# UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

# FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

## DESARROLLO DE UN RADIÓMETRO PORTÁTIL PARA LA MEDICIÓN DEL ÍNDICE DE RADIACIÓN ULTRAVIOLETA CON BITÁCORA DE LECTURAS

## T E S I S

Que para obtener el título de:

**INGENIERO ELÉCTRICO ELECTRÓNICO**

Presenta:

**JESÚS MARTÍN ESCALONA CAMARILLO**

Director de tesis:

**MTRO. ALEJANDRO ANDRÉS SERAPIO CARMONA**

**Ciudad Nezahualcóyotl, Estado de México, 2021**

# Agradecimientos

A Dios por la vida y la hermosa familia que me dio.

A todos mis seres queridos, en especial a mi madre, Francisca Camarillo Vázquez, el tesoro más grande y preciado que tengo en la vida, por siempre estar a mi lado dándome ánimo para seguir adelante, por todo su cariño y confianza y a mi Padre y amigo, Efrén Martin Escalona Cornejo, politólogo y Maestro ejemplar, que me llevó por el camino del bien y me dio la mejor herencia: buenos valores y estudios profesionales, quien estoy seguro me cuida desde el cielo.

A mis hermanos, Yeslén y Geovani, que de una y otra forma me brindan su sincera amistad, apoyándome en todo lo que les es posible.

A una persona muy importante en mi vida, Areli Jaqueline López Gómez, por su compañía incondicional, por motivarme siempre a ser mejor y por formar parte de este triunfo.

A todos los docentes que me guiaron en el proceso para culminar mi carrera universitaria, pero especialmente a mi asesor, por darme la oportunidad de desarrollar este trabajo, por su infinito apoyo, paciencia y dedicación.

# Contenido

# Índice de tablas

# 1 Introducción

La vitamina D es un nutriente necesario para la salud, mayormente se sabe que ayuda a mantener los huesos fuertes. Para ello, logra que el cuerpo absorba el calcio y fósforo (una de las piezas fundamentales de los huesos) de los alimentos y suplementos (Nacional Institutes of Health, 2019). Las personas que consumen dicha vitamina en cantidades muy escasas pueden tener huesos delgados y frágiles, un trastorno al que se le denomina raquitismo en los niños y osteomalacia en los adultos. Esto provoca que los huesos lleguen a doblarse o fracturarse, retraso en el crecimiento, piernas arqueadas, debilidad y dolor en la columna vertebral, la pelvis y las piernas (U.S.A. National Library of Medicine, 2019).

Además, este nutriente es muy importante para el cuerpo de muchas otras formas. Los músculos requieren esta vitamina para el movimiento. Por ejemplo, los nervios la necesitan para transmitir mensajes entre el cerebro y cada parte del cuerpo, y el sistema inmunológico la emplea para combatir los virus y bacterias que lo invaden. Junto con el calcio, la vitamina D ayuda a proteger a los adultos mayores contra la osteoporosis, en otras palabras, se encuentra en las células de todo el cuerpo, puede proteger contra diversas afecciones de salud, como algunos tipos de cáncer, debilidad muscular, trastornos del estado de ánimo, diabetes, enfermedad renal, enfermedades del corazón y presión arterial alta (American Academy of Family Physicians, 2017).

Como se puede entender, es de suma importancia entregarle al cuerpo los niveles requeridos de dicha vitamina, y para ello se deben consumir los alimentos que la poseen, como lo son los pescados grasos (salmón, el atún y la caballa), hígado de res, lácteos, hongos, yema de huevo, cereales y jugo de naranja. Pero otra gran fuente de vitamina D (el 90%), es la exposición directa a la luz del sol, pues posterior a someter nuestra piel a la radiación ultravioleta que brinda el astro, el cuerpo produce de manera natural la mencionada vitamina (U.S. National Library of Medicine, 2019).

El M. en C. Jorge Maldonado Hernández, responsable del Laboratorio de Espectrometría de Masas en el Hospital de Pediatría Centro Médico Nacional Siglo XXI del IMSS, y su equipo de trabajo realizan una investigación acerca del tema, y menciona en sus primeros informes

que para los tratamientos dirigidos a la deficiencia de vitamina D, forzosamente será requerida la exposición a la radiación UV, dependiendo de la gravedad variará el tiempo de dicha exposición, ya que el exceso puede provocar otras afectaciones como lo es el envejecimiento prematuro o cáncer de piel, un problema que causa temor entre la sociedad, lo que conlleva a muchos pacientes a no seguir la instrucción del médico. El método empleado en su investigación para realizar un registro de dicha rutina es llenando a lápiz un formato donde se establecen los horarios iniciales y finales de las exposiciones con su respectiva fecha, quedando como trabajo para los investigadores, buscar el índice de radiación UV más próximo a la zona determinada, brindado por una estación meteorológica cercana al sitio donde reside cada paciente, dándose así la idea únicamente de una aproximación. Ante esta situación, el equipo del doctor Maldonado, solicitó el apoyo del Grupo IDEA de la carrera de Ingeniería Eléctrica Electrónica de la Facultad de Estudios Superiores Aragón, sumarse al equipo desarrollando un radiómetro del índice de radiación ultravioleta portátil y con la capacidad de generar de manera automática una bitácora, para proporcionarse a los participantes del estudio.

## 1.1 Objetivo general

Diseñar e implementar un dispositivo que realice mediciones del índice de radiación ultravioleta, y que genere una bitácora con nivel de radiación, fecha y hora.

