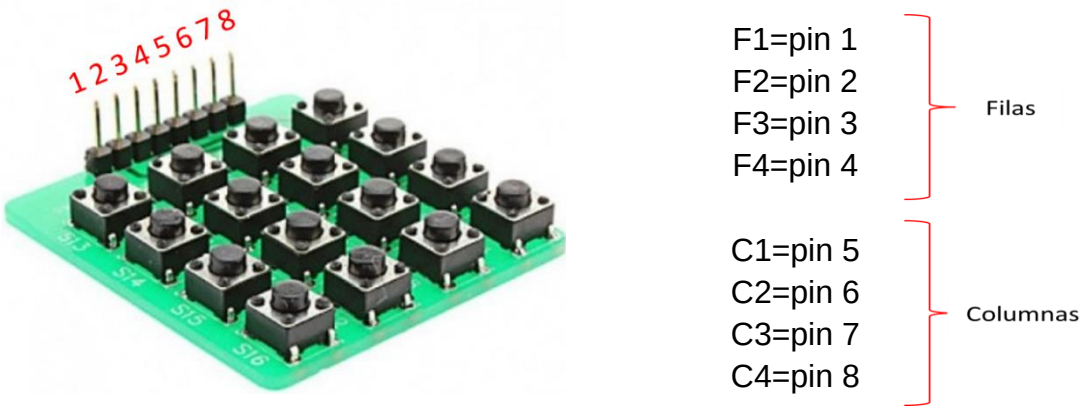


Figura 3. Teclado matricial 4x4

Para la parte mecánica de la apertura desde el exterior se ha utilizado un servo. El servo es un dispositivo que gira, adquiriendo una posición diferente según el pulso que reciba. El funcionamiento mecánico en este prototipo sería el siguiente. El servo se encuentra en la posición de reposo cuando la puerta está cerrada y cuando recibe la orden de abrir, se posiciona en el ángulo de apertura durante unos segundos para poder abrir la puerta desde el exterior, tras los cuales vuelve a la posición inicial. Es decir, actuará de la misma manera que una cerradura eléctrica con retardo. Desde dentro, la apertura de la puerta sería manual y siempre posible, por seguridad.

En el desarrollo de este dispositivo se ha utilizado un Micro servo SG90 Tower Pro de 9 g, pero se podría utilizar cualquier otro servo con la misma programación, simplemente cambiando el servo. Para una aplicación real sería conveniente utilizar uno de mayor par que soportase esfuerzos mayores.

Por último, hay que indicar que entre el cable de alimentación y la toma de tierra del servo se ha conectado un condensador de 100µF para compensar la bajada de voltaje en la placa de pruebas, debido a que el servo consume más corriente en el arranque que cuando se encuentra en movimiento.

Las características del Micro servo SG90 son:

- Peso: 9g
- Voltaje de alimentación: 5V
- Velocidad: 0,1s/60 grados
- Par: 1,8kg
- Rango de temperatura soportada: 0-55ºC

Figura 4. Micro servo 9g SG90 (izquierda) y Condensador de 100µF (derecha)

Para la visualización de la temperatura y humedad relativa de la sauna húmeda, se dispone de una pantalla LCD de 16x2 con módulo I2C. Este módulo permite reducir los pines necesarios para la conexión de la pantalla a la placa, de 16 a 4. La pantalla también reaccionará cuando se introduzca un código de acceso, dando el paso, si este es correcto, o indicando error, si no lo es. Situada en la parte exterior de la estancia, su función es la de servir para la supervisión de las condiciones ambientales del baño turco y para participar en el control de acceso.

La pantalla dispone de cuatro pines que se conectan a la placa Arduino Mega 2560 de la siguiente manera:

- GND: conexión a tierra (GND en la placa)
- VCC: conexión a 5V
- SDA: señal de datos (pin 20)
- SCL: señal de reloj (pin 21)

Figura 5. Pantalla LCD con módulo I2C

También se han utilizado dos LED's rojos y uno verde, cuyas funciones son las siguientes. Los LED's rojos estarán dispuestos de tal manera que uno de ellos quede visible a la parte exterior de la sauna y el otro a la parte interior. Ambos comenzarán a parpadear si alguno de los valores, temperatura y/o humedad, salen del rango óptimo establecido. Además, en caso de que un código erróneo sea introducido a través del teclado, el LED exterior se encenderá durante un segundo y después, se apagará.

Por otro lado, el led verde, situado para que sea visible desde el exterior, únicamente se encenderá cuando el teclado lea un código correcto y, tras un segundo, se apagará.

La conexión de estos diodos luminosos a tierra, es necesaria hacerla a través de una resistencia de 220Ω cada uno, para evitar que se produzca una caída de tensión en los diodos mayor a la soportada por estos. Por último, el ánodo se conectará a un pin digital de la placa. El ánodo siempre irá conectado a la alimentación y el cátodo a tierra, puesto que los diodos no permiten el paso en dirección contraria y, por tanto, no se encenderían.

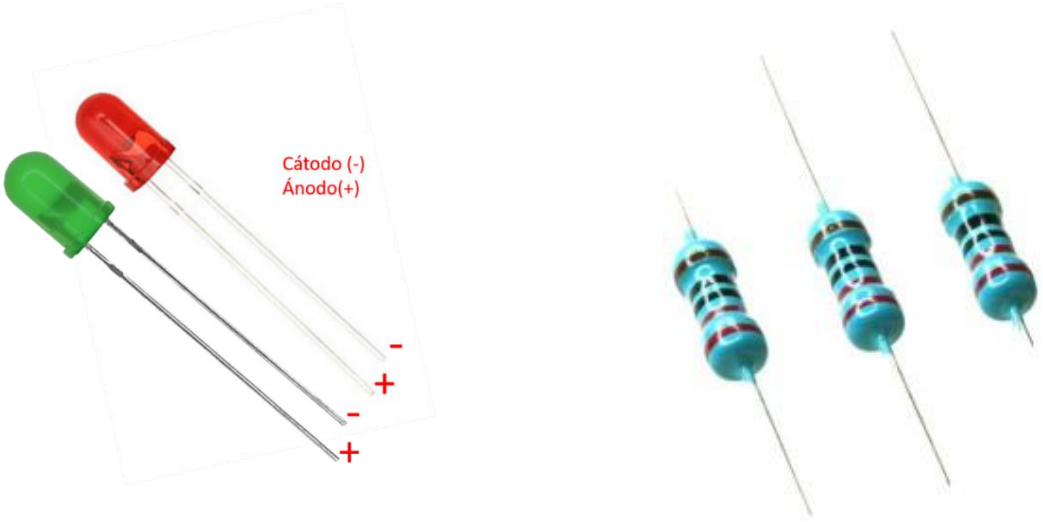


Figura 6. Diodos LED (izquierda). Resistencias 220 Ohm (derecha)

En el caso de que la temperatura sobrepase el valor establecido como límite superior, aparte de las señales luminosas que ya se han expuesto, se activarán unos zumbadores activos. Estos estarán dispuestos de manera que alerten, tanto a los usuarios que se encuentran en ese momento en la estancia, como al personal, que deberá introducir el código de “personal autorizado” para poder apagar los zumbadores.

Figura 7. Zumbador activo

Como en este proyecto se pretende realizar un prototipo, para la conexión de los componentes electrónicos se va a usar una placa de pruebas. Es decir, una caja de plástico con orificios interconectados eléctricamente donde se pueden encajar las patillas de los diferentes componentes. Esta placa permite conectar y desconectar fácilmente los diferentes sensores y actuadores, sin necesidad de soldaduras.

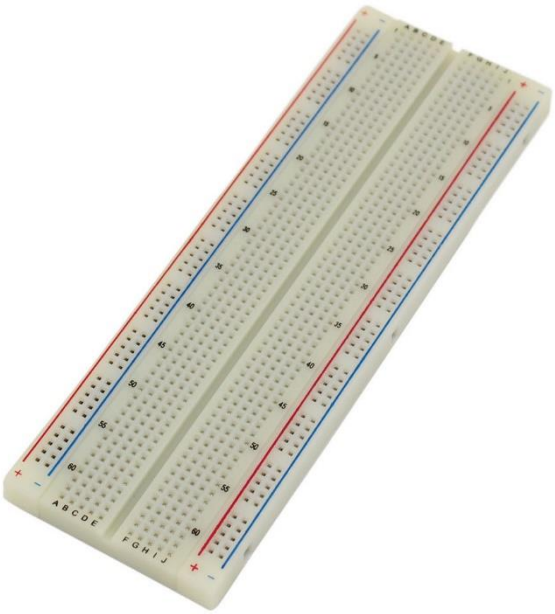


Figura 8. Placa de pruebas

Para la conexión entre sí de los componentes, la placa base y a la placa de pruebas, se han utilizado cables conectores macho-macho, hembra-macho, hembra-hembra y cubiertas de contacto con clavijas, que facilitan el ensamblado de los componentes

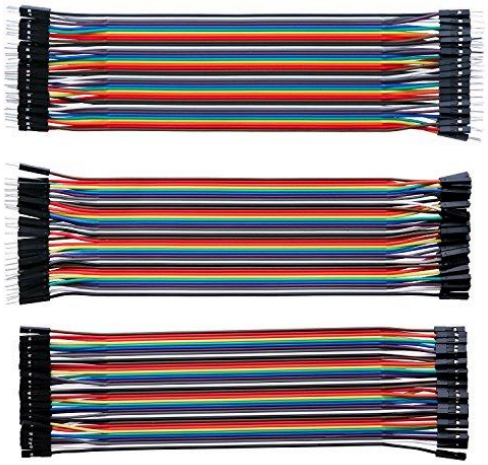


Figura 9. Cables de conexión

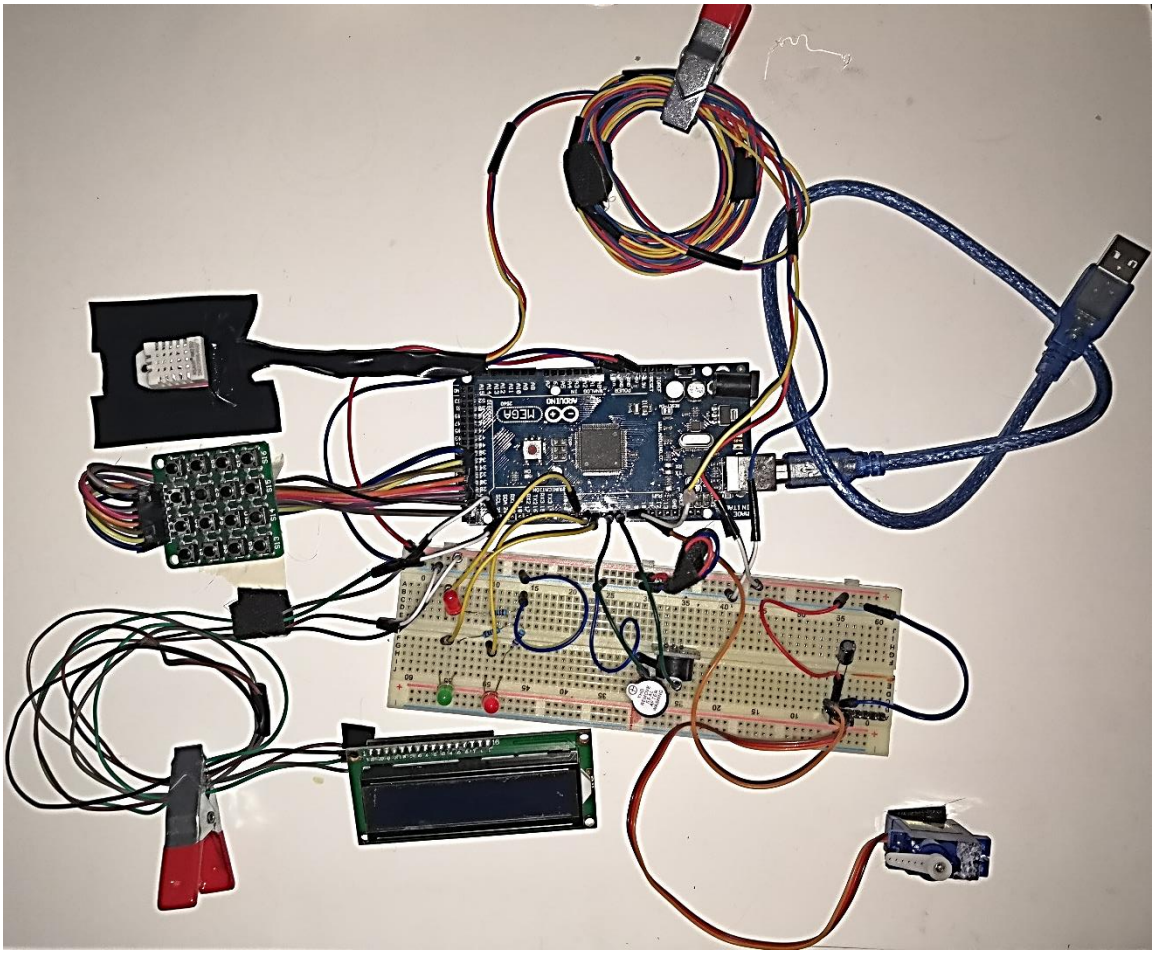


Figura 10. Montaje completo del prototipo

3.2. Software

Como se ha comentado en la introducción del proyecto, para su desarrollo se ha utilizado Arduino. Arduino es una plataforma informática física de código abierto basada en una placa de E/S simple y un entorno de desarrollo que implementa el lenguaje de procesamiento. Arduino se puede utilizar para desarrollar objetos interactivos independientes o se puede conectar al software del ordenador.

Para el desarrollo del código se ha utilizado el IDE (Integrated Development Environment) de software libre que nos ofrece Arduino de manera gratuita. Este entorno de desarrollo integrado (IDE) facilita la escritura de código y su carga en la placa.

3.3. Entorno de pruebas

El entorno de pruebas en este trabajo se puede conseguir de una manera simple. Para verificar el correcto funcionamiento del prototipo que se está desarrollando, tanto a nivel de hardware como de software, ha sido necesario reproducir las condiciones del baño turco en humedad y temperatura, y generar las condiciones de todas las casuísticas posibles y ante las cuales, el dispositivo debe de reaccionar de diferentes maneras. Para conseguir estas condiciones se usará un secador de pelo, el cual servirá para aumentar la temperatura a la vez que para disminuir la humedad. Para aumentar la temperatura y la

humedad se ha utilizado un quemador de aceites esenciales. La vela ira calentando progresivamente el agua de la superficie del quemador generando vapor. En caso de querer obtener una mayor temperatura de la que se puede llegar a alcanzar con el secador sin aumentar la humedad relativa, se ha utilizado el quemador sin agua. Para reducir la temperatura el sensor se aparta de la fuente de calor dejando que se vaya adaptando a la temperatura ambiente.

El quemador no se ha utilizado tal cual, ya que el calor se disipa rápidamente y no se consiguen las temperaturas deseadas. Por ello, se ha creado una estructura capaz de reproducir las condiciones de la sauna. Se han utilizado los materiales enumerados en la siguiente lista y dispuestos como se puede observar en las fotografías.

- Materiales utilizados para realizar los ensayos experimentales con el quemador:
- Quemador de aceites esenciales
- Vela
- Tupperware customizado
- Pequeño bol de plástico
- Bayeta

Además, se ha utilizado un termómetro higrómetro para tener conocimiento de la temperatura y humedad relativa del ambiente, lo que será útil para testear el correcto funcionamiento del DHT22, como se explica en el capítulo 6.

Será necesario soldar un cable a cada patilla del sensor DHT22, para poder manipularlo adecuadamente en el entorno de pruebas, y evitar que se dañen el resto de los componentes.

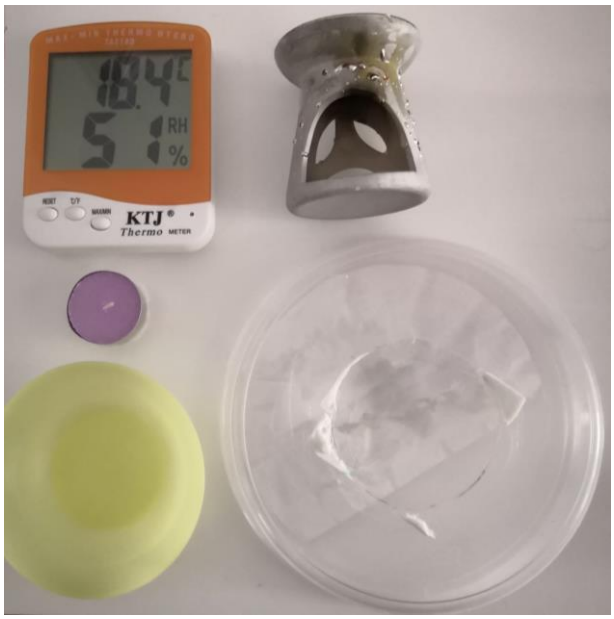


Figura 11. Componentes utilizados para crear un entorno que reproduzca condiciones análogas a la sauna húmeda.

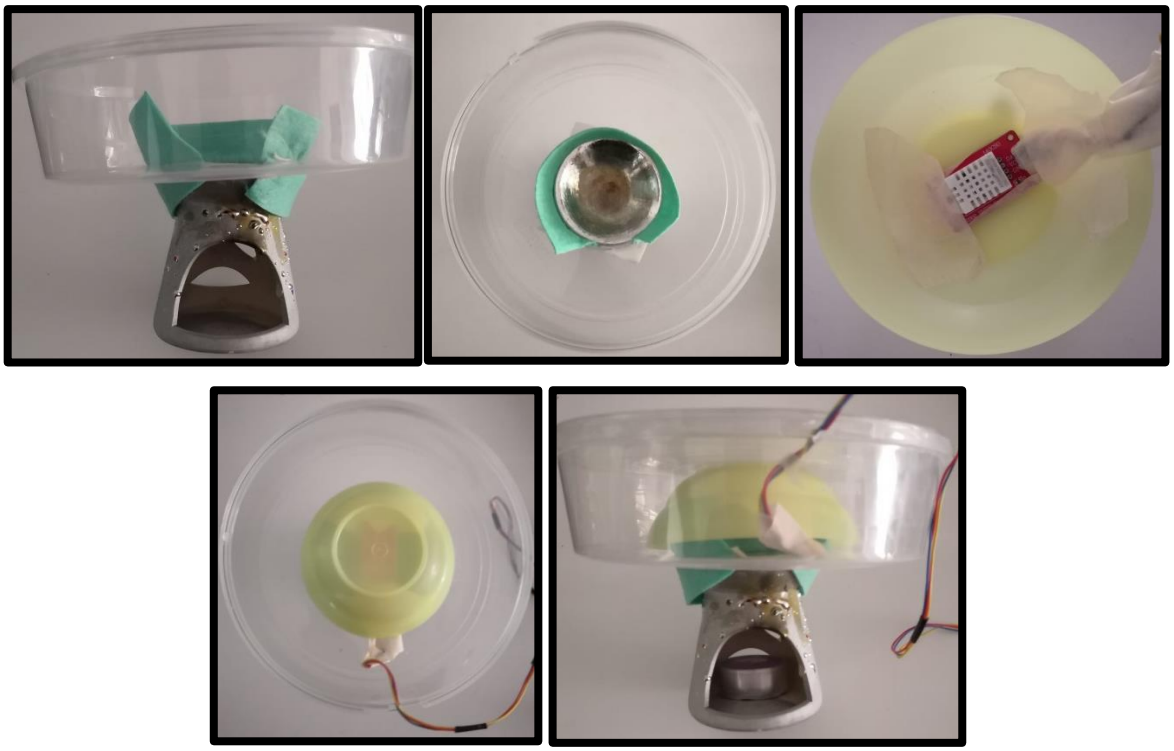


Figura 12. Montaje del entorno de pruebas

CAPÍTULO 4. DESARROLLO DEL PROTOTIPO

Definidos los objetivos, el análisis del problema y los materiales que se han utilizado para desarrollar este prototipo, lo que permite hacerse una idea preliminar para poder entender el proceso que se ha llevado a cabo para el completo desarrollo del proyecto, se procede a la exposición de este.

4.1. Pasos preliminares

Antes de comenzar con el proyecto en sí, se ha tenido que realizar un estudio previo de Arduino, puesto que, aunque la programación es en lenguaje C++, que sí se estudia en el grado, sí que existen algunas variaciones, y al trabajar con el IDE de Arduino por primera vez, ha sido necesario familiarizarse con este antes de comenzar.

Ha sido de gran ayuda “El Libro de Proyectos de Arduino” (Fitzgerald, S., & Shiloh, M., 2014) que viene con el kit de Arduino UNO. Aunque en este caso se haya utilizado la placa Arduino Mega 2560, ha sido de gran utilidad para la introducción a este software libre. Lo que se ha hecho, en primer lugar, ha sido descargar el IDE de Arduino, que se puede obtener gratuitamente de la página oficial www.arduino.cc y configurar adecuadamente el puerto del ordenador que se iba a utilizar, así como el tipo de placa usada. En el momento de comenzar a realizar el proyecto la versión más actual disponible era la 1.8.12 y es la que se ha utilizado.

También ha sido necesario aprender sobre el uso de las librerías, donde encontrarlas si no venían ya instaladas, y cómo instalarlas e incluirlas en el código.

Al tratarse de un software libre podemos encontrar librerías creadas por otros usuarios en https://github.com/arduino-hub. Se trata de una plataforma donde desarrolladores y programadores comparten y desarrollan código.

Una vez está el IDE instalado y se tienen los conocimientos necesarios para empezar a programar, se ha comenzado realizando un esquema general de los componentes necesarios y su función en el sistema para conseguir los objetivos que necesitamos y que se han definido previamente.

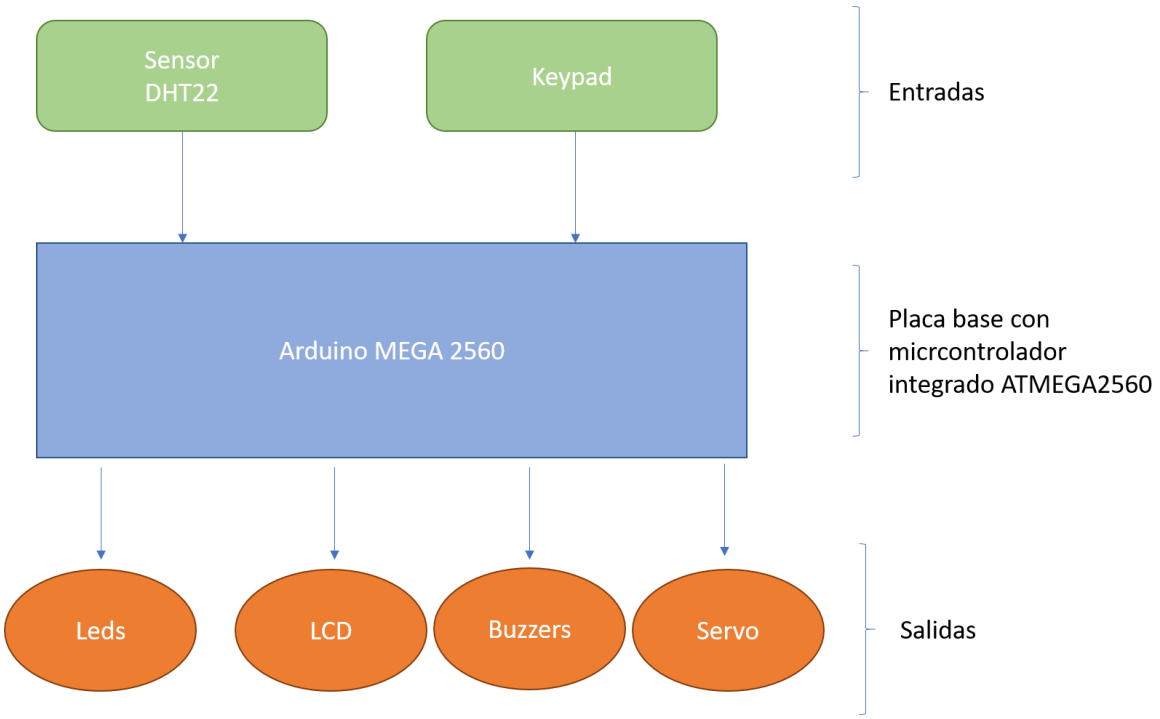


Figura 13. Esquema de entradas y salidas del sistema

Aunque se trata de un esquema simple, este ha sido de gran ayuda, tanto a la hora de conectar los componentes electrónicos a la placa base como para programarlos, ya que nos ofrece de un vistazo el esquema general del sistema.

Como se observa en la figura 13, hay dos entradas digitales, la señal del sensor DHT22, que ofrece una medición de la temperatura y humedad relativa, y por otro lado, está la lectura que se hará al pulsar alguna de las teclas del teclado (keypad). El sistema reacciona dependiendo de las lecturas que haga de estas dos entradas.

Los dispositivos que componen las salidas del sistema son: el servo, los leds, los zumbadores y la pantalla LCD, que se comportarán de diferentes maneras según las lecturas que el sistema haga de las entradas.

4.2. Hardware: ensamblado de los componentes

Tras el estudio del microcontrolador Arduino, la instalación del IDE, la configuración de este para poder comunicarse con la placa y, una vez definidos los materiales que vamos a utilizar, se ha podido comenzar con la electrónica. Esta parte consiste en la conexión de los componentes electrónicos a la placa Arduino Mega 2560 con la ayuda de la placa de pruebas, para no tener que soldarlos y poder manejarlos de una manera sencilla. En este caso, ha sido necesario soldar el sensor DHT22 a un cable de 2 metros para poder manejarlo con facilidad a la hora de realizar las pruebas experimentales. Para la pantalla LCD también ha sido necesario el uso de cables soldados a las patillas, en este caso de 50 cm, ya que no es posible ensamblarla directamente en la placa de pruebas y los cables hembra que trae el kit de Arduino no cumplían su función, puesto que, encajaban con holgura soltándose al mínimo movimiento.

Para el resto de los componentes ha sido suficiente el ensamblarlos directamente o a través de conectores en la placa de pruebas o directamente a la placa Arduino, como en el caso del teclado.

Con los conocimientos de electrónica y circuitos obtenidos en el grado, ha sido suficiente para poder conectar los componentes adecuadamente y acorde con sus características. Es muy importante conocer los datos técnicos de los componentes para evitar dejar inhabilitado algún elemento, debido a una inadecuada conexión o alimentación a un voltaje que no corresponde.

Al no disponer de las características del fabricante, ha sido necesario buscar las propias de cada elemento en diferentes fuentes, que se pueden encontrar en la bibliografía.

Como se puede ver en el esquema, el sensor DHT22, la pantalla y el servo, se conectan al pin de alimentación de 5V de la placa y el resto de los componentes utilizarán pines digitales configurados como entrada o salida. Todos los componentes, excepto el teclado, deberán ir conectados a tierra directamente o a través de una resistencia para compensar la caída de tensión.

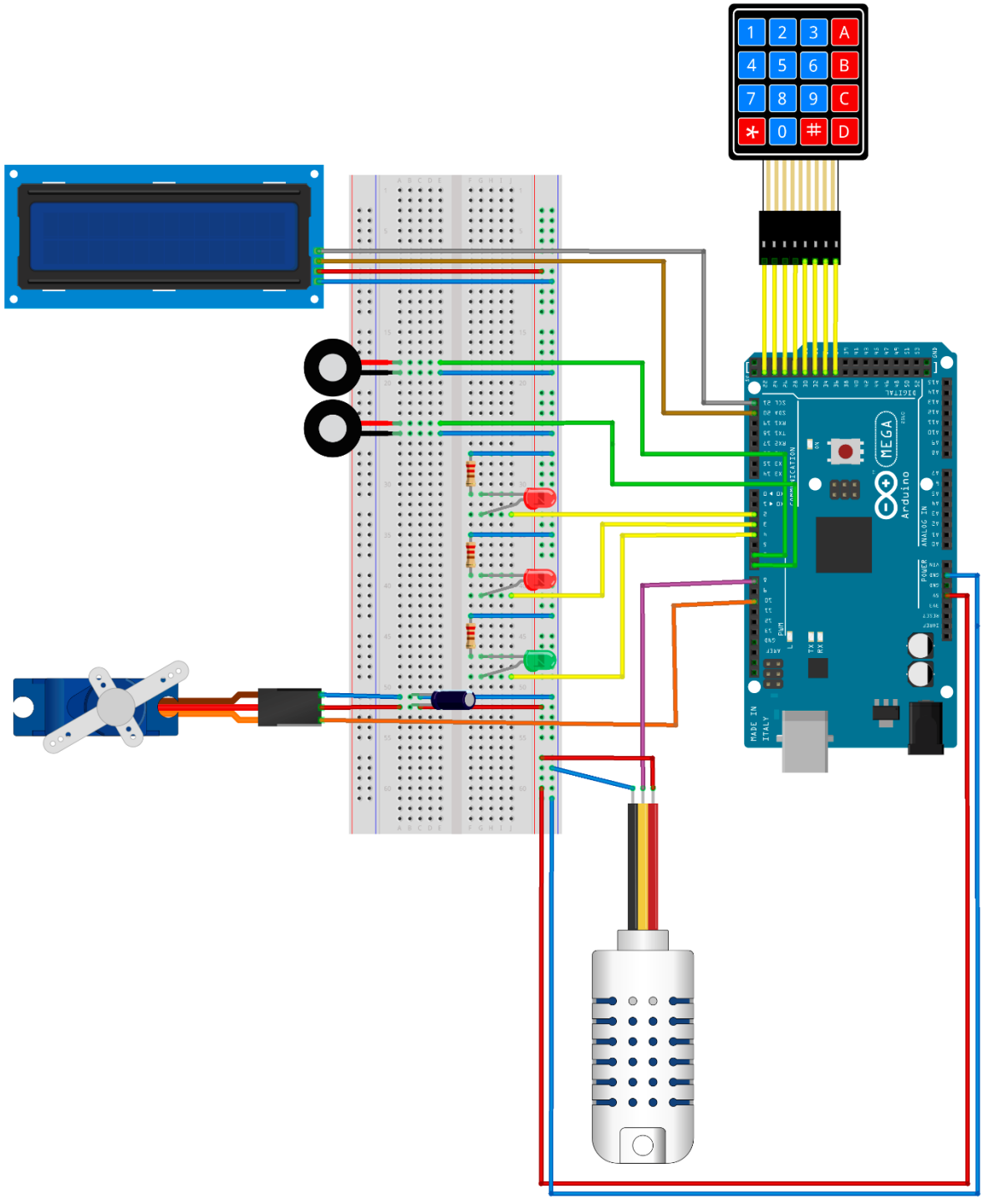


Figura 14. Esquema del hardware del prototipo ensamblado

4.3. Software: desarrollo del código

4.3.1. Estudio de casuísticas

Una vez está el hardware montado y definida la función que tendrá cada componente en el sistema (Tabla 1), se comienzan a estudiar todas las casuísticas posibles para después definir cómo actuará el dispositivo en cada caso. Esto ha sido esencial para desarrollar la estructura del programa. Una vez definidas todas las casuísticas, se ha interpretado gráficamente con un esquema que servirá de guía a la hora de desarrollar el código.

Componente		Función
Entradas	DHT22	Proporciona la medida de la temperatura y la humedad relativa.
	Teclado matricial 4x4	Controla el acceso a la sauna.
Salidas	Leds	Señal visual de aviso por mal funcionamiento. Señalizar el paso a la sauna.
	Servo	Abrir y cerrar puerta.
	Zumbadores	Señal auditiva de aviso por mal funcionamiento.
	LCD	Información de los datos obtenidos con el DHT22. Indica si se introduce un código erróneo o si, al contrario, es válido, de que se puede pasar. Muestra los fallos. Avisa si la sauna está fuera de uso.

Tabla 1. Descripción de las funciones de los sensores y actuadores

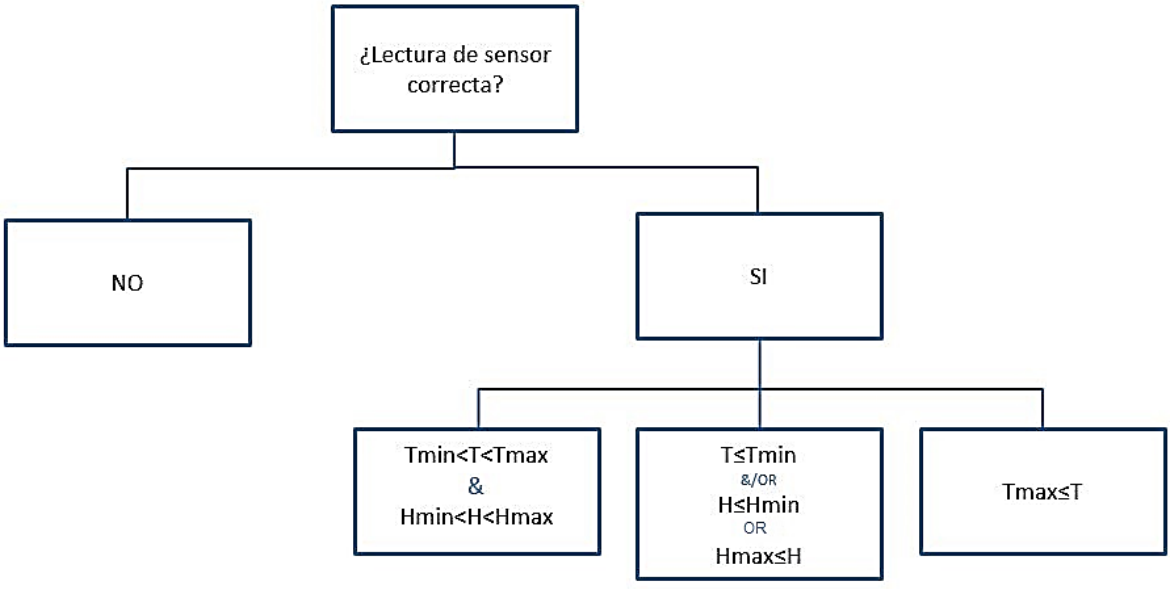


Figura 15. Esquema de casuísticas

Una vez identificadas las situaciones posibles, se define cómo se quiere que actúe el dispositivo en cada caso; lo que se ha resumido en la Tabla 2.