## 1.2 Actividades

- Estudiar el tema de la radiación ultravioleta.
- Investigar los diversos sensores que miden el nivel de radiación UV para determinar el más conveniente.
- Familiarizarse con los módulos para memoria microSD y las diversas tareas que se pueden realizar con ellos.
- Aprender a utilizar los relojes de tiempo real, ya que serán necesarios para el registro de horarios y fechas.
- Identificar las diversas pantallas con las que se puede trabajar en el intercambio de información con el microcontrolador.
- Indagar los diversos tipos de botones e interruptores.

- Determinar un sistema de control eficiente para aplicarse en un dispositivo con las características que requerimos.
- Aprender a utilizar algún software de diseño 3D para desarrollar la fabricación de la carcasa que protegerá el radiómetro.

## 1.3 Objetivos particulares

- Mediciones del índice de radiación ultravioleta fiables.
- Restringir mediante programación horario de uso para toma de mediciones.
- Guardar los registros de mediciones, fecha y hora en una MicroSD extraíble.
- Incorporar una pantalla LCD informativa de estatus del radiómetro.
- Integrar alertas (audibles y sensibles) de errores en algunos componentes del sistema (RTC y MicroSD).
- Incluir una batería recargable de alta capacidad con un control de carga.
- Contar con una carcasa protectora.

## 1.4 Justificación

El proyecto titulado "Determinantes sociodemográficos, ambientales y genéticos de la deficiencia de 25-hidroxivitamina-D en adultos de la Ciudad de México", desarrollado por el M. en C. Jorge Maldonado Hernández, Investigador Asociado D. Laboratorio de Espectrometría de Masas. Centro de Instrumentos, Centro Médico Nacional Siglo XXI, del IMSS, requiere la fabricación de un instrumento de medición de radiación UV-B con las características previamente mencionadas, por ello solicitaron su diseño y fabricación.

## 1.5 Descripción del capitulado

Capítulo 2. Se brinda la información teórica más importante que se necesita para entender el funcionamiento del radiómetro, considerando pertinente conocer el tema de la radiación ultravioleta y sus características. Además, se describen los diversos componentes electrónicos que se requieren para el diseño de dicho dispositivo.

Capítulo 3. Se describe a detalle la metodología a seguir para la elaboración del radiómetro portátil, tanto en software como en hardware, para lograr cumplir los objetivos necesarios.

Capítulo 4. En este capítulo se reportan las pruebas que se hicieron al dispositivo en el proceso de su fabricación y de igual manera cuando se tuvo un prototipo final para presentar los resultados obtenidos, se brindan también las conclusiones a partir de los diversos objetivos requeridos.

# 2 Marco teórico

## 2.1 La radiación solar

La radiación solar es la energía radiante emitida en el espacio interplanetario del sol (ilustración 2.1). Esta radiación se genera a partir de las reacciones termonucleares de fusión que se producen en el núcleo solar y que producen la radiación electromagnética en varias frecuencias o longitudes de onda, que se propaga entonces en el espacio a las velocidades típicas de estas olas. Esta propagación permite llevar energía solar con ellas (Planas, 2017).

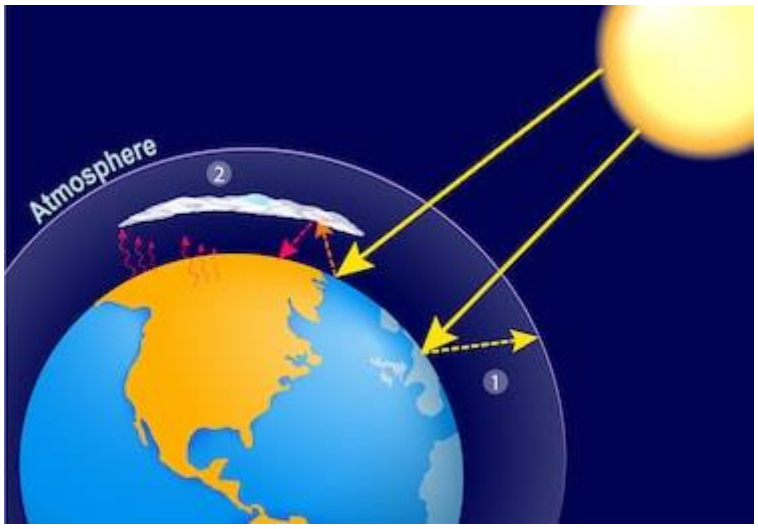



*Ilustración 2.1 Radiación solar.*

La constante solar es la cantidad de energía recibida en forma de radiación solar por unidad de tiempo y unidad de superficie, medida en la parte externa de la atmósfera terrestre en un plano perpendicular a los rayos del Sol.

La energía solar y consecuentemente la radiación solar resulta del proceso de fusión nuclear que tiene lugar en el sol. Esta energía es la principal fuente energética y, por lo tanto, el motor que mueve al medio ambiente. La energía solar que recibimos mediante la radiación solar es responsable directa o indirectamente de aspectos importantes para la vida como la fotosíntesis, el aprovechamiento de la vitamina D presente en los alimentos para los seres humanos, el mantenimiento de una temperatura del planeta compatible con la vida, del viento, etc. La energía solar que llega a la superficie terrestre es 10,000 veces mayor que la energía consumida actualmente por toda la humanidad.

### *2.1.1* Características de la radiación solar

La radiación solar no se concentra en una sola frecuencia, sino que se distribuye en un amplio espectro de amplitud no uniforme con la forma típica de una campana, como es típico del espectro de un cuerpo negro con el que se modela la fuente solar. El máximo de radiación se centra en la banda de radiación o luz visible con un pico a 500 nm fuera de la atmósfera terrestre según la ley de Wien, Además de la radiación visible, un componente energéticamente minoritario, pero sin embargo digno de mención por sus efectos es el infrarrojo y, sobre todo, los rayos ultravioletas.

### 2.1.2 Los rayos ultravioletas

Al cruzar la atmósfera la radiación solar se somete a fenómenos de reflexión, refracción, absorción y difusión por los diversos gases atmosféricos en un grado variable en función de la frecuencia, de modo que el suelo del espectro solar es irregular en comparación con la detectada en los umbrales externas de atmósfera (TOA) con presencia de bandas típicas de absorción o reflexión.

La radiación ultravioleta se divide en tres tipos:

- Los **rayos UV-A** envejecen a las células de la piel y pueden dañar el ADN de estas células cuando su presencia es excesiva. La mayoría de las camas bronceadoras emiten grandes cantidades de UVA que según se ha descubierto aumentan el riesgo de cáncer de piel.
- Los **rayos UV-B** (ilustración 2.2) tienen un poco más de energía que los rayos UVA, se cree que causan la mayoría de los cánceres de piel cuando se es sometido en exceso a su energía, pero no podemos evitar someternos a su exposición, ya que como se ha mencionado es la principal fuente de vitamina D, y su ausencia causa graves problemas de salud al ser humano.
- Los **rayos UV-C** tienen más energía que otros tipos de rayos UV, pero no penetran nuestra atmósfera y no están en la luz solar. No son normalmente una causa de cáncer de piel.

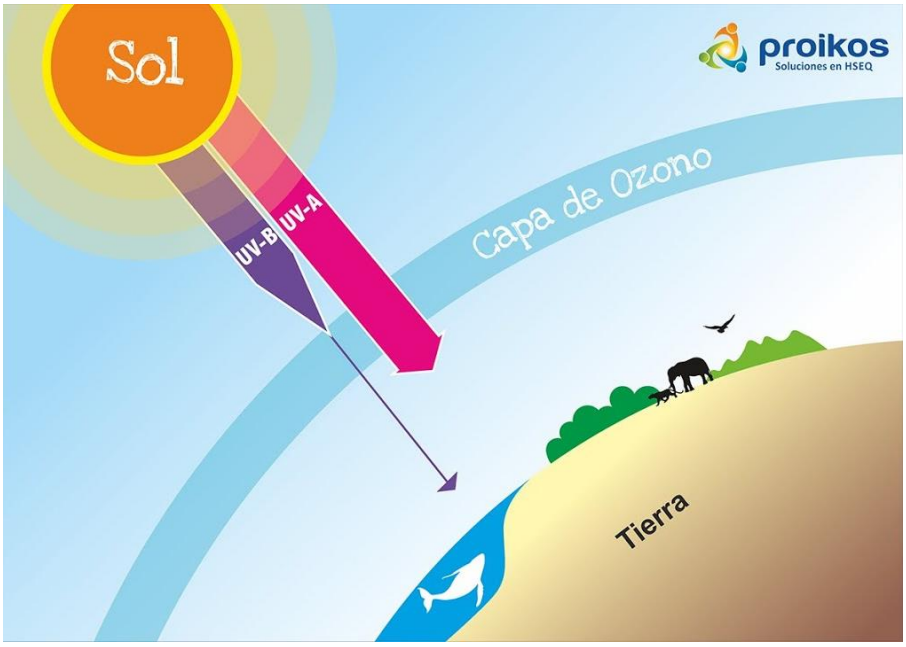



*Ilustración 2.2 Radiación UV.*

### 2.1.3 Características de la radiación UV

La potencia de los rayos UV (ilustración 2.3) que llegan al suelo depende de un número de factores, tales como:

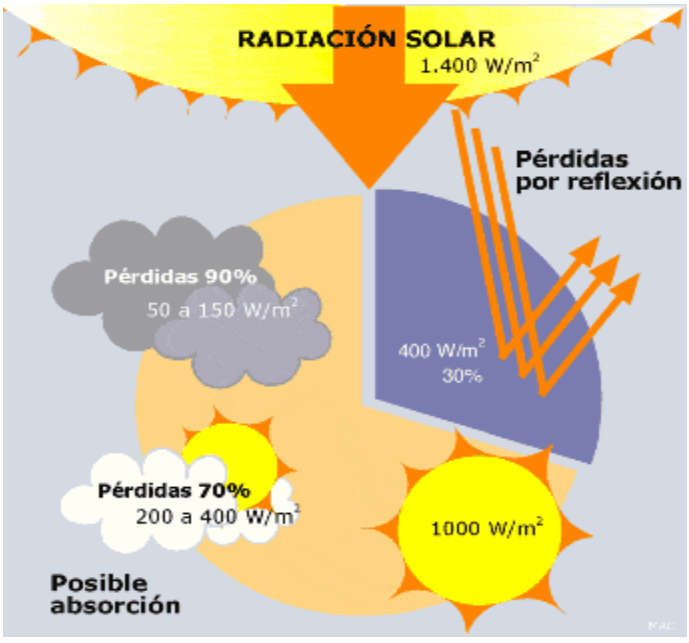



*Ilustración 2.3 Potencia de los rayos UV.*

- Hora del día: los rayos UV son más potentes entre 10 a.m. y 4 p.m.