<table>
<tr><th colspan="5">Casuísticas y respectivos comportamientos del prototipo</th></tr>
<tr><td>Error en la lectura del sensor DHT22</td><td colspan="4">- Aviso por pantalla.
- Señales lumínicas.
- Señales sonoras.
- Acceso permitido solo a personal autorizado.</td></tr>
<tr><td rowspan="3">Lectura del sensor DHT22 correcta</td><td>Funcionamiento en condiciones normales. Temperatura y humedad dentro del rango deseado.</td><td colspan="3">- Muestra la información leída del sensor por pantalla.
- Acceso para usuarios habilitado.</td></tr>
<tr><td rowspan="2">Funcionamiento en condiciones anormales.</td><td rowspan="2">En situación la instalación quedaría fuera de uso dejando solo paso a personal autorizado y se activarán determinadas señales de aviso.</td><td>Temperatura por encima del límite superior.</td><td>- Señales de aviso lumínicas y sonoras en el interior y exterior de la sauna.
- Parpadeo de la temperatura en la pantalla LCD.
- Acceso a usuarios deshabilitado.</td></tr>
<tr><td>Temperatura por debajo del límite inferior y/o
humedad por debajo del límite inferior o por encima del límite superior.</td><td>- Señal de aviso lumínica en el interior y exterior de la sauna.
- Parpadeo de la humedad en la pantalla LCD.
- Acceso a usuarios deshabilitado</td></tr>
</table>

Tabla 2. Tabla con los comportamientos del prototipo

4.3.2. Integración de los componentes en el programa y comunicación con el microcontrolador

Una vez se tiene una idea de la estructura general que tendrá el programa, y definido cómo es necesario que actúe en cada situación determinada, es posible comenzar a escribir el código. Lo primero, es identificar qué componentes necesitarán librerías específicas y cuáles no, y los pines de la placa Arduino que se han utilizado para cada uno de ellos.

Algunas librerías están ya instaladas y solo hay que actualizarlas o simplemente importarlas, otras habrá que descargarlas e instalarlas.

Para poder trabajar con el sensor DHT22, se debe instalar la librería DHT.h. Esta librería permite trabajar tanto con el DHT22 como con el DHT11, lo único que hay que cambiar es el segundo parámetro de la función en la llamada.

```
#include <DHT.h>
#define pinDHT22 8   // DHT en pin digital 8
#define tipoDHT DHT22 //Declaración del modelo de sensor de temperatura y humedad

DHT dht(pinDHT22, tipoDHT);
```

Figura 16. Inserción y llamada de la librería DHT

La pantalla LCD viene integrada con un módulo adaptador I2C (Inter-Integrated Circuit), que permite controlar la pantalla a través del bus I2C, usando solo dos pines en lugar de 16. La dirección I2C del controlador, por defecto, puede ser 0x3F o 0x27. Se debe identificar correctamente la dirección I2C del módulo a utilizar para que el programa funcione correctamente. Para identificar la dirección específica de nuestro módulo podemos utilizar el sketch de prueba que se puede encontrar disponible en la página oficial de Arduino (https://playground.arduino.cc/Main/I2cScanner/).

Tanto en el código para identificar la dirección del módulo como en el programa principal del prototipo también es necesaria la librería Wire.h, que contiene las funciones necesarias para el control del hardware integrado. Esta librería permite comunicar la placa Arduino con dispositivos que utilizan el protocolo I2C/TWI. El bus I2C es un sistema de comunicación que utiliza dos líneas de transmisión conectadas a resistencias pull-up a 5V:

- SDA (System Data/línea de datos): pin 20 en Arduino Mega 2560.
- SCL (System Clock/línea de reloj): pin 21 en Arduino Mega 2560.

Una vez está identificada la dirección del módulo I2C, se puede continuar con la escritura del código para incluir la pantalla en el programa. Como se trata de una LCD con comunicación a través del bus I2C, es necesario descargar e instalar la librería LiquidCrystal_I2C en lugar de la librería LiquidCrystal que es la que se utilizaría para la pantalla LCD sin módulo y que ya viene instalada en el IDE.

```
#include <LiquidCrystal_I2C.h>  //importa la libreria LiquidCrystal_I2C para trabajar con la LCD
#include<Wire.h>  //importa la librería Wire que nos permite trabajar con el bus I2C
```

```
//LCD (definición de características)
LiquidCrystal_I2C lcd(0x27,16,2);
```

Figura 17. Fragmento de código correspondiente a las librerías necesarias para trabajar con la pantalla LCD y su configuración

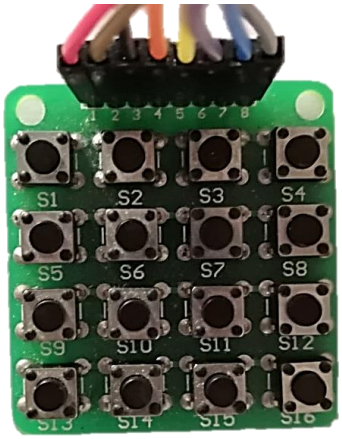
El paso siguiente es configurar el keypad, o teclado matricial 4x4, de forma que quede asignado un carácter a cada tecla. Para el control de este dispositivo es necesario descargar e instalar la librería Keypad.h, que contiene las funciones necesarias para el correcto funcionamiento del teclado de una manera sencilla.

El teclado que se ha utilizado para el desarrollo de este prototipo no dispone de membrana sobre los pulsadores con los caracteres impresos (como, por ejemplo, la mostrada en la figura 18). En el caso de que sí lo tuviese, esto no sería condicionante, pues se tendría que configurar igualmente las teclas, pudiéndole adjudicar a cada una el número, letra o símbolo que se desee. En este caso se ha utilizado la disposición más común en este tipo de teclados, tal y como podemos ver en la imagen siguiente.

Figura 18. Display del teclado

Para el desarrollo de este prototipo se ha podido utilizar el teclado únicamente con los pulsadores, sin la membrana con los números, letras y símbolos. Pero para su aplicación real esta membrana sí sería necesaria.

Puesto que en este prototipo no se tienen los pulsadores identificados de una manera visual, en la figura 19 se puede observar la correspondencia de cada pulsador con el carácter asignado.



S1	S2	S3	S4
1	2	3	A
S5	S6	S7	S8
4	5	6	B
S9	S10	S11	S12
7	8	9	C
S13	S14	S15	S16
*	0	#	D

Figura 19. Asignación de caracteres al teclado

El último componente que requiere de librería es el servo. En este caso, lo único que hay que hacer es incluirla y llamarla en el código, puesto que el IDE viene con esta instalada.

```
#include <Servo.h>      //importa libreria Servo

Servo puertaServo;

puertaServo.attach(10);//Servo en pin digital 10 PWM
```

Figura 20. Fragmentos de código correspondiente a la librería del servo y la identificación del pin de datos.

En el caso de los leds y zumbadores, lo único que hay que hacer es adjudicar el pin correspondiente al que se ha conectado y configurar si ese pin trabajará como salida digital (OUTPUT) o entrada (INPUT). En este caso, todos trabajarán como salida digital, encendiendo o apagando los componentes.

4.3.3. Desarrollo e implementación del código

Una vez los dispositivos que componen el prototipo están conectados y comunicados correctamente con la placa Arduino, el paso que sigue a continuación, es la escritura de la parte del código que controlará los sensores y actuadores y hará que el prototipo funcione tal y como se requiere. En este punto se explicará dicho proceso.

La Tabla 2 ha servido de guía a la hora de avanzar en el desarrollo del código, ya que en ella se consideran y estructuran todos los puntos principales que hay que cubrir en cuanto a las situaciones posibles y el funcionamiento del prototipo.

Las lecturas de la temperatura y humedad relativa son las variables en torno a las que girará el comportamiento del dispositivo, por lo tanto, lo primero que hará el programa será realizar la lectura del sensor. Si se obtiene una lectura errónea, que puede ser debida al deterioro del sensor o un fallo en la conexión, es necesario informar al personal encargado de las instalaciones y prohibir el paso a los usuarios, ya que, no es posible saber qué condiciones climáticas hay en la sauna y por lo tanto identificar un mal funcionamiento. En este caso, se ha decidido utilizar señales visuales a través de leds rojos parpadeando y zumbadores con sonido intermitente. Además, se mostrará un mensaje por pantalla indicando esta situación. El dispositivo bloqueará el acceso a los usuarios desde el exterior, reaccionando únicamente al introducir la clave del personal autorizado.

Un error en el sensor dejará la sauna inutilizada hasta que se solvente el problema, lo que puede llevar horas o días, por lo que las señales sonoras y lumínicas (zumbadores y leds) se apagarán al introducir la clave autorizada y quedará el mensaje en la pantalla y la puerta bloqueada para usuarios, hasta que deje de haber error en la lectura del DHT22.

Figura 21. Mensaje mostrado por pantalla cuando no se puede leer el DHT22

En caso de que la lectura del sensor sea correcta, el programa considerará los diferentes escenarios posibles, definidos en la figura 15 y en la tabla 2. Para cada situación se programan unas instrucciones que ordenan al prototipo actuar de manera que funcione tal y como se requiere.

Para cada escenario se ha utilizado el control de flujo if() y operadores lógicos y de comparación, para definir la condición o las condiciones que se han de cumplir para que se ejecuten determinadas instrucciones.

```
if (t>Tmin && t<Tmax && h>Hmin && h<Hmax)

if (t<=Tmin || h<=Hmin || h>=Hmax)

if (t>=Tmax)
```

Figura 22. Condiciones que consideran las diferentes casuísticas

Para que se considere un funcionamiento normal, es necesario que se cumplan todas las condiciones establecidas, es decir, la temperatura y la humedad deben de estar dentro del rango óptimo de funcionamiento. Si es así, el prototipo, que está continuamente leyendo la información del sensor DHT22, muestra esta información por pantalla y está a la espera de recibir información del teclado. Cuando detecta un cambio en el teclado, por una parte, se inicializa un temporizador de cinco segundos y se comienza a almacenar los dígitos introducidos para compararlos con las claves almacenadas. Tanto si este proceso se lleva a cabo en su totalidad o no, a los cinco segundos el programa se saldrá del bucle y seguirá con su ejecución, quedando otra vez preparado para recibir información del teclado en cualquier momento.

Los códigos de acceso constan de cinco caracteres, el primero es una letra y está reservado para la identificación del tipo de clave (usuario o personal autorizado) y los cuatro siguientes son números y corresponden a la clave de acceso.

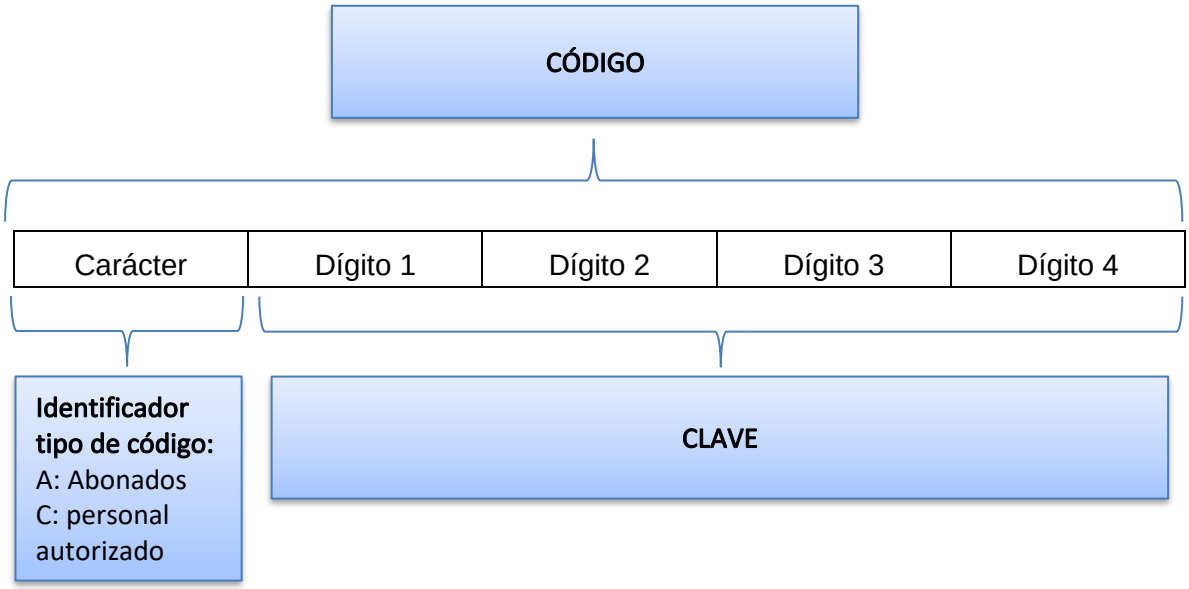


Figura 23. Estructura del código de acceso al baño turco

Todos los códigos de usuarios comienzan con la A y el del personal autorizado con la C. De esta manera podemos, por una parte, hacer una distinción entre ambos códigos y dar instrucciones diferentes de una manera sencilla, y por otra, que el programa no se interrumpa al pulsar cualquier tecla, si no, que hasta que no se pulsa la A o la C no comienza a leer la clave.

Para conseguir que funcione de esta manera, la lectura del teclado se ha configurado con la instrucción switch(),cuyo parámetro es la primera tecla que se pulsa. Como solo se definen dos casos, uno con la A y otro con la C, el programa solo reaccionará cuando se pulse una de ellas. Este tipo de configuración deja abierta la posibilidad de incluir fácilmente otras funciones activables desde el teclado.

Una vez se ha explicado cómo funciona la introducción y lectura de las claves de acceso se pasa a exponer como actúa el prototipo ante esta situación.Si la clave se introduce completamente, se compara con todas las claves almacenadas, y si coincide con alguna se da paso. Por el contrario, si esta no coincide, se deniega el paso.

Para abrir la puerta de la sauna, el servo se mueve desde su posición de reposo, correspondiente a la puerta cerrada, hasta la posición en la que se abre, donde se mantiene durante un segundo y vuelve a la posición de reposo. A su vez, si la clave es correcta, aparece un mensaje en la pantalla dando paso y se enciende el led verde durante un segundo. Si la clave es errónea, se informa a través de un mensaje en la pantalla y se enciende el led rojo situado fuera de la sauna durante un segundo.

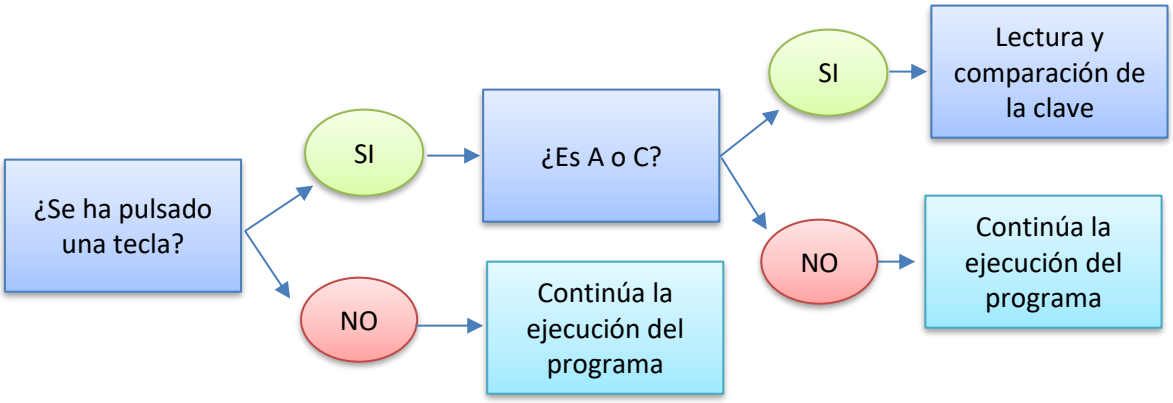


Figura 24. Esquema del funcionamiento de la lectura y verificación de una clave de acceso

Fuera del funcionamiento que se establece como correcto, es decir, cuando, tanto la humedad como la temperatura se encuentran dentro de un rango determinado, se pueden dar diferentes situaciones que hay que tener en cuenta.

Para cada variable controlada existen dos límites que puede sobrepasar, uno inferior y otro superior. Por lo tanto, existen cuatro situaciones diferentes que se pueden producir en un mal funcionamiento, algunas de ellas de manera simultánea, es decir, que ambas variables se encuentren fuera de rango.

Por un lado, se consideran las posibilidades de que la temperatura esté por debajo del límite inferior y/o la humedad esté fuera de rango, tanto por debajo del límite mínimo como superando el límite máximo. Hay que puntualizar que, aunque se considera la posibilidad de que haya un límite superior de humedad, en este caso, la humedad relativa de un baño turco es del 99-100%, por lo que nunca se verá superada, pero como en este proyecto se está desarrollando un prototipo en vistas de que pudiera ser utilizado para otras aplicaciones simplemente cambiando los valores de los rangos de funcionamiento, se ha configurado de manera de que sí se controle este límite.

En el caso de que se cumpla una sola de las condiciones nombradas, el dispositivo cambia de funcionamiento. Se activan las señales de emergencia, en este caso, solo los leds rojos situados en el interior y exterior de la sauna y parpadea en la pantalla LED la variable que se ha salido del rango, de manera que se pueda identificar rápidamente dónde se está detectando el error. En cuanto al acceso, se bloquea el paso a los usuarios desde el exterior y solo se puede abrir con la clave del personal autorizado.

Por último, se va a explicar la condición para la situación en la que la temperatura supere el límite máximo. En esta condición, al igual que en el resto, el programa está preparado para leer una entrada del teclado en cualquier momento, pero, como ya se ha comentado anteriormente, el acceso solo está permitido a personal autorizado.