- Temporada del año: los rayos UV son más potentes durante los meses de la primavera y el verano. Este es un factor menos importante cerca del ecuador.
- Distancia desde el ecuador (latitud): la exposición a UV disminuye a medida que se aleja de la línea ecuatorial.
- Altitud: más rayos UV llegan al suelo en elevaciones más altas.
- Formación nubosa: el efecto de las nubes puede variar, ya que a veces la formación nubosa bloquea a algunos rayos UV del sol y reduce la exposición a rayos UV, mientras que algunos tipos de nubes pueden reflejar los rayos UV y pueden aumentar la exposición a los rayos UV. Lo que es importante saber es que los rayos UV pueden atravesar las nubes, incluso en un día nublado.
- Reflejo de las superficies: los rayos UV pueden rebotar en superficies como el agua, la arena, la nieve, el pavimento, o la hierba, lo que lleva a un aumento en la exposición a los rayos UV.
- El grado de exposición a la luz ultravioleta que una persona recibe, depende de la intensidad de los rayos, del tiempo que la piel ha estado expuesta, y de si ésta ha estado protegida con ropa o bloqueador solar (Anónimo, American Cancer Society, 2019).

### 2.1.4 **Índice de radiación UV**

La OMS define una escala para el Índice UV que va de 1 a 11+, sin embargo, en algunas ciudades, incluyendo la Ciudad de México, se utiliza el valor de 0 para referirse a la ausencia de radiación. El valor de 11+ se utiliza para expresar un índice de 11 o superior. Cuando el Índice UV alcanza o supera el valor de 11 existe un riesgo importante de sufrir daños en la piel sin protección en un periodo de tiempo breve. En la Ciudad de México el Índice puede alcanzar un valor máximo equivalente a 15, sin embargo, se reporta como 11+ en apego a las recomendaciones de la OMS.

### 2.1.5 Características de la radiación UV-B en México

La Ciudad de México se encuentra en una latitud que le permite recibir la radiación del sol durante todo el año, además por su altitud está expuesta a un 20% más de radiación ultravioleta con respecto al nivel del mar. Para una exposición saludable es importante conocer la intensidad de la radiación solar, el tipo de piel y los daños que provoca. El Sistema

de Monitoreo Atmosférico (SIMAT) de la Ciudad de México tiene un programa continuo de monitoreo de los niveles de radiación solar ultravioleta, que se difunden cada hora como un índice de radiación solar UV (Índice UV o IUV), el cual utiliza las recomendaciones establecidas por la Organización Mundial de la Salud (OMS) (Anónimo, Índice de Radiación Ultravioleta, 2019).

## 2.2 Sensores

El término sensor (ilustración 2.4) se refiere a un elemento de medición que detecta la magnitud de un parámetro físico, y lo cambia por una señal que se puede procesar en el sistema con el que vamos a desarrollar el control. Al elemento activo de un sensor, se le conoce comúnmente como transductor. El diseño de sensores y transductores siempre involucra alguna ley o principio físico o químico, que relaciona la cantidad de interés con algún evento medible, comúnmente siendo características eléctricas que pueden ser comparadas y medidas por cualquier microcontrolador o microprocesador, como lo es el voltaje, corriente y resistencia (Anónimo, MecatrónicaLATAM, 2020).

*Ilustración 2.4 Sensores.*

### 2.2.1 Tipos de sensores

#### 2.2.1.1 Sensores inteligentes

Algunos sensores disponen un módulo especial para su acondicionamiento de señal, por lo tanto, es posible tener el sensor y el acondicionamiento de señal combinados con un microprocesador en el mismo paquete. Un estándar para los sensores inteligentes es el IEEE 1451.

#### 2.2.1.2 Sensores pasivos

Son aquellos que generan señales de la magnitud a medir, pero requieren una fuente auxiliar para funcionar.

#### 2.2.1.3 Sensores activos

Son aquellos que generan señales de la magnitud a medir de manera autónoma, es decir, no requiere de una fuente auxiliar.

Los sistemas de monitorización y control requieren sensores para medir cantidades físicas tales como posición lineal, posición angular, desplazamiento, deformación, aceleración, presión, caudal, fuerza, velocidad lineal y velocidad angular, temperatura, intensidad lumínica, distancia y vibración.

### 2.2.2 Sensores de medición de radiación ultravioleta

#### 2.2.2.1 Módulo ML8511 Detector UV

El módulo ML8511 (ilustración 2.5) es un sensor de luz ultravioleta (UV), entrega una señal analógica que depende de la cantidad de luz UV que detecta un fotodiodo (Ilustración 2.6) en conjunto con un amplificador operacional interno incorporado, así es posible convertir la fotocorriente a salida de voltaje dependiendo de la intensidad de la luz UV. Es usado en proyectos de monitoreo de condiciones ambientales como el índice UV, en aplicaciones Meteorológicas, cuidado de la piel, medición industrial de nivel UV, etc.

El sensor ML8511 detecta luz con una longitud de onda entre 280-390nm, este rango cubre tanto al espectro UV-B como al UV-A. La salida analógica está relacionada linealmente con la intensidad UV (mW/cm2). Esta señal analógica puede ser conectada a un microcontrolador para ser convertido por un ADC (Convertidor Analógico – Digital) y así trabajar con la medición tal como se puede observar en la ilustración 2.7 (LAPIS Semiconductor, 2013).

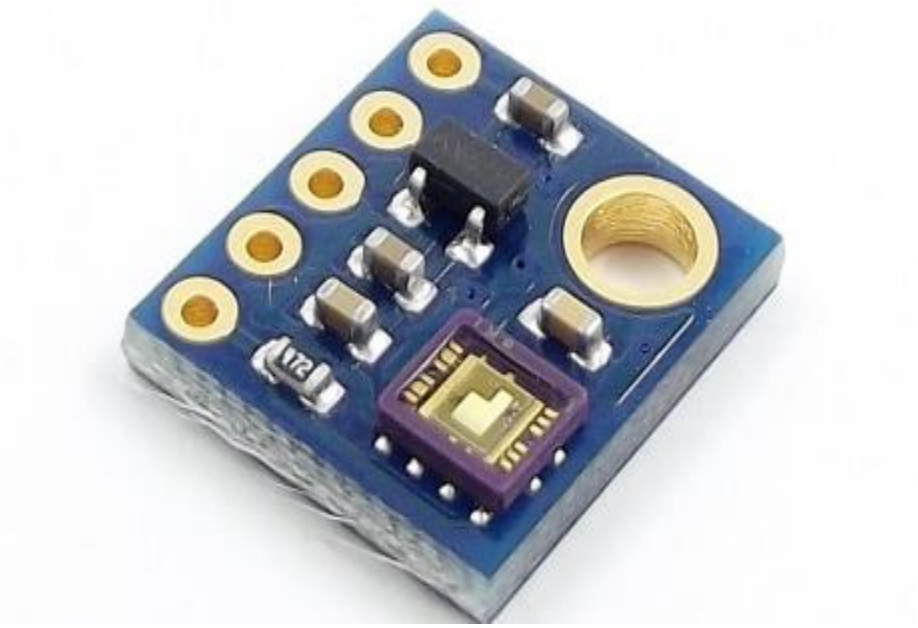


*Ilustración 2.5 ML8511*.

Entre sus especificaciones eléctricas encontramos:

- Voltaje de Operación: 5 $V_{DC}$.
- Salida analógica.
- Longitud de onda: 280-390nm.
- Consumo ultra bajo de energía (Anónimo, Naylamp mechatronics, 2020).

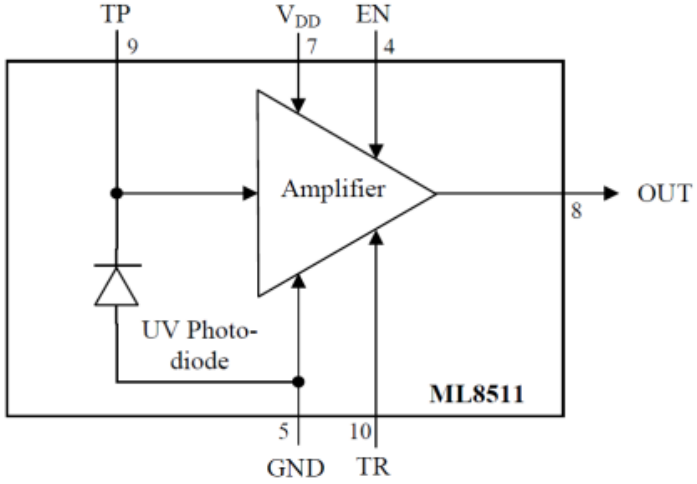



*Ilustración 2.6 Fotodiodo del ML8511*.

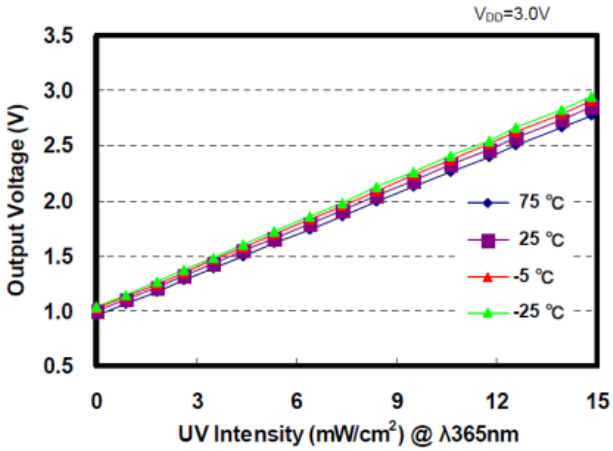



*Ilustración 2.7 Gráfica del ML8511.*

#### 2.2.2.2 Sensor UV GUVA-S12SD

El módulo de medición de radiación ultravioleta (ilustración 2.8), basado en el sensor GUVA S12SD permite medir la radiación a partir de su salida analógica. Este valor puede traducirse en un índice UV (Anónimo, Robótica fácil, 2017).

Descripción:

- Voltaje De Funcionamiento: DC 3.3-5 V.
- Tensión de Salida (analógica): DC 0-1 V.
- Precisión: ± 1 Índice UV.
- Current:0.06mA (típico).
- Longitud de onda: 200nm-370nm.
- Temperatura de trabajo: -20~ 85 ºC.
- Tamaño:19.80*15mm.