En el momento en el que se cumple esta condición, se activan las señales de emergencia, que en este caso son: parpadeo de leds rojos, zumbadores sonando intermitentemente, en ambos casos tanto el situado en el exterior de la sauna como el del interior, y en la pantalla se muestra el dato de la temperatura parpadeando y el mensaje de no pasar.

Los zumbadores se activan en una primera instancia para poner en aviso rápidamente tanto a los usuarios como a los responsables de las instalaciones, pero, una vez los responsables se aseguran de que se ha evacuado la sauna, podrá apagar los zumbadores simplemente introduciendo el código del personal autorizado, que a su vez también abre la puerta. El resto de las señales se quedan activas hasta que la temperatura cambia de estado y vuelve al rango de funcionamiento.

CAPÍTULO 5. ANÁLISIS Y SOLUCIONES PROPUESTAS A LA PROBLEMÁTICA SURGIDA DURANTE EL DESARROLLO DEL PROYECTO

5.1. Estudio del rango de funcionamiento del micro servo

El micro servo es el encargado de abrir y cerrar la puerta. Tomará para ello dos posiciones en diferentes ángulos. La posición de reposo será la de la puerta cerrada y solo tomará la posición de abierto cuando reciba la orden, permanecerá un segundo en esa posición y se volverá a tomar la posición de cerrado.

Para la instrucción que determina la posición del servo, se ha utilizado la función writeMicroseconds(),incluida en la librería Servo.h. Este comando define el ancho de pulso que se le quiere enviar al servo. Se trata de una función mucho más precisa que write(), cuyo argumento es en grados y su uso puede dar lugar a un mal funcionamiento. El ancho de pulso mínimo y máximo, que corresponderían al ángulo 0º y el máximo del servo, pueden variar de un servo a otro, incluso entre servos del mismo modelo. Es debido a esto que, aunque el recomendado en la página de referencia del comando attach() en arduino.cc y en la biblioteca Servo.h se establecen como predeterminados los valores 544 y 2.400 µs para el valor mínimo y máximo respectivamente, estos pueden variar. Lo más usual es que los servos tomen la posición 0º en 1000 µs y 180º en 2000 µs, pero en otros, este rango abarca de 500 a 2500 µs. Es muy importante conocer los valores del servo que se está utilizando, ya que si se envía un pulso fuera del rango este se puede estropear. Para conocer los valores del micro servo que se está utilizando hay que definirlos experimentalmente. En este proyecto se podrían probar a coger unos valores intermedios que con seguridad sabríamos que están dentro del rango y ver si el ángulo resultante es válido, pero si no es así, habría que ir probando igualmente. Por eso, se ha decidido que la mejor opción es la de delimitar previamente el rango del micro servo que se va a utilizar. Para ello se ha creado un programa muy sencillo donde se han definido dos variables a las que se les irá cambiando el valor de manera iterativa hasta encontrar el pulso que corresponde a la posición de 0º y el pulso correspondiente a la posición máxima, que serían 180º. Se comienza la iteración utilizando los valores más seguros de 1000 µs y 2000 µs y dependiendo de los resultados se continua con unos valores u otros sin exceder el rango de [500-2500] µs.

```
#include <Servo.h>
Servo pruebaServo;
int i;
int pos_inicial=540; //Pulso límite inferior (µs)
int pos_final=2400; //Pulso límite superior (µs)
void setup() {

  pruebaServo.attach(8);//Servo en pin digital 8 PWM
//Inicilizamos el servo en posición de apertura.
    Serial.begin(9600);

}

void loop() {
pruebaServo.writeMicroseconds(pos_inicial);
delay (5000); //retardo de 3s para identificar la posición
 pruebaServo.writeMicroseconds(pos_final);
 delay(3000);
}
```

Figura 25. Código para prueba experimental con servo

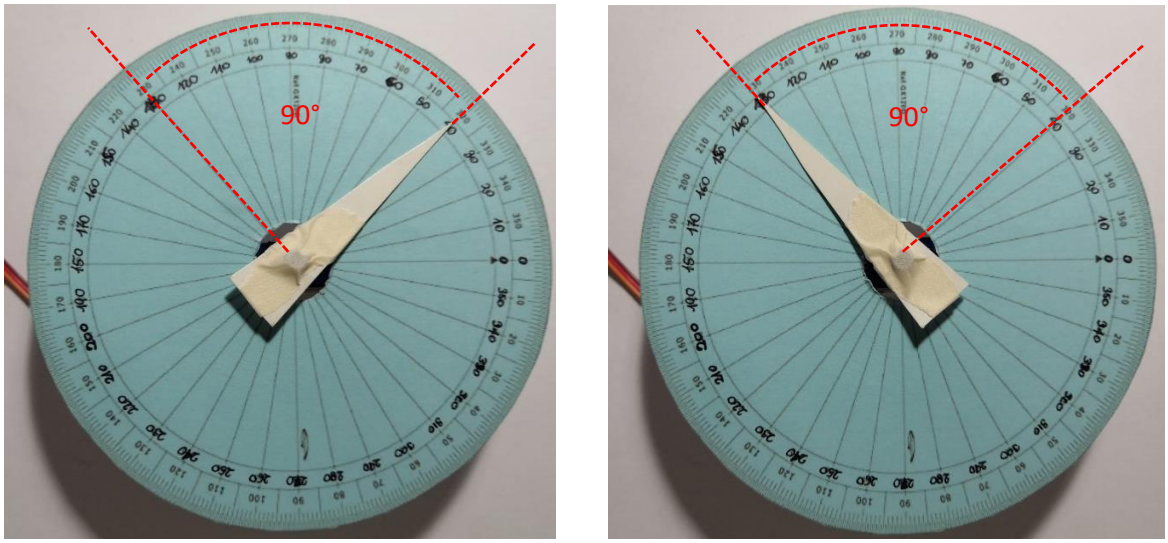


Figura 26. Posición resultante del pulso de 1000 µs de (izq.) y 2000 µs (dcha.)

En esta prueba nos guiamos por la diferencia de grados que hay entre ambas posiciones, que es lo que nos ofrece información, puesto que no se puede decir que un pulso de 1000µs corresponde a un ángulo de 40º y el de 2000µs a 130º ya que no sabemos si se ha colocado de manera que el ángulo 0º en el transportador de ángulos corresponda con la posición inicial del servo.

Se observa que entre la posición correspondiente a cada pulso hay 90º. De esta

información se deduce que entre cada grado hay una diferencia de 11,11 µs y se supone, a partir de los datos teóricos antes mencionados, que 1000 µs corresponden a 45º y, por lo tanto, los 2000 µs a 135º. Esta suposición se ha de comprobar más adelante.

A partir de esta suposición y una simple operación se proponen los pulsos que corresponderían a las posiciones de 0º y 180º:

$$1000\mu s - (45° \times 11{,}11\mu s/°) = 500{,}05\mu s$$

$$2000\mu s + (45° \times 11{,}11\mu s/°) = 2500{,}05\mu s$$

Los datos obtenidos parecen lógicos pero es preferible apostar por la seguridad y seguir iterando dentro de los vapores predeterminados. Por lo que la siguiente prueba se hará con los valores 544 µs y 2400 µs.

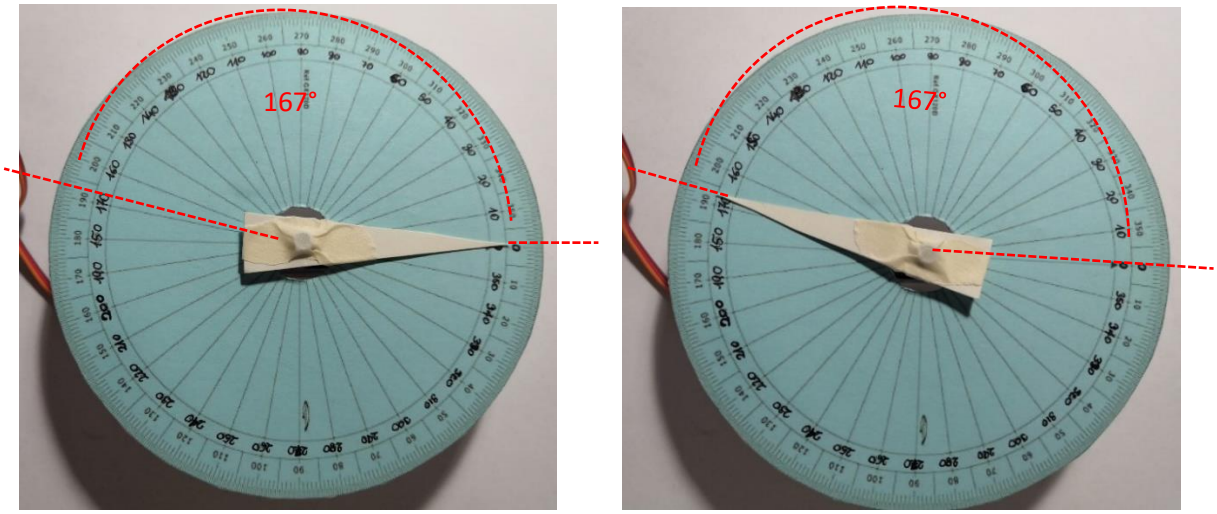


Figura 27. Posición resultante del pulso de 544 µs (izq.) y de 2400 µs (dcha.)

Se obtiene una diferencia de 167º por lo que hay que aumentar el rango. Las pruebas que se han hecho hasta ahora indican que los cálculos hechos son correctos y los valores del servo van de 500 µs a 2500 µs, así que, en el siguiente ensayo se utilizan estos valores, pero los resultados no son los esperados.

Se obtienen las mismas posiciones del servo que en la experiencia anterior, teóricamente debería de tener un recorrido de 180º, pero en este caso lo tiene de 167º aproximadamente. Para comprobar que no se está enviando un pulso mayor al soportado y que el servo alcanza sus límites con pulsos mayor a 544 y 2400 se hace una última prueba con unos valores lijeramente diferentes. Se utilizan 550 µs y 2300 µs, obteniento un rango inferior a 167º, como se observa en la figura 28.

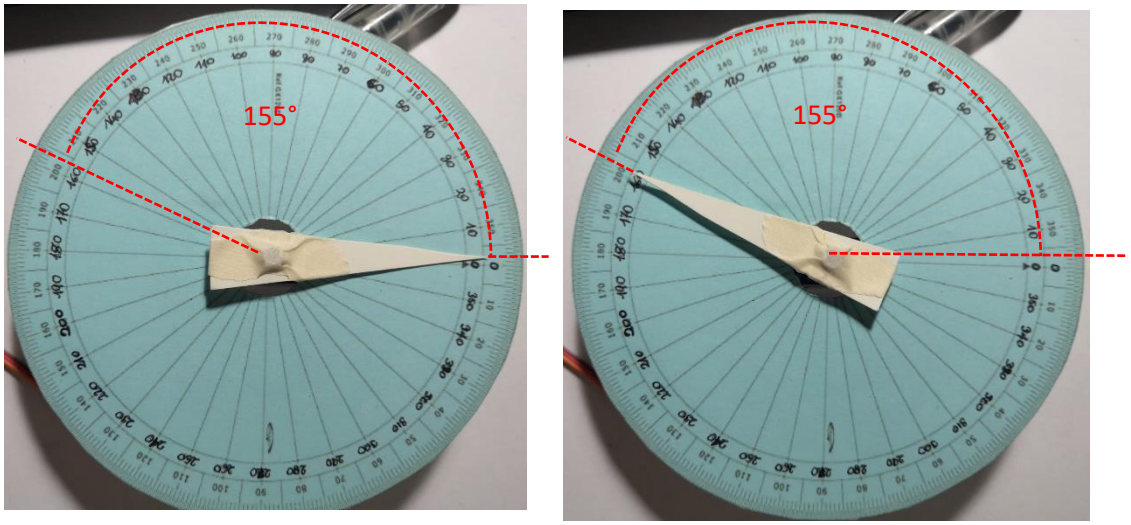


Figura 28. Posición resultante del pulso de 544 µs (izq.) y de 2400 µs (dcha.)

La conclusión a la que se llega analizando los resultados optenidos experimentalmente es que el servo tiene un recorido real de 167º y los pulsos correspondientes a las posiciones inicial y final respectivamente son, aproximadamente, 544 µs y 2400 µs.

Con esta información ya se pueden introducir los datos para el servo en el programa (figura 29) de forma segura y sabiendo el ángulo de apertura con el que este responderá, lo que es necesario a la hora de pasar de prototipo a aplicación real.

```
  //Declaración de variables que contienen las posiciones del servo (puerta abierta o puerta cerrada)
int abierto=550;
int cerrado=2300;
```

Figura 29. Variables que definen la posición del servo

5.2. Función millis()

A lo largo del desarrollo del código, han surgido una serie de problemas a los que se ha tenido que buscar una solución alternativa a la pensada en un principio, ya que estas no han tenido los resultados esperados o no se habían tenido en cuenta ciertas situaciones como las que se describen a continuación. En esta resolución ha representado un papel esencial la función millis(). Esta función devuelve el número de milisegundos desde que la placa Arduino comienza a ejecutar el programa.

En una primera ejecución del código se han detectado los siguientes problemas:

- Al comenzar a introducir una clave por el teclado, si no se completaba el proceso, el programa se quedaba detenido hasta que fuese finalizado. Esto supone un problema grave, ya que se deja de obtener información del sensor y no se detectarían las variaciones de temperatura y humedad, dejando inhabilitado el sistema de seguridad.

- Para el parpadeo de los leds y la intermitencia de los zumbadores se había utilizado la función delay(). Esta función paraliza la ejecución del código durante el tiempo que se le asigna al parámetro de entrada, impidiendo detectar cualquier entrada durante este periodo, por lo que es prácticamente imposible introducir un código con el teclado y que sea detectado.

Ambas situaciones se han solucionado utilizando la función millis con un procedimiento con una base similar, pero obteniendo diferentes resultados, adaptados a lo requerido en cada situación.

5.2.1. Temporizador

En el primer caso, lo que se necesita es que, si se comienza a introducir una clave y no se terminan de teclear todos los dígitos en un tiempo determinado, se salga del bucle y el programa continúe ejecutándose. Esto se consigue creando un temporizador con la función millis(). Para ello se declara una variable que guarde el momento exacto en el que se pulsa la primera tecla, otra que vaya tomando el valor del tiempo transcurrido dentro del bucle y el tiempo de espera que se quiera. Y se hace una comparación cada vez que se ejecuta el bucle. Cuando la resta de ambas variables sea mayor que el tiempo de espera establecido, el programa saldrá del bucle que lee y almacena las entradas del teclado y seguirá su curso. Además, para que funcione tal y como se ha explicado, no solo se ha tenido que incluir el temporizador, sino que también se ha cambiado la manera en la que se hacía la lectura del teclado, ya que se había utilizado el comando waitForKey(), el cual, como su nombre indica, espera hasta que se pulsa una tecla. Esto hacía que, aunque terminara la cuenta atrás del temporizador, el programa se quedase paralizado en esta instrucción. Para solucionar este inconveniente se ha utilizado el comando getKey(), que no paraliza la lectura del código esperando la señal. Además de intercambiar estas funciones, se han tenido que hacer otras modificaciones para que el programa haga exactamente lo mismo, pero sin quedarse en espera. Antes se había usado un bucle while(), con condición de ejecutarse mientras n sea menor a cuatro (while(n<4)), para almacenar los cuatro dígitos de la clave, de manera de que el bucle solo se ejecutaba cuatro veces, porque el código solo avanzaba al pulsar una tecla, y una vez se detectaban las cuatro pulsaciones el programa se salía del bucle. Usando getKey(), se hace de forma similar pero hay que añadir una condición para que en cada ejecución del bucle la n no aumente en una unidad provocando la salida del bucle antes de introducir la clave completa. Esta condición es que la n no aumenta si no se pulsa una tecla. De esta manera el while() se ejecutará continuamente aunque no se pulsen teclas, y cuando transcurra el tiempo establecido de espera, el programa se saldrá del bucle. En los fragmentos de código de la figura 30 y el esquema de la figura 31 se puede apreciar la diferencia entre ambos códigos y el funcionamiento del temporizador.

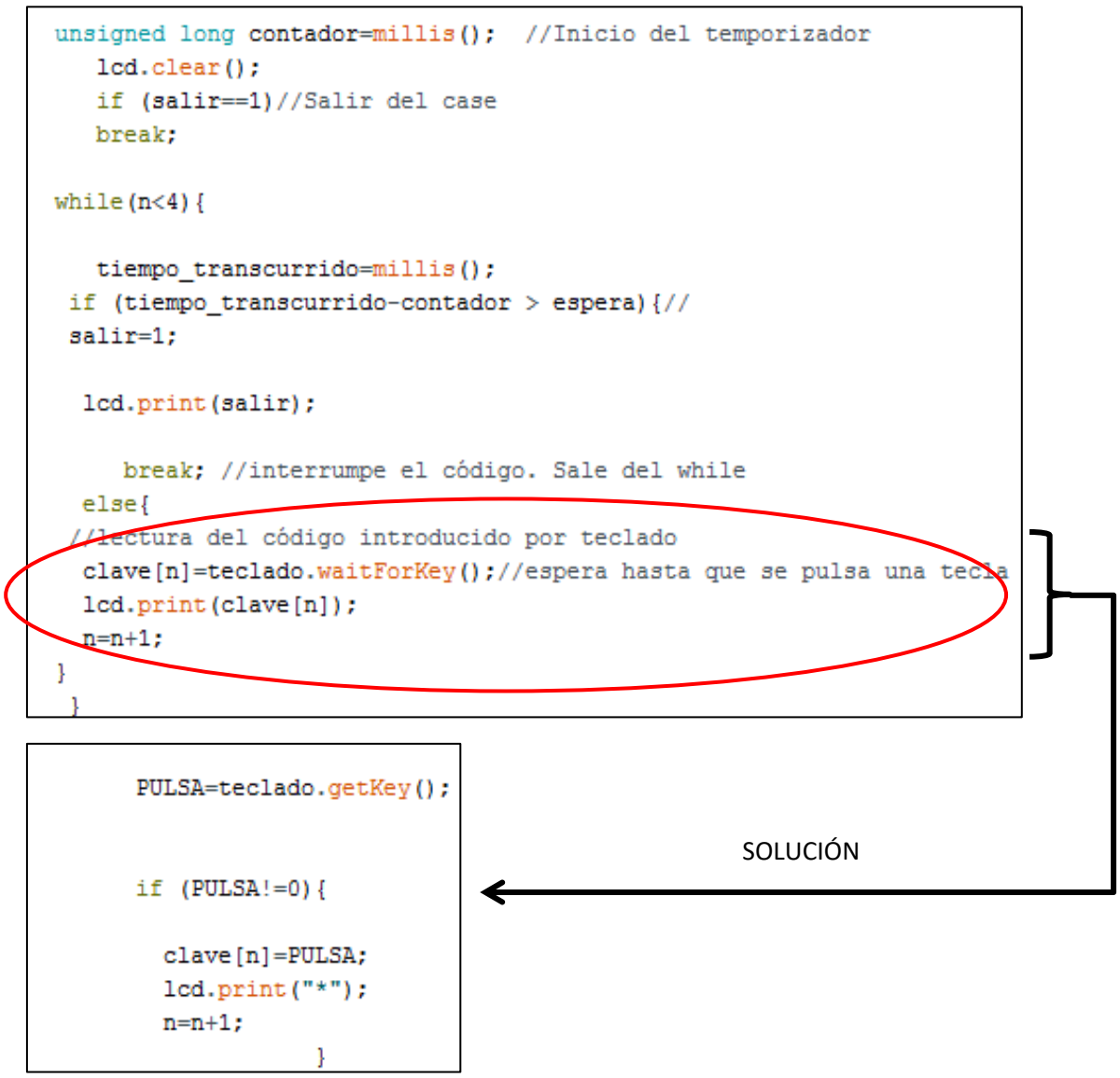


Figura 30. Solución para el correcto funcionamiento del temporizador

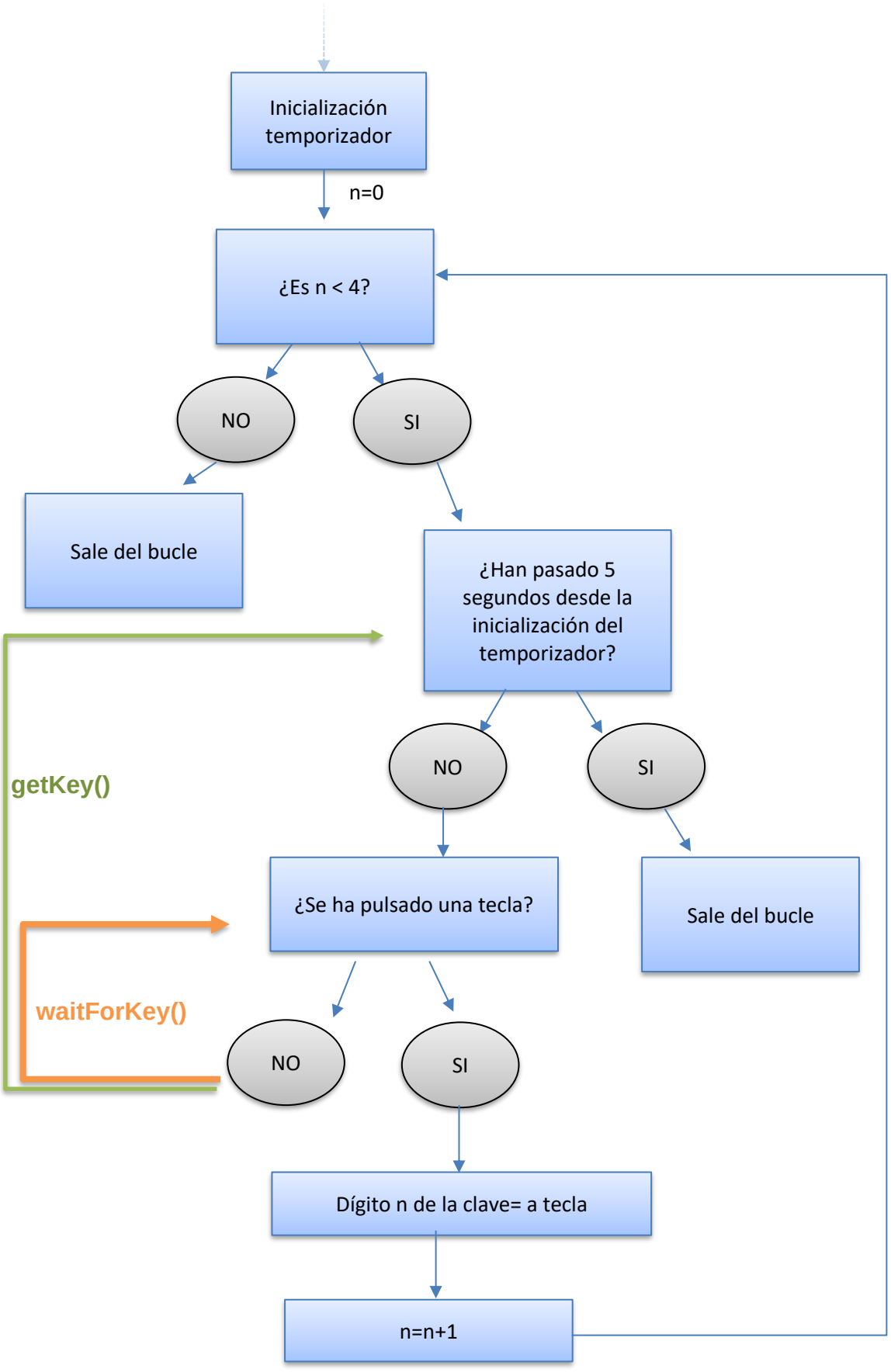


Figura 31. Esquema de funcionamiento del temporizador

5.2.2. Intermitencia de las señales de emergencia

En el segundo caso, las señales de aviso, tanto lumínicas como sonoras, y el parpadeo de la información en la pantalla LCD, deben de estar activas con una intermitencia de un segundo en cada estado mientras dure la condición, es decir, un segundo en estado alto (encendidas) o HIGH en programación, y un segundo en estado bajo (apagadas) o LOW. Para conseguir esto, en el momento en el que se cumple la condición en la que el dispositivo tiene que actuar de esta manera, se activa un temporizador que, a diferencia de la situación anteriormente mencionada, el temporizador se debe reiniciar cada segundo, y en cada reinicio el estado de las salidas cambiará. Se resume el funcionamiento gráficamente en el siguiente esquema.

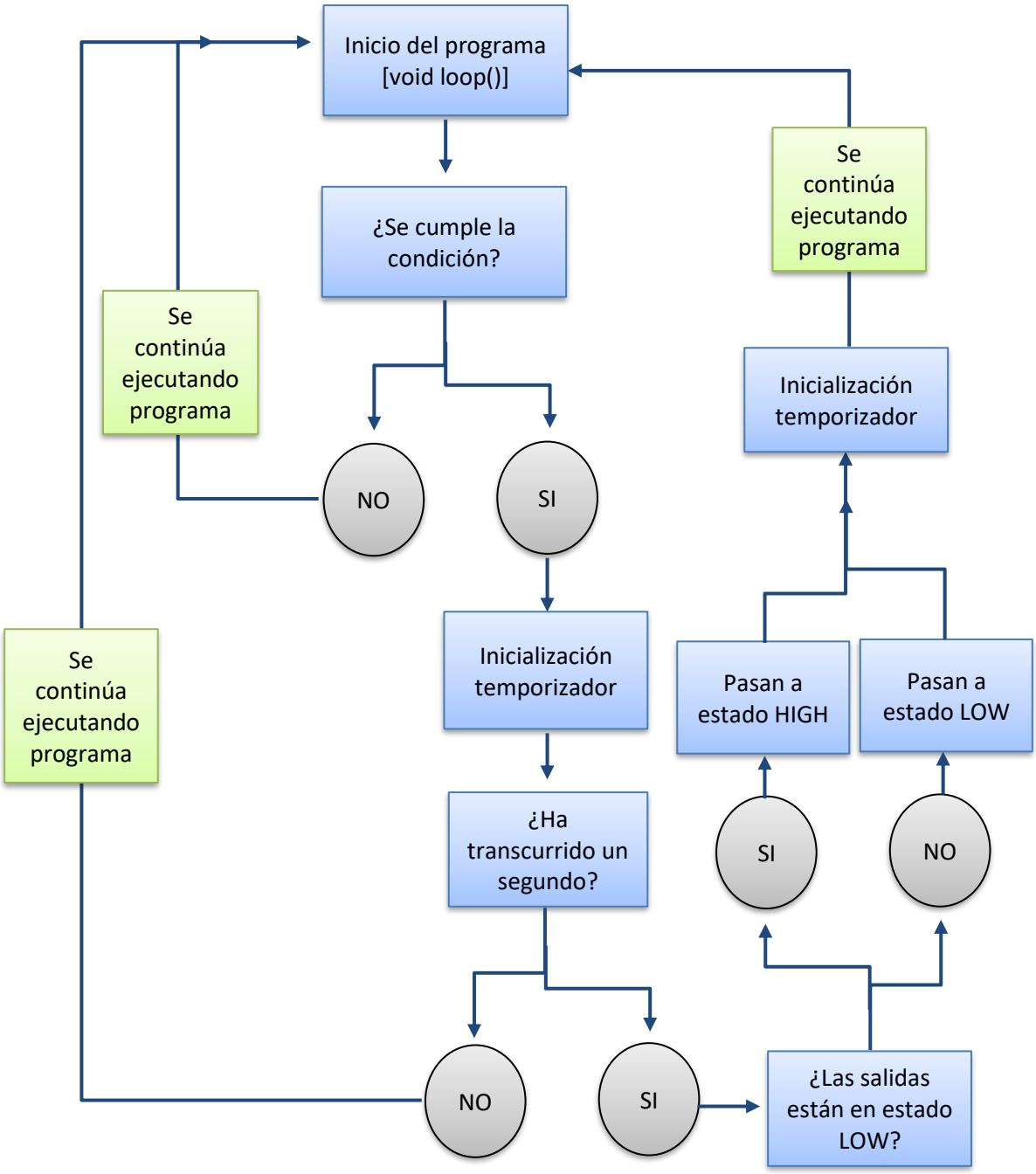


Figura 32. Esquema de intermitencia de las señales de emergencia

5.3. Apagado de las señales de emergencia

Como se ha comentado en capítulos anteriores, en algunas situaciones de comportamiento fuera del funcionamiento normal, algunas señales de emergencia se apagan al introducir el código del personal autorizado. En las primeras pruebas del programa se detectó que no se había configurado de manera adecuada y que una vez se apagaban, cuando las condiciones cambiaban y luego volvían a la condición anterior, las señales de aviso no se encendían porque la variable que controla su estado no se había reiniciado al salir de la condición y mantenía el estado de apagado. Para solucionar este problema, cuando se cumple una condición con estas características y se introduce la clave, dichas señales se apagan mientras dura esa condición, pero una vez cambia, se reinician, de manera que si vuelve a darse la misma condición de emergencia estas señales se vuelvan a activar.

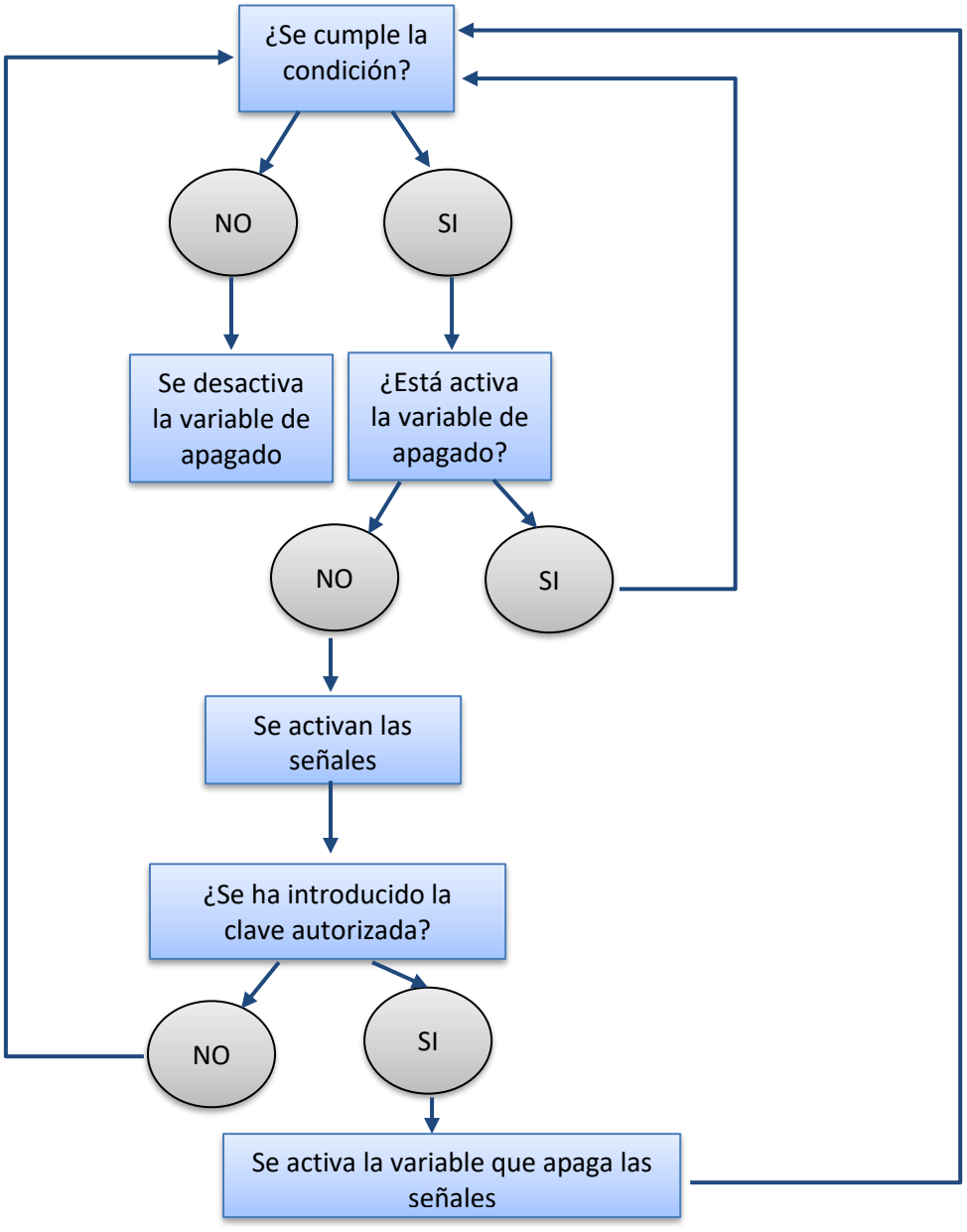


Figura 33. Esquema de apagado de señales en situación de emergencia

CAPÍTULO 6. PRUEBAS DE FUNCIONAMIENTO Y AJUSTES

La última parte que se va a exponer en esta memoria, antes de llegar a las conclusiones, son las pruebas de funcionamiento y los ajustes necesarios del prototipo. Se trata de un trabajo que es imprescindible, ya que permite comprobar que todo lo que se hace, tanto a nivel lógico y de planteamiento, como a nivel de ejecución, está bien y se obtienen los resultados esperados.

Este tipo de pruebas experimentales han sido necesarias a lo largo del desarrollo de todo el proyecto. Desde su inicio hasta su fin, ha sido fundamental ir haciendo comprobaciones conforme se iba avanzando y era posible poner a prueba la parte desarrollada, obteniendo unos resultados posibles de analizar y comparar con los esperados.

Esta manera de proceder resulta muy útil para identificar los fallos rápidamente, tanto en etapas tempranas del proyecto como en estadios más avanzados, ya que al ir haciendo comprobaciones a la vez que se va avanzando, las pruebas se pueden acotar más y evaluar aspectos concretos e ir haciendo correcciones.

En el punto 3.3 ya se ha introducido el material usado en estos ensayos, aunque también hay que considerar otras pruebas que no han requerido de ningún material adicional más allá del desarrollo de códigos de apoyo. Estos programas de apoyo, muy sencillos, se han utilizado para comprobar que el montaje de los componentes era el adecuado. Debido a que el hardware se debe montar antes de desarrollar el software que controla el dispositivo, se necesita saber si este montaje se ha hecho correctamente y todos los componentes están ensamblados como es debido. Simplemente se trata de comprobar que la placa se comunica correctamente con ellos, para lo que se utilizan programas que constan de la librería necesaria y una instrucción que saca por el monitor serie los valores leídos de una entrada o manda una instrucción a un actuador. Estos pequeños códigos permiten, por un lado, comprobar que la parte de electrónica está correcta, y por otro, que la comunicación de la placa con los componentes y su configuración se ha hecho correctamente.