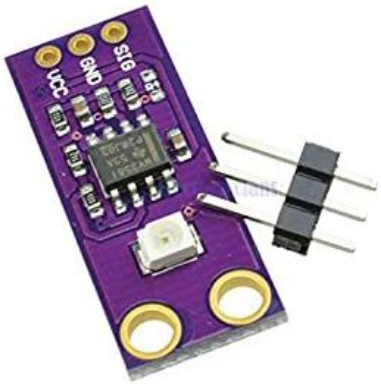



*Ilustración 2.8 GUVA-S12SD.*

#### 2.2.2.3 Módulo SEN_0777 detector de luz UV

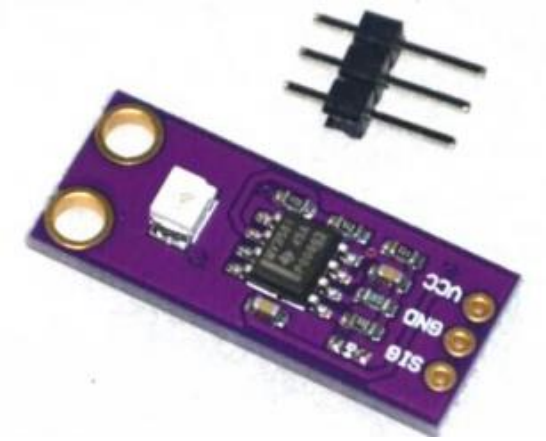



*Ilustración 2.9 SEN_0777.*

Este sensor UV (ilustración 2.9), se utiliza para detectar el índice de intensidad ultravioleta (UV). Esta forma de radiación electromagnética, tiene longitudes de onda más cortas que la radiación visible, y son estas longitudes cortas las que detecta este sensor.

Este módulo se basa en el sensor UVM-30A, que tiene una amplia gama espectral de 200nm hasta 370nm*. La señal eléctrica de salida del módulo, es de tipo analógica, que varía respecto a la intensidad de los rayos UV, lo que nos permite darnos una sugerencia, si es una buena idea o no ir a la playa hoy.

* El nanómetro (nm), es la unidad de longitud que equivale a unos mil millonésimos partes de un metro (1 nm = $10^{-9}$ m). Se utiliza para medir la longitud de onda de la radiación ultravioleta, radiación infrarroja y la luz.

Características:

- Voltaje de funcionamiento: 3 ~ 5 $V_{DC}$.
- Corriente: 0.06mA (Standard) /0.1mA (Max).
- Respuesta de Longitud de onda: 200 ~ 370 nm.
- Temperatura de trabajo: -20 ~ 85℃ (Anónimo, tdrobótica.co, 2020).

## 2.3 Memorias extraíbles

En el campo del almacenamiento de datos, los medios extraíbles son aquellos soportes de almacenamiento diseñados para ser extraídos del dispositivo empleado sin tener que apagarlo. Este tipo de medios extraíbles están diseñados para ser leídos por el dispositivo con el que se está trabajando, como podría ser una computadora, un celular, estéreo, microcontrolador, diversos aparatos multimedia, etc., además de poder escribir sobre él (FANDOM, 2020).

### 2.3.1 Tipos de memorias extraíbles

A continuación, se enlistan algunos ejemplos de este tipo de medios:

- Disco óptico.
- Disco compacto (CD, DVD, Blu-ray).
- Disquetes, discos Zip.
- Cintas magnéticas.
- Tarjeta perforada, cinta perforada.
- Memorias USB.
- Discos duros externos.

Y actualmente las memorias más populares por su reducido tamaño, fácil manejo y bajo costo son las tarjetas de memoria, como SD y sus variantes.

### 2.3.2 Memoria microSD

MicroSD (ilustración 2.10) es un formato de memoria flash desarrollado por SanDisk en julio de 2005 y estandarizado por la Asociación de Tarjetas SD. Un dispositivo de almacenamiento externo y extraíble tremendamente popular que se extiende por decenas de millones de unidades en múltiples productos profesionales y de consumo.

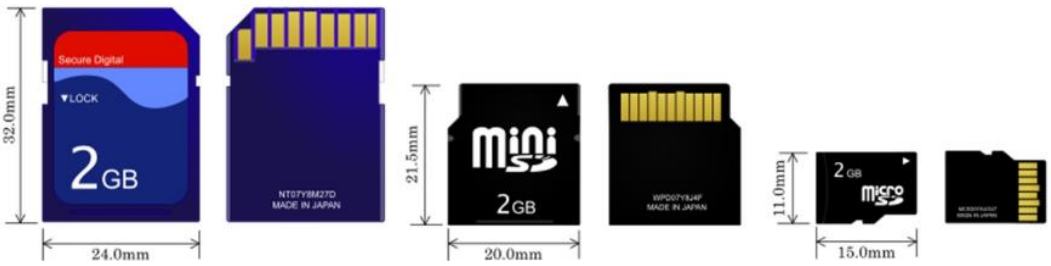



*Ilustración 2.10 Memoria microSD.*

Esta tarjeta de memoria es la variante más pequeña en tamaño de la especificación SD, que incluye estas mismas SD genéricas y las miniSD. Su tamaño reducido a 15×11×1 mm ha permitido incluirlas en todo tipo de dispositivos.

Su lanzamiento coincidió con la explosión de ventas de móviles inteligentes y aunque hoy algunos terminales ya no las soportan, su utilización es masiva y en algunos casos son imprescindibles para aumentar la capacidad de almacenamiento interno del dispositivo.

#### 2.3.2.1 Formatos microSD

Igual que la norma general de tarjetas SD se divide en tres formatos (SD, SDHC y SDXC) las tarjetas microSD se han comercializado en estas variantes. Lo primero a considerar en la adquisición es la compatibilidad con el dispositivo, en ocasiones no son compatibles todos los formatos de memoria con todos los dispositivos.

**microSD**: La más antigua. Tienen una capacidad de hasta 2 gigabyte (GB) y se pueden utilizar en cualquier ranura microSD.

**microSDHC**: Tienen una capacidad superior a los 2 GB y hasta 32 GB. se puede utilizar en dispositivos que soporten SDHC y SDXC.

**microSDXC**: Son las más modernas y la adquisición solo debe realizarse siempre que el dispositivo la soporte, porque solo son compatibles con hardware SDXC. Su capacidad varía desde 32 Gbytes a 2 TBytes. Es el máximo teórico de la norma, si bien, hasta ahora, el máximo ofertado por la industria es de 512 Gbytes.

#### 2.3.2.2 Rendimiento – Clase

Los responsables de la especificación se empeñaron en complicar las cosas, a la hora de elegir la tarjeta tenemos que tener en cuenta que, el rendimiento sea suficiente para el uso al que vamos a destinarla. Afortunadamente es fácil de distinguir, con base en un indicador presente en la memoria, ésta formado por unos números que se encuentran dentro de un semicírculo, lo cual designa la “Clase”. Generalmente viene impreso en todas las tarjetas que podemos adquirir.

El significado de esos números no reside en el rendimiento real o máximo de la memoria, sino en la velocidad mínima en Megabytes (MB) por segundo que en modo escritura es capaz de soportar la tarjeta. La velocidad mínima en escritura soportada de cada clase es:

- Clase 2: Al menos 2 Mbps.
- Clase 4: Al menos 4 Mbps.
- Clase 6: Al menos 6 Mbps.
- Clase 10: Al menos 10 Mbps.

### 2.3.3 Módulo de memoria microSD (Ilustración 2.11)

Este módulo nos permite insertar una memoria Micro SD que son las más comunes en el mercado, el módulo se puede alimentar con 3.3 V o 5 V usando los pines respectivos.

La comunicación de la memoria es por SPI, pero trabajan con 3.3 V, para utilizarlo con Arduino necesitamos módulos externos que aparte de tener el socket traen los componentes necesarios para adaptar los voltajes a TTL y poder conectarlo de forma fácil a al microcontrolador (Anónimo, Naylamp Mechatronics SAC., 2020).

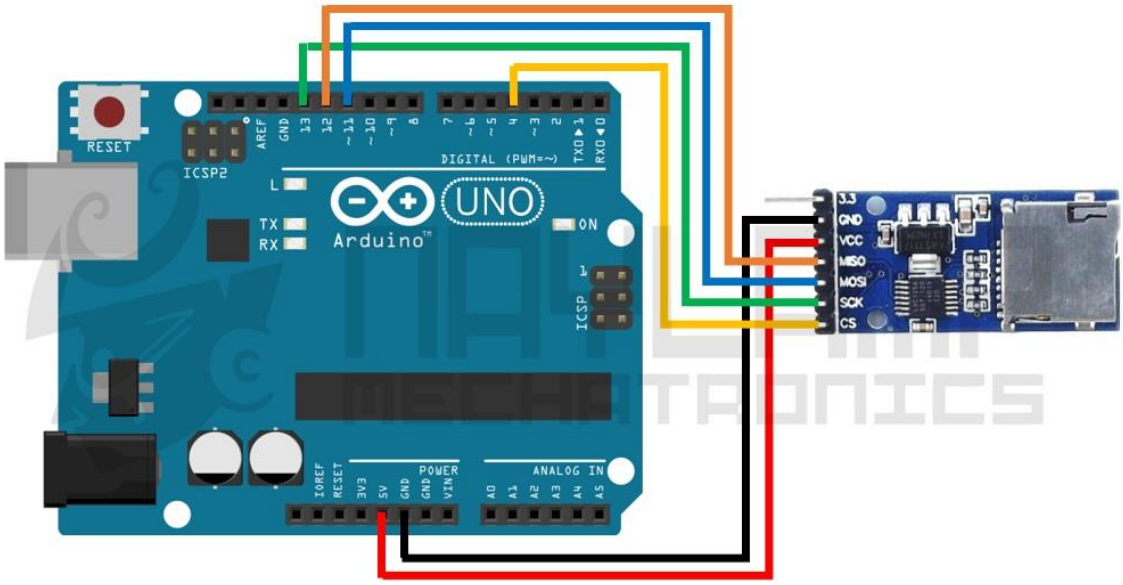



*Ilustración 2.11 Módulo de memoria micro SD.*

Arduino posee una biblioteca para usar este tipo de memorias, funciona con los dos módulos existentes en el mercado (para memorias SD y micro SD). La biblioteca ya viene incluida en el IDE de Arduino, por lo que no necesitamos instalar ni descargar nada, únicamente requerimos incluir la biblioteca SD al inicio del código:

```
#include<SD.h>
```

Así podemos realizar numerosas y variadas acciones con la memoria, es posible escribir y leer datos, crear y borrar archivos, etc. En la página oficial de Arduino podemos encontrar diferentes ejemplos como:

- SD.begin(cspin)

Inicializa la biblioteca SD y la tarjeta, como parámetro se le indica el pin CS al que está conectado el módulo, si no se especifica *cspin* (terminal en la que se conectara la entrada "chip select"), se usa el valor por defecto del CS por hardware. Los demás pines deben estar conectados a las terminales asociadas al protocolo de comunicación SPI de la tarjeta Arduino.

- SD.exists(filename)

Comprueba si existe el archivo especificado, *filename* es el nombre del archivo y/o directorio en la tarjeta SD si este existe la función nos retorna un true, de lo contrario retorna false.