A la hora de hacer comprobaciones sobre la escritura y funcionalidad del programa, ha sido necesario reproducir las condiciones de la sauna húmeda en cuanto a temperatura y humedad relativa se refiere, y simular todas las casuísticas posibles ya mencionadas en capítulos anteriores además de las transiciones de unas condiciones a otras.

Como las temperaturas de funcionamiento de la sauna húmeda son relativamente altas en comparación con la temperatura ambiente, y con los medios utilizados se requiere de cierto tiempo para alcanzarlas, además de que no se tiene un control más allá de esperar a que se alcancen ciertas temperaturas y alejar el sensor de la fuente de calor para que se vuelva a enfriar. Valorando esta premisa y teniendo en cuenta que, durante el desarrollo del código, se ha ido comprobando su funcionalidad por partes y no en su conjunto, hasta que ya estaba en un punto muy avanzado; para realizar estas pruebas y no tener que alcanzar valores tan altos en cada comprobación, se ha ido variando el valor de los límites de la temperatura y la humedad definidos en el código para generar las condiciones deseadas de una manera más sencilla y rápida. Esto no tiene ninguna connotación negativa, pues se trata de crear situaciones análogas con otros rangos de funcionamiento.

Antes de comenzar con las comprobaciones del código, se ha probado que el sensor estuviera en buenas condiciones y realizara unas mediciones reales. Para ello, simplemente se ha comparado la lectura obtenida del sensor DHT22 con un termómetro

higrómetro tomando diferentes medidas y se han obtenido los mismos resultados en todas ellas. Como se puede observar en la figura 34, la lectura de la temperatura prácticamente es la misma, difiere en 0,1ºC en ambas medidas. Sin embargo, la humedad relativa difiere en 7 puntos, sin contar los decimales, ya que el termómetro higrómetro no tiene esa resolución. Teniendo en cuenta que el sensor DHT tiene una precisión en la medición de la humedad relativa del ±2% y de 0,5ºC en la temperatura, y el termómetro higrómetro tiene una precisión en la medición de la humedad relativa de ±5% y de 1ºC en la de la temperatura y que además tiene un contacto menos directo con el ambiente que el sensor, podemos considerar estos resultados como válidos para confirmar que el sensor realiza una correcta medición de ambas variables.

Figura 34. Comparación de la medida de la temperatura y humedad del sensor DHT22 con la de un termómetro higrómetro

Una vez se ha comprobado que los componentes están correctamente conectados y configurados para que se comuniquen correctamente con la placa, y que el sensor DHT22 funciona correctamente, se ha podido seguir con la escritura de código y las correspondientes pruebas experimentales.

Como se ha comentado anteriormente en este capítulo, para comprobar el correcto funcionamiento de diferentes fragmentos de código, se han ido adaptando los límites de

las variables para que se cumpliera la condición deseada de una manera más rápida, sin tener que montar todo el entorno de pruebas. Simplemente en caso de querer comprobar un cambio de condición, como sí que se necesita una variación en las medidas, con el secador de pelo ha sido suficiente.

Gracias a estas pruebas se han ido detectando todos los errores de programación, pudiendo subsanarlos. La mayoría de estos errores han sido de planteamiento inicial porque no se han tenido en cuenta todas las posibilidades y/o consecuencias de una acción. Por ejemplo, como se ha detallado en el Capítulo 5, en una primera ejecución del código no se había tenido en cuenta que, al apagar los zumbadores, no se reiniciara el estado al salir de la condición. Esto daba lugar a que, si se salía de esa condición y se volvía a dar más tarde, los zumbadores no se encendiesen. Aunque esto es muy sencillo de solucionar en el código, si no se hubiera detectado, es un error que puede dar lugar a situaciones graves en su aplicación. La situación podría ser la siguiente. Al comenzar la ejecución del código, la sauna se encuentra en condiciones normales. Si se da la situación de que la temperatura supera el límite máximo, se tienen que encender todas las señales de aviso. Tal como estaba programado el sistema, al introducir el código de personal autorizado se apagan los zumbadores. Se subsana el problema y la sauna deja de tener una temperatura superior al límite, pero en otro momento vuelve a haber un error que genera esta situación de emergencia. En este caso no se enciende la señal de aviso sonora, lo que puede provocar una respuesta más lenta por parte del personal y de los usuarios.

Una vez se consideran todos los errores corregidos y se ha completado el código, sí que se ha realizado una prueba experimental con los valores de funcionamiento del baño turco, realizando las modificaciones pertinentes para generar todas las casuísticas posibles, comprobando así el correcto funcionamiento en su conjunto.

Para elegir los valores límites de temperatura, hay que conocer el rango de funcionamiento del baño turco y la distribución de la temperatura dentro de la sauna. De la información recabada y expuesta en la introducción de esta memoria, se sabe que la temperatura dentro de un baño turco oscila entre los 25ºC y 50ºC. Pero no quiere decir que la temperatura sea constante en toda la estancia y que se puede encontrar entre estos valores, sino que, midiendo la temperatura desde el suelo hasta la parte más alta, se encuentra un gradiente de temperatura cuyos valores irán desde 25ºC en el suelo hasta 50ºC en el techo, pasando por todos los puntos intermedios de uno a otro.

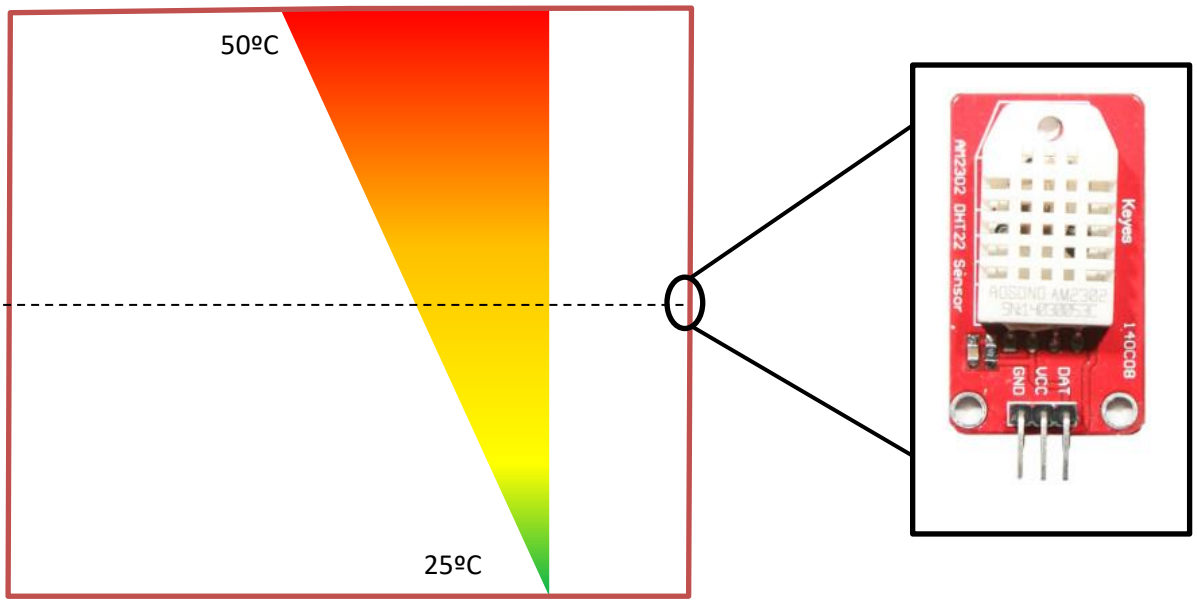


Figura 35. Distribución de la temperatura dentro del baño turco

Debido a esta distribución de la temperatura lo ideal sería colocar un sensor en cada extremo de la estancia y que cada uno se encargase de controlar un límite estableciendo un margen de error con los límites del rango de funcionamiento. Pero al utilizar solo un sensor, este se coloca en el punto intermedio entre el suelo y el techo y se establecen como límites inferior y superior los valores límites del rango de funcionamiento, ya que, en un punto intermedio, con un funcionamiento adecuado, la temperatura debe estar entre ambos valores.

6.1. Evaluación final y ajustes

Desarrollado el código completamente y establecido es rango de funcionamiento, se procede a realizar la prueba experimental final. Esta prueba servirá para comprobar el funcionamiento del dispositivo bajo las condiciones para las cuales ha sido diseñado y para subsanar pequeños errores de programación que se han pasado por alto, como limpiar la pantalla al cambiar de una instrucción a otra, reiniciar variables y otros errores parecidos.

Antes de comenzar con el ensayo sobre el prototipo, se ha considerado necesario estudiar la evolución de la temperatura y la humedad en el entorno de pruebas bajo las diferentes condiciones. De esta manera, se tiene conocimiento del comportamiento de ambas variables y se pueden crear las diferentes situaciones necesarias para la comprobación y ajuste del prototipo de la manera más eficiente. Las siguientes gráficas reflejan esta evolución en el tiempo.

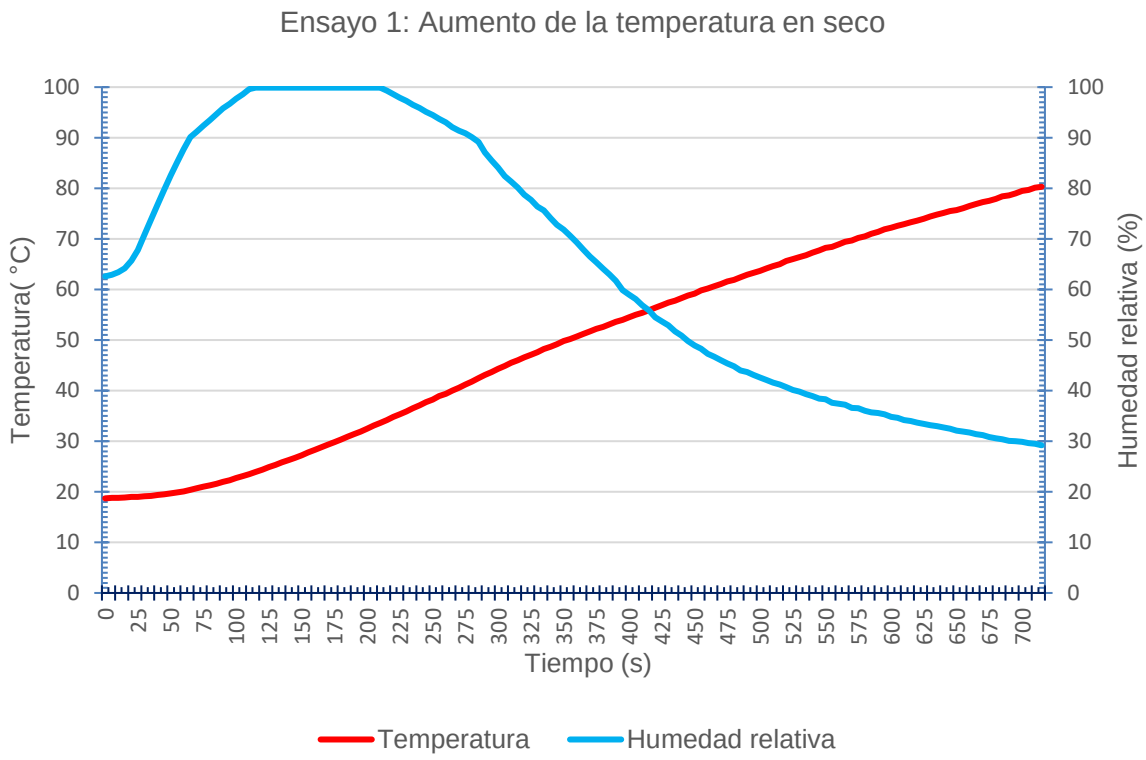


Figura 36. Gráfica representativa de los datos obtenidos en el ensayo 1 (aumento de la temperatura en seco)

En el ensayo 1 se observa un aumento de la humedad al inicio, que se debe a la evaporación de la humedad en el ambiente y que se condensa dentro del recipiente. La temperatura tiene un incremento más suave durante los primeros segundos, debido a que el quemador se acaba de encender y gran parte del calor lo absorbe el propio material. Conforme la temperatura sigue aumentando paulatinamente, se aprecia un descenso de la humedad, esto es debido a que la humedad relativa está directamente relacionada con la temperatura, la cual define la presión de saturación del vapor de agua. No obstante, no es el único factor que influye en su medición. También dependerá de la presión del sistema.

En resumen, la humedad relativa obedece a dos reglas:

- A mayor temperatura, el aire se vuelve más seco (la humedad relativa disminuye).
- A menor presión, el aire se vuelve más seco (la humedad relativa disminuye).

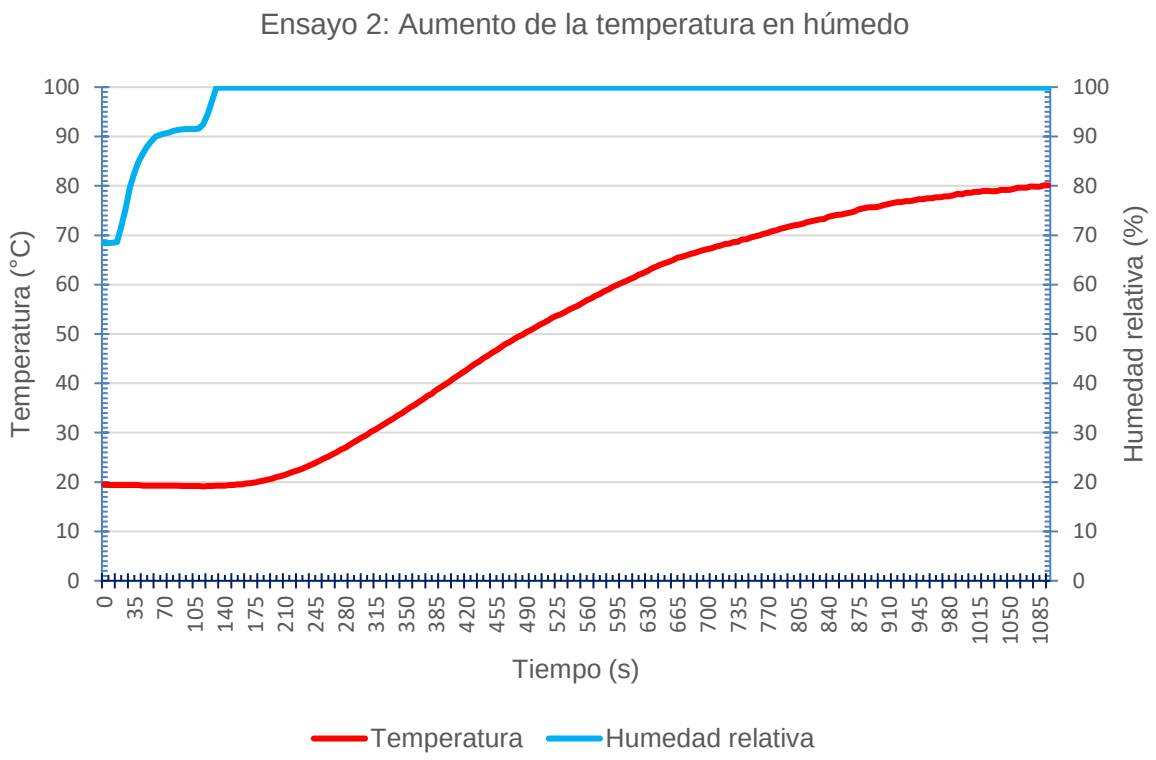


Figura 37. Gráfica representativa de los datos obtenidos en el ensayo 2 (aumento de la temperatura en húmedo)

En el ensayo 2 se consigue generar unas condiciones en las que la humedad relativa alcance el 100% y se mantenga en el tiempo, a pesar de que la temperatura aumente. Esto es debido a que la parte superior del quemador se llena de agua y el vapor incide directamente sobre el sensor que está situado justamente encima a unos centímetros. Cuando el agua se evapore por completo la humedad relativa comenzará a disminuir cumpliéndose la regla anteriormente mencionada (No se refleja esto en la figura 37).

Una vez se tiene conocimiento de la evolución de las variables en el entorno de pruebas se comienza con los ensayos correspondientes para detectar errores y ajustar el prototipo.

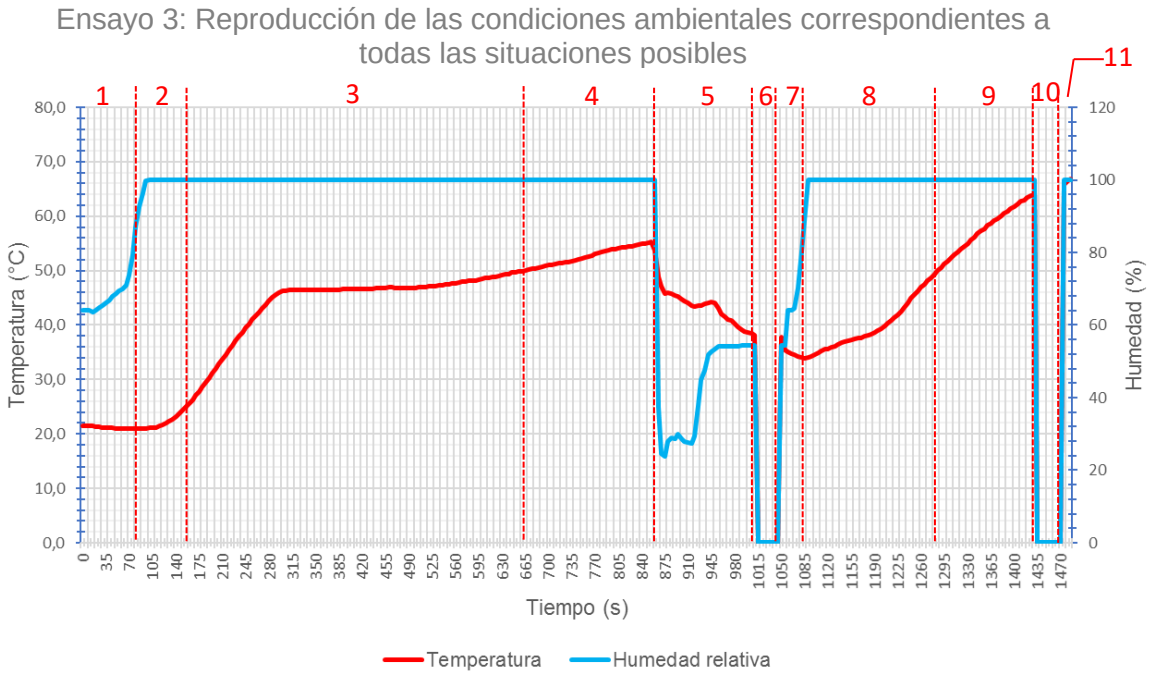
En cada casuística considerada es necesario comprobar que el prototipo actúa de acuerdo con lo establecido teóricamente en las siguientes situaciones:

- Al cumplirse por primera vez la condición.
- Al dejar de cumplirse.
- Al volver a cumplirse.

Hay que reproducir estas tres situaciones para comprobar que el prototipo actúa como se espera en cada caso, ya que, bajo la misma condición, una vez se introduce el código del personal, ciertos actuadores cambiarán su comportamiento, y una vez se sale de la condición, hay que restaurar los valores iniciales.

6.1.1. Procedimiento llevado a cabo para el ensayo

Se comienza el ensayo a temperatura ambiente, con condiciones de temperatura y humedad por debajo de los valores mínimos del rango de funcionamiento, lo que nos permite comprobar el correcto funcionamiento bajo estas condiciones de funcionamiento fuera del rango óptimo. Se continúa colocando el sensor en el entorno de pruebas, lo que provoca un aumento repentino de la humedad relativa y un aumento progresivo de la humedad permitiendo comprobar otras situaciones posibles dentro de la misma condición. Se mantiene así, pasando por las condiciones de funcionamiento normal y aumentando la temperatura hasta que alcanza límite superior y lo sobrepasa, quedando comprobada la condición T>Tmax. Apartando el sensor del quemador se induce una disminución de la humedad relativa con el secador para comprobar la última situación posible dentro de la condición de T<Tmin y/o H<Hmin o H<Hmax. Se continúa desconectando el sensor para comprobar que el dispositivo actúa de manera esperada al no recibir datos del este y luego se vuelve a conectar, confirmando que se vuelve a leer los datos del sensor correctamente y para finalizar se vuelve a sobrepasar el límite máximo de temperatura y a desconectar y conectar el sensor para comprobar que la transición de una a otra se hace correctamente volviéndose a activar las señales de emergencia. Se han recogido los datos de este ensayo representándose en la siguiente gráfica de la figura 38.



(1) T<Tmin y H<Hmin
(2) T<Tmin y Hmin<H<Hmax
(3) Tmin<T<Tmax y Hmin<H<Hmax
(4) T>Tmax y Hmin<H<Hmax
(5) Tmin<T<Tmax y H<Hmin
(6) No se leen datos del sensor
(7) Tmin<T<Tmax y H<Hmin
(8) Tmin<T<Tmax y Hmin<H<Hmax
(9) T>Tmax y Hmin<H<Hmax
(10) No se leen datos del sensor
(11) T>Tmax y Hmin<H<Hmax

Figura 38. Gráfica representativa de los valores tomados en la prueba experimental de todas las casuísticas

Una vez que se verifica que el dispositivo cumple todas las funciones requeridas, se vuelve a conectar el pin de datos del sensor DHT22 para continuar con la prueba y terminar de comprobar que se ha programado correctamente el reinicio de la variable que apaga las señales de aviso al introducir el código y que si vuelve a esta situación o se cumpla otra en las que se deben de activar, estas lo hagan.

Durante esta prueba se han detectado los siguientes errores:

- Al introducir la clave correcta la primera vez, las siguientes veces solo con pulsar la letra C abrirá la puerta una vez finaliza la cuenta atrás del temporizador.

  Después de repasar el código se ha detectado que la condición en la que es necesario introducir una clave para poder reconocerla y hacer la comparación, no estaba activa, simplemente porque, por equivocación, se había puesto un punto y coma entre la condición y el primer corchete.

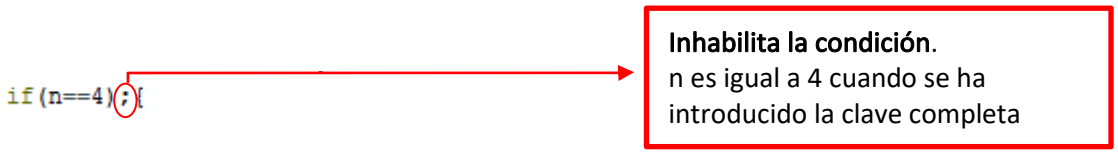


Figura 39. Error detectado

- Faltan lcd.clear(), comando para borrar lo mostrado por pantalla, en algunos puntos que hay que identificar.

Estos errores detectados se han explicado como ejemplo de la utilidad de estas pruebas, que son imprescindibles. Pero no se ha considerado necesario enumerar todos los errores que se han ido identificando.

Tras comprobar que el prototipo funciona como se espera tras todas las situaciones posibles en el caso de error en la lectura, se continúa con el resto de casuísticas, comprobando que se ejecutan todas las instrucciones de la manera esperada en las tres situaciones anteriormente nombradas. Gracias a las reiteriativas pruebas realizadas, aislando condiciones conforme se ha ido avanzando en el código, no se han encontrado más errores a destacar.

CAPÍTULO 7. CONCLUSIONES

Tras completar los ensayos experimentales y comprobar que el prototipo funciona acorde a lo esperado, se da por concluido este proyecto, con resultados satisfactorios. La idea principal de este trabajo era poder desarrollar un dispositivo que permitiera controlar la temperatura y humedad de una sala y que dispusiese de un sistema de seguridad para poder identificar rápidamente comportamientos fuera del rango de funcionamiento. En este caso se ha querido enfocar su desarrollo para el uso específico en un baño turco, ya que se ha identificado una necesidad real, pero este prototipo se puede adaptar fácilmente a su uso en otras instalaciones, simplemente ajustando el rango de funcionamiento.

El desarrollo de este proyecto ha servido para conocer el papel que tiene Arduino hoy en día a nivel educativo y práctico, ya que pone al alcance de todos, de una manera sencilla y económica, la tecnología necesaria para desarrollar una inmensa variedad de proyectos útiles.

CAPÍTULO 8. BIBLIOGRAFÍA

A., C. (2017, 26 diciembre). *Chris--A/Keypad*. GitHub. https://github.com/Chris--A/Keypad

Allsop, R. O. (1890). *The Turkish Bath: Its Design and Construction* (Vol. 1). Library of Alexandria.

Arduino Library - Keypad. (2016, 1 julio). DomoticX Knowledge Center. http://domoticx.com/arduino-library-keypad/

Arduino paso a paso. (2018). Forma Roboti-k.

Arduino Playground - I2cScanner. (s. f.). The Arduino Playground. https://playground.arduino.cc/Main/I2cScanner/

Bermejo Herrero, I. (s. f.-a). *Buzzer Activo 5V - Zumbador activo »*. IBEROBOTICS. https://www.iberobotics.com/producto/buzzer-activo-5v-zumbador-activo/

Bermejo Herrero, I. (s. f.-b). *Micro Servo SG90 1.8Kg/9g/0.12seg »*. IBEROBOTICS. https://www.iberobotics.com/producto/micro-servo-towerpro-sg90-1-8kg9g0-12seg/

Breathnach, T. (2004). *For health and pleasure: the Turkish Bath in Victorian Ireland*. Victorian Literature and Culture, 32(1), 159-175.

Fitzgerald, S., & Shiloh, M. (2014). *El libro de proyectos de Arduino*. Turin, Italia: Arduino.

I. (2019, 11 octubre). *DHT22: el sensor de temperatura y humedad de precisión*. Hardware libre. https://www.hwlibre.com/dht22/

Kauppinen, K. (1997). *Facts and fables about sauna*. Annals of the New York Academy of Sciences, 813, 654-662.

Kukkonen-Harjula, K., & Kauppinen, K. (2006). *Health effects and risks of sauna bathing*. International journal of circumpolar health, 65(3), 195-205.

Lajara Vizcaino, J. R. (2014). *Sistemas integrados con Arduino*. Marcombo.

Llamas, L. (2016, 2 octubre). *Usar un teclado matricial con Arduino*. Luis Llamas. https://www.luisllamas.es/arduino-teclado-matricial/

Lucia Reyes, A. (2005). *Diferencia entre sauna y baño turco*. Revista Palestra, 40–43.

Ribas Lequerica, J. (2013). *Manual imprescindible de Arduino práctico*. Anaya Multimedia.

Tecnopura. (2020, 2 enero). *Pantalla LCD 1602 con módulo I2C soldado - Display 16x2 dígitos*. https://www.tecnopura.com/producto/pantalla-lcd-1602-con-modulo-i2c-soldado-display-16x2-digitos/

Weitzel, A. (2020, 17 junio). *andresWeitzel/LiquidCrystal_I2C.h*. GitHub. https://github.com/andresWeitzel/LiquidCrystal_I2C.h/tree/master/Arduino-LiquidCrystal-I2C-library-master

Parte II
Presupuesto

PRECIO DE LA MANO DE OBRA

Código			Precio (€/h)
001	h	Ingeniera graduada en Tecnologías Industriales	35,00

PRECIO DE LOS MATERIALES

Prototipo

Código			Precio unitario (€)
002	u	Placa Arduino Mega 2560	35,00
003	u	Led rojo [1]	0,045
004	u	Led verde [1]	0,045
005	u	Zumbador activo	1,27
006	u	Teclado matricial 4x4	2.99
007	u	Microservo tower pro 9g SG90	3,88
008	u	Pantalla LCD con módulo I2C	6,87
009	u	Placa de pruebas	4,99
010	u	Condensador 100 µF [2]	0,044
011	u	Resistencia 220Ω [3]	0,034
012	u	Cables hembra-macho y macho-macho [4]	0,028
013	u	Conectores [5]	0,01
014	u	Cable USB	1,80
016	u	Sensor DHT22	3,44
018	u	Flux para soldar	0,75

[1] Un pack de 50 leds de colores variados cuesta 2,25 €

[2] Un pack de 50 condensadores cuesta 2,22 €

[3] Un pack de 200 resistencias cuesta 6,85€

[4] Un pack de 120 unidades cuesta 3,41 €

[5] La tira de 40 unidades cuesta 0,40€

Código			Precio (€/m)
015	m	Cable trenzado	0,54
017	m	Estaño	1,60

Herramientas

Código			Precio unitario (€)
019	u	Soldador	10,00
020	u	Pela cables	2,00

Entorno de pruebas

Código			Precio unitario (€)
021	u	Quemador de aceites esenciales	1,20
022	u	Vela [6]	0,23
023	u	Termómetro higrómetro	6,95
024	u	Mechero	1,50
025	u	Secador de pelo de viaje	5,00
026	u	Envase	1,00
027	u	Bayeta [7]	0,25
028	u	Bol	0,50
029	u	Cartulina	0.40

[6] Un pack de 12 velas cuesta 2,75

[7] Tres unidades cuestan 0,75

Medio de trabajo

Tienen coste cero por disponer de ellos previamente o ser gratuitos.

Código			Precio unitario (€)
030	u	Ordenador	0,00
031	u	IDE Arduino	0,00
032	u	Impresora	0,00

PRECIOS UNITARIOS

Nº Actividad		Descripción de las unidades de obra	Medición	Precio	Importe
01		DISEÑO Y DESARROLLO DEL PROTOTIPO			
01.01	u	Diseño y montaje del hardware	1,00	1843,44	1843,44
01.02	u	Preparación y desarrollo del software	1,00	4242,00	4242,00
01.03	u	Estudio del rango de funcionamiento del servo	1,00	212,50	212,50
		Precio total unidad de obra 01			**6297,94 €**
02		DISEÑO Y DESARROLLO DEL ENTORNO DE PRUEBAS			
02.01	u	Diseño y montaje del entorno de pruebas	1,00	156,68	156,68
02.02	u	Pruebas de funcionamiento	1,00	1062,02	1062,02
		Precio total unidad de obra 02			**1218,70 €**

PRECIOS DESCOMPUESTOS

Nº Actividad	Código		Descripción	Rendimiento	Precio	Importe
01			DISEÑO Y DESARROLLO DEL PROTOTIPO			
01.01	**U01.01**	u	**DISEÑO Y MONTAJE DEL HARDWARE.** Incluye el trabajo de análisis del problema y elección de una solución, incluyendo la selección de materiales y su ensamblado.			
	001	h	Ingeniera Graduada en Tecnologías Industriales	50,00	35,00	1750,00
	002	u	Placa Arduino Mega 2560	1,00	35,00	35,00
	003	u	Led rojo	2,00	0,045	0,09
	004	u	Led verde	1,00	0,045	0,05
	005	u	Zumbador activo	2,00	1,27	2,54
	006	u	Teclado matricial 4x4	1,00	2.99	1,00
	007	u	Microservo tower pro 9g SG90	1,00	3,88	3,88
	008	u	Pantalla LCD con módulo I2C	1,00	6,87	6,87
	009	u	Placa de pruebas	1,00	4,99	4,99
	010	u	Condensador 100 µF	1,00	0,044	0,04
	011	u	Resistencia 220Ω	3,00	0,034	0,10
	012	u	Cables conectores hembra-macho y macho-macho	26,00	0,028	0,73
	013	u	Conectores	3,00	0,01	0,03
	014	u	Cable USB	1,00	1,80	1,80
	015	m	Cable trenzado	2,00	0,54	1,08
	016	u	Sensor DHT22	1,00	3,44	3,44
	017	m	Estaño	0,50	1,60	0,80
	018	u	Flux para soldar	1,00	0,75	0,75
	019	u	Soldador	1,00	10,00	10,00
	020	u	Pela cables	1,00	2,00	2,00
			Coste total de mano de obra y material			1825,19
		%	Costes directos y complementarios	0,01	1825,19	18,25
			Coste total de unidad de obra			**1843,44 €**

01.02	**U01.02**	u	**PREPARACIÓN Y DESARROLLO DEL SOFTWARE.** Incluye la instalación del IDE, comunicación con la placa y los componentes y el desarrollo del código.			
	001	h	Ingeniera Graduada en Tecnologías Industriales	120,00	35,00	4200,00
	029	u	Ordenador	1,00	0,00	0,00
	030	u	IDE Arduino	1,00	0,00	0,00
			Coste total de mano de obra y material			4200,00
		%	Costes directos y complementarios	0,01	4200,00	42,00
			Coste total de unidad de obra			**4242,00 €**

01.03	**U01.03**	u	**ESTUDIO DEL RANGO DE FUNCIONAMIENTO DEL SERVO.** Incluye el montaje del entorno de pruebas, desarrollo del código de pruebas y los ensayos levados a cabo.			
	001	h	Ingeniera Graduada en Tecnologías Industriales	6,00	35,00	210,00
	029	u	Cartulina	1.00	0.40	0,40
	032	u	Impresora	1.00	0,00	0,00
			Coste total de mano de obra y material			210,40
		%	Costes directos y complementarios	0,01	210,40	2,10
			Coste total de unidad de obra			**212,50 €**

02			DISEÑO Y DESARROLLO DEL ENTORNO DE PRUEBAS			
02.01	**U02.01**	u	**DISEÑO Y MONTAJE DEL ENTORNO DE PRUEBAS.** Incluye la elección de materiales y montaje del entorno de pruebas.			
	001	h	Ingeniera Graduada en Tecnologías Industriales	4,00	35,00	140,00
	021	u	Quemador de aceites esenciales	1,00	1,20	1,20
	022	u	Vela	1,00	0,23	0,23
	023	u	Termómetro higrómetro	1,00	6,95	6,95
	025	u	Secador de pelo de viaje	1,00	5,00	5,00
	026	u	Envase	1,00	1,00	1,00
	027	u	Bayeta	1,00	0,25	0,25
	028	u	Bol	1,00	0,50	0,50
			Coste total de mano de obra y material			155,13
		%	Costes directos y complementarios	0,01	155,13	1,55
			Coste total de unidad de obra			**156,68 €**
02.02	**U02.02**	u	**PRUEBAS DE FUNCIONAMIENTO.** Incluye las pruebas de funcionamiento del prototipo y ajustes.			
	001	h	Ingeniera Graduada en Tecnologías Industriales	30,00	35,00	1050,00
	024	u	Mechero	1,00	1,50	1,50
			Coste total de mano de obra y material			1051,50
		%	Costes directos y complementarios	0,01	1051,50	10,52
			Coste total de unidad de obra			**1062,02 €**

PRESUPUESTO

Nº Actividad		Importe
01	Diseño y desarrollo del prototipo	6297,94 €
02	Diseño y desarrollo del entorno de pruebas	1218,70 €