- SD.mkdir(directory)

Crea el directorio especificado, si los subdirectorios no existen, también se crearán. Por ejemplo: *SD.mkdir*("Arduino/proyecto1/archivos), crea la carpeta "archivos" y si las carpetas Arduino y proyecto1 no existen, entonces también se crearan. La función retorna true si la creación del directorio fue exitosa de lo contrario nos retorna un false.

- SD.remove(filename)

Elimina el archivo (filename) de la tarjeta SD, se debe de incluir el directorio. Solo elimina el archivo más no el directorio. Devuelve true se logra eliminar el archivo de lo contrario nos retorna un false.

- SD.rmdir(dirname)

Eliminar el directorio (dirname) de la tarjeta SD. El directorio debe estar vacío. Devuelve TRUE si la eliminación del directorio tuvo éxito o FALSE en caso contrario.

- SD.open(filepath, mode)

Abre el archivo especificado y se debe de incluir el directorio si el archivo está en carpetas. Si el archivo no existe, se creará un archivo con el nombre especificado, pero no será posible crear el directorio si este no existe. Se puede abrir un archivo como solo lectura (si mode es

FILE_READ) o como lectura y escritura (si mode es FILE_WRITE), el modo por defecto en caso no se especifique es FILE_READ.

Esta función retorna un objeto tipo FILE, el cual es necesario declararlo antes como una variable. Por ejemplo:

- File myFile;

```
myFile = SD.open("archivo.txt", FILE_WRITE);
```

Funciones de la clase File:

- file.available()

Comprueba si hay bytes disponibles para leer en el archivo y retorna el número de bytes disponibles

- file.read()

Lee un byte de la variable File (archivo abierto anteriormente con SD.open())

- file.write(data)

Escribe un byte en el archivo, el archivo debe estar abierto en modo lectura y escritura. Usando file.write(buf, len) se puede escribir un array de byte (buf) pero se debe especificar el tamaño (len).

- file.print(data)

Esta función tiene las mismas características que un Serial.print(); data puede ser una variable o texto, el cual será enviado como caracteres. Si queremos agregar al final un salto o nueva línea se usa file.println(data)

- file.size()

Retorna el tamaño en bytes del archivo

- file.position()

Retorna la posición actual en donde se leerá o escribirá el siguiente byte.

- file.seek(pos)

Ubica en una posición específica el archivo. “Pos” debe ser un número entre 0 y el tamaño en bytes del archivo

- file.close()

Cierra el archivo, y en ese instante los datos se guardan en la memoria SD, así se puede extraer de forma segura la memoria SD.

## 2.4 Relojes de tiempo real (RTC)

Un RTC (“Real Time Clock” por sus siglas en inglés) es un dispositivo electrónico que permite obtener mediciones de tiempo en las unidades temporales que empleamos de forma cotidiana.

El término RTC se creó para diferenciar este tipo de relojes de los relojes electrónicos habituales, que simplemente miden el tiempo contabilizando pulsos de una señal, sin existir relación directa con unidades temporales.

Por el contrario, los RTC son más parecidos a los relojes y calendarios que usamos habitualmente, ya que funcionan con segundos, minutos, horas, días, semanas, meses y años.

Los RTC normalmente están formados por un resonador de cristal integrado con la electrónica necesaria para contabilizar de forma correcta el paso del tiempo. La electrónica de los RTC tiene en cuenta las peculiaridades de nuestra forma de medir el tiempo, como por ejemplo el sistema sexagesimal, los meses con diferentes días, o los años bisiestos.

Los RTC aportan la ventaja de reducir el consumo de energía, aportar mayor precisión y liberar al microcontrolador o microprocesador de tener que realizar la contabilización del tiempo. Además, frecuentemente los RTC incorporan algún tipo de batería que permite mantener el valor del tiempo en caso de pérdida de alimentación.

### 2.4.1 Relojes de tiempo real más comunes

En el mercado existen dos RTC habituales, el DS1307 mostrado en la ilustración 2.12 y el DS3231 de la ilustración 2.13, ambos fabricados por Maxim (anteriormente Dallas

Semiconductor). El DS3231 tiene una precisión muy superior y puede considerarse sustituto del DS1307.

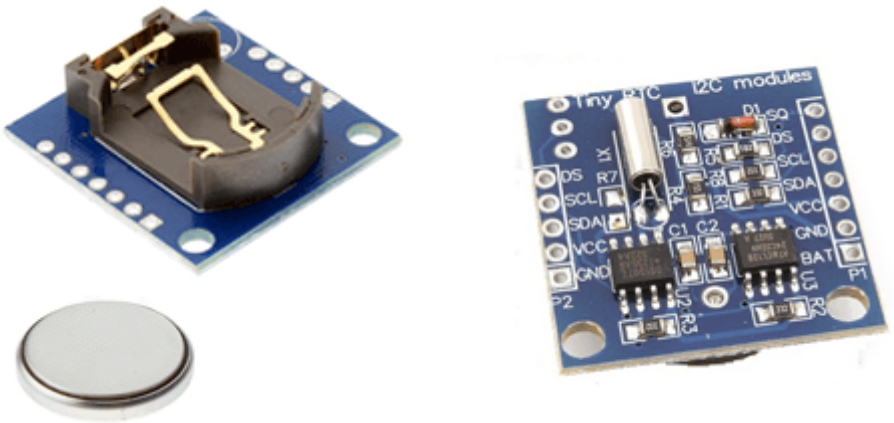