Presupuesto de ejecución material	7516,64 €
Gastos generales 13%	977,16 €
Beneficio Industrial 6%	451,00 €
Presupuesto de ejecución por contrata	8944,80 €
IVA 21%	1878,41 €
Presupuesto Base de Licitación	10823,21 €

Asciende el presente presupuesto a la expresada cantidad de

DIEZ MIL OCHOCIENTOS VEINTITRES EUROS Y VEINTIUN CÉNTIMOS

Anexo I
Código de funcionamiento del prototipo

```
//LIBRERÍAS

#include <DHT.h>
#include <LiquidCrystal_I2C.h>  //importa la libreria LiquidCrystal_I2C para trabajar con la pantalla LCD con módulo I2C
#include <Wire.h>  //importa la librería Wire que nos permite trabajar con el bus I2C
#include <Servo.h>      //importa la libreria Servo
#include <Keypad.h>     // importa la libreria Keypad

//DHT



#define pinDHT22 8   // DHT en pin digital 8
#define tipoDHT DHT22 //Declaración del modelo de sensor de temperatura y humedad

DHT dht(pinDHT22, tipoDHT);

//LÍMITES
//Declaración de variables que definen los límites de temperatura (°C) y humedad (%)inferior y superior

float Tmax=50.00;
float Tmin=25.00;
float Hmin=90.00;
float Hmax=100.00;

//Pantalla LCD (definición de la dirección del bus I2C y de las características de la pantalla)
LiquidCrystal_I2C lcd(0x27,16,2);

//SERVO
Servo puertaServo;



  //Declaración de variables que contienen las posiciones del servo (puerta abierta o puerta cerrada)
int abierto=550;
int cerrado=2300;
```

```
//KEYPAD (TECLADO MATRICIAL)
//Declaración de variables para la configuración de las teclas del Keypad
const byte FILAS = 4;     // define numero de filas
const byte COLUMNAS = 4;    // define numero de columnas
char keys[FILAS][COLUMNAS] = {    // define la distribucion de teclas en las filas y columnas
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

byte pinesFilas[FILAS] = {36,34,32,30};   //adjudicación pines correspondientes a las filas
byte pinesColumnas[COLUMNAS] = {28,26,24,22}; // adjudicación pines correspondientes a las columnas

Keypad teclado = Keypad(makeKeymap(keys), pinesFilas, pinesColumnas, FILAS, COLUMNAS);  // crea objeto

//Declaración de variables para el control de acceso
String clave = "0000"; //Variable para almacenar la clave introducida
String claves_abonados[] = {"1111", "2222","3333","4444","5555","6666", "7777", "8888"}; //Claves de abonados con acceso a la sauna
String codigo_emergencia = "1234"; //Clave de acceso de personal autorizado


//Adjudicación pines zumbadores
#define Buzzer_OUT 7
#define Buzzer_IN 6

//Adjudicación pines LED
#define LEDrojoIN 2
#define LEDrojoOUT 3
#define LEDverde 4

//Declaración de variables
```

```
int estadoLED=LOW;
long previoMillis=0;
long intervalo=1000;
int pantalla=0;
long espera= 5000;
long tiempo_transcurrido=0;
int salir=0;
char TECLA;
char PULSA;
int apagar=0;
int apagar2=0;
unsigned long contador=0;
unsigned long contajeMillis=0;

int n;
int Tiempo=0;
void setup() {
     Serial.begin(9600);

  dht.begin();  //inicializa sensor DHT22

  puertaServo.attach(10);//Servo conectado en el pin digital 10 PWM

  //Inicialización pantalla LCD
  lcd.init();
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(0,0);

  //Configuración pin LEDs (salida)
  pinMode(LEDrojoIN,OUTPUT);//LED rojo fuera de la sauna
  pinMode(LEDrojoOUT,OUTPUT);//LED rojo dentro de la sauna
  pinMode(LEDverde,OUTPUT);//LED verde (fuera de la sauna9. Utilizado
únicamente para señalizar el paso a la sauna cuando se introduce una
clave correcta.

  //Configuración zumbadores activos (salida)
  pinMode(Buzzer_OUT, OUTPUT); //Zumbador fuera de la sauna
  pinMode(Buzzer_IN, OUTPUT); //Zumbador dentro de la sauna

   puertaServo.writeMicroseconds(cerrado); //Inicialización del servo en
```

```
la posición de reposo (puerta cerrada)

   //Inicialización señales de aviso apagadas)
    digitalWrite(LEDrojoOUT,LOW);
      digitalWrite(LEDrojoIN,LOW);
      digitalWrite (LEDverde,LOW);
      digitalWrite(Buzzer_OUT,LOW);
      digitalWrite(Buzzer_IN,LOW);
   }

void loop() {

//lectura de humedad y temperatura desde el sensor DHT22
float h = dht.readHumidity();
float t = dht.readTemperature();
Tiempo=Tiempo+5;
Serial.print(t);
Serial.print("    ");

Serial.print (h);
Serial.print("    ");
Serial.println (Tiempo);
delay(5000);

if (isnan(t) || isnan(h)) {//Si no se puede obtener la lectura del sensor
se inhabilitará la sauna

    apagar2=0;
    lcd.setCursor(0,0);
    lcd.print ("Fuera de uso    ");
    lcd.setCursor(0,1);
    lcd.print("   NO PASAR     ");

    //Inicio temporizador intermitencia de señales
     contajeMillis=millis();

  if(contajeMillis - previoMillis > intervalo){ //Condición de cambio de
estado
    previoMillis=contajeMillis;
    if (estadoLED==LOW){
```

```
      estadoLED=HIGH;}
      else{

        estadoLED=LOW;}
if (apagar!=1){
  digitalWrite (LEDrojoIN,estadoLED);
  digitalWrite (LEDrojoOUT, estadoLED);

  digitalWrite(Buzzer_OUT,estadoLED);
 digitalWrite(Buzzer_IN,estadoLED);
  }
 else{
    digitalWrite (LEDrojoIN,LOW);
  digitalWrite (LEDrojoOUT,LOW);
    digitalWrite(Buzzer_OUT,LOW);
 digitalWrite(Buzzer_IN,LOW);
 }
  }


 TECLA= teclado.getKey(); //Se prepara para recibir información del
keypad


  if (TECLA != 0){

    switch(TECLA){

      case 'C':

          digitalWrite(Buzzer_OUT,LOW);
 digitalWrite(Buzzer_IN,LOW);
       contador=millis(); //Inicio del temporizador

       lcd.clear();

       if (salir==1){//Si el temporizador ha finalizado sale del case
       break;
       salir=0; //Se resetea el t
       }
    n=0;
```

```
        while (n<4){
           tiempo_transcurrido=millis();
           if (tiempo_transcurrido-contador > espera){ //Comprueba si ha
transcurrido el tiempo establecido para introducir la clave
             salir=1;
             break;
                   }
           else{

             PULSA=teclado.getKey();



             if (PULSA!=0){

               clave[n]=PULSA;
               lcd.print("*");
               n=n+1;
                           }
                             }
                               }
        if(n==4);{
        if (clave==codigo_emergencia) {
              apagar=1;
              lcd.clear();
          lcd.print("PUERTA ABIERTA");
          digitalWrite (LEDverde, HIGH);
          puertaServo.writeMicroseconds(abierto);
             delay(1000);
          digitalWrite (LEDverde,LOW);
          puertaServo.writeMicroseconds(cerrado);
          lcd.clear();
         }

         else {

        if (salir==0){
         lcd.clear();
         lcd.print("CODIGO ERRONEO");



         digitalWrite(LEDrojoOUT,HIGH);
```

```
            delay(1000);
         digitalWrite(LEDrojoOUT,LOW);
         lcd.clear();
        }
         }
        }
         break;

         }
           }
             }
```

```
else {
  apagar=0;

     if (t>Tmin && t<Tmax && h>Hmin && h<Hmax){ //Funcionamiento mientras
la temperatura se encuentra entre los rangos óptimos

      //Se ponen los componentes que actuan como señal de aviso a 0, es
decir, estado LOW

      digitalWrite(LEDrojoOUT,LOW);
      digitalWrite(LEDrojoIN,LOW);
      digitalWrite (LEDverde,LOW);
      digitalWrite(Buzzer_OUT,LOW);
      digitalWrite(Buzzer_IN,LOW);

    //Código visualización temperatura y humedad en pantalla LCD

      lcd.setCursor(0,0);
      lcd.print("T= ");
      lcd.print(t);
      lcd.print("*C      ");
      lcd.setCursor(0,1);
      lcd.print("H= ");
      lcd.print(h);
      lcd.print("%       ");
```

```
       TECLA= teclado.getKey(); //Se prepara para recibir información del
keypad


        if (TECLA != 0){

          switch (TECLA){

         //Todos los códigos de los abonados tiene el formato A+4
dígitos, de esta manera al pulsar la tecla A activamos las instrucciones
para el control de acceso de abonados
         //El código del personal autorizado tiene el formato C+4
dígitos, al pulsar la C se activan las instrucciones correspondientes
             case 'A':

               contador=millis();  //Inicialización temporizador
               lcd.clear();
               lcd.setCursor(0,0);

               if (salir==1){//Salir del case
                break;
                salir=0;}
                 n=0;
               while (n<4){

                  tiempo_transcurrido=millis();
                  if (tiempo_transcurrido-contador > espera){//
                     salir=1;
                     break;
                     }

                  else{

                      PULSA=teclado.getKey();

                      if (PULSA!=0){

      clave[n]=PULSA;
        lcd.print(clave[n]);

        n=n+1;
```

```
      }

}
              }

        if (n==4){


     delay(500);
     lcd.clear();
      int i;
      int estado;
      estado=0;
       for (i=0;i<=7;i++){
        //Compara las claves guardadas con la introducida
        //Si la clave es correcta da paso, si no, lo deniega

        if (clave==claves_abonados[i])
              {
          estado = 1;
           }
             }
     if (estado==1){

        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print("PASE");

        digitalWrite (LEDverde, HIGH);
        puertaServo.writeMicroseconds(abierto);
        delay(1000);
        digitalWrite (LEDverde,LOW);
        puertaServo.writeMicroseconds(cerrado);
        lcd.clear();
        }

        else {

       if (salir==0){
        lcd.clear();
```

```
      lcd.setCursor(0,0);
      lcd.print("ACCESO DENEGADO");
            digitalWrite(LEDrojoOUT,HIGH);
      delay(1000);
      digitalWrite(LEDrojoOUT,LOW);
      lcd.clear();
     }
      }
      }

    break;

    case 'C':
      contador=millis(); //Inicio temporizador
    lcd.clear();
    if (salir==1){
    break;
    salir=0;
    }
          while (n<4){
tiempo_transcurrido=millis();
if (tiempo_transcurrido-contador > espera){
     salir=1;
     break;
}
else{

    PULSA=teclado.getKey();
}
    if (PULSA!=0){

    clave[n]=PULSA;
      lcd.print(clave[n]);

      n=n+1;
    }
}
if (clave==codigo_emergencia && n==4)
            {
       lcd.clear();
```

```
        lcd.setCursor (0,0);
        lcd.print("PUERTA ABIERTA");

      digitalWrite (LEDverde, HIGH);
      puertaServo.writeMicroseconds(abierto);
      delay(1000);
      digitalWrite (LEDverde,LOW);
      puertaServo.writeMicroseconds(cerrado);
      lcd.clear();
      }

      else if (salir==0 && n==4){

      lcd.clear();
      lcd.setCursor (0,0);
      lcd.print("CODIGO ERRONEO");

      digitalWrite(LEDrojoOUT,HIGH);
           delay(1000);
      digitalWrite(LEDrojoOUT,LOW);
      lcd.clear();
     }
      break;
      }
      }

  }

  //Instrucciones en caso de que la temperatura esté por debajo de la
recomendada o la humedad relativa esté fuera del rango de funcionamiento
  if (t<=Tmin || h<=Hmin || h>=Hmax){


  TECLA= teclado.getKey();
  if (TECLA != 0){

    switch (TECLA){
        case 'C':
          contador=millis();
    lcd.clear();
    if (salir==1){
```

```
    break;
    salir=0;}

     int n=0;
          while (n<4){
tiempo_transcurrido=millis();
if (tiempo_transcurrido-contador > espera){//
     salir=1;
     break;
}
else{

    PULSA=teclado.getKey();

    if (PULSA!=0){

    clave[n]=PULSA;
      lcd.print("*");

      n=n+1;
        }
          }
            }
            if(n==4){
if (clave==codigo_emergencia)
            {
              lcd.clear();
       lcd.print("PUERTA ABIERTA");

      digitalWrite (LEDverde, HIGH);
      puertaServo.writeMicroseconds(abierto);
      delay(1000);
      digitalWrite (LEDverde,LOW);
      puertaServo.writeMicroseconds(cerrado);
      lcd.clear();
      }

      else {

     if (salir==0){
```

```
        lcd.clear();
        lcd.print("CODIGO ERRONEO");

       digitalWrite(LEDrojoOUT,HIGH);
            delay(1000);
       digitalWrite(LEDrojoOUT,LOW);
       lcd.clear();
          }
            }
             }
       break;
       }
      }

   contajeMillis=millis();

 if(contajeMillis - previoMillis > intervalo){
   previoMillis=contajeMillis;
   if (estadoLED==LOW){
    pantalla=0;
     estadoLED=HIGH;}
     else{
       pantalla=1;
       estadoLED=LOW;}

 digitalWrite (LEDrojoIN,estadoLED);
 digitalWrite (LEDrojoOUT, estadoLED);

if (t<=Tmin){
switch (pantalla){
case 0:

  lcd.setCursor(0,0);
 lcd.print("T= ");
 lcd.print(t);
 lcd.print("*C     ");
 break;
 case 1:
  lcd.setCursor(0,0);
 lcd.print("                ");
```

```
 break;
    }
 }
 else{
     lcd.setCursor(0,0);
  lcd.print("T= ");
  lcd.print(t);
  lcd.print("*C     ");}


 if (h<=Hmin || h>=Hmax){
 switch (pantalla){
 case 0:
   lcd.setCursor(0,1);
  lcd.print("H= ");
  lcd.print(h);
  lcd.print("%       ");
  break;
  case 1:
   lcd.setCursor(0,1);
  lcd.print("                ");
  break;
    }
 }
 else{
  lcd.setCursor(0,1);
  lcd.print("H= ");
  lcd.print(h);
  lcd.print("%       ");

 }
 }
  }

if (t<Tmax){ //Resetea la variable que "apaga" los zumbadores para que
vuelva al estado inicial

apagar2=0;
}
//Instrucciones en caso de que la temperatura sobrepase el límite
superior
```

```
 if (t>=Tmax){
  lcd.setCursor(0,1);
 lcd.print("H= ");
 lcd.print(h);
 lcd.print("%       ");

      contajeMillis=millis();

 if(contajeMillis - previoMillis > intervalo){
   previoMillis=contajeMillis;
   if (estadoLED==LOW){
    pantalla=0;
     estadoLED=HIGH;}
     else{
       pantalla=1;
       estadoLED=LOW;}

 digitalWrite (LEDrojoIN,estadoLED);
 digitalWrite (LEDrojoOUT, estadoLED);
 if (apagar2!=1){
 digitalWrite(Buzzer_OUT,estadoLED);
digitalWrite(Buzzer_IN,estadoLED);
 }
else{
   digitalWrite(Buzzer_OUT,LOW);
digitalWrite(Buzzer_IN,LOW);
}
switch (pantalla){
case 0:

 lcd.setCursor(0,0);
 lcd.print("T=");
 lcd.print(t);
 lcd.print("*C");
 lcd.setCursor(12,0);
 lcd.print("NO   ");
 lcd.setCursor(10,1);
 lcd.print("PASAR  ");
 break;
 case 1:
```

```
lcd.setCursor(12,0);
lcd.print("NO   ");
lcd.setCursor(10,1);
lcd.print("PASAR  ");
 break;
  }
}
int n=0;

TECLA= teclado.getKey();
if (TECLA != 0){

  switch (TECLA){
      case 'C':
       digitalWrite(Buzzer_OUT,LOW);
        digitalWrite(Buzzer_IN,LOW);

      contador=millis();
      lcd.clear();
      if (salir==1){
      break;
      salir=0;}

    while(n<4){

     tiempo_transcurrido=millis();
if (tiempo_transcurrido-contador > espera){
     salir=1;
     break;
}
else{

    PULSA=teclado.getKey();
}
    if (PULSA!=0){

    clave[n]=PULSA;
      lcd.print("*");
```

```
        n=n+1;
      }
        }

        if(n==4){
        if (clave==codigo_emergencia){
          lcd.clear();
          lcd.print ("PUERTA ABIERTA");

        apagar2=1;
         digitalWrite(Buzzer_OUT,LOW);
digitalWrite(Buzzer_IN,LOW);

       digitalWrite (LEDverde, HIGH);
            puertaServo.writeMicroseconds(abierto);
       delay(1000);
        digitalWrite (LEDverde, LOW);
             puertaServo.writeMicroseconds(cerrado);
              lcd.clear();
               }

        else if(salir==0){
          lcd.clear();
          lcd.setCursor(0,0);
          lcd.print("Codigo erroneo");

         digitalWrite(LEDrojoOUT,HIGH);
     delay(1000);
     digitalWrite(LEDrojoOUT,LOW);
     lcd.clear();
        }
          }

      break;
    }
   }
  }
 }
}
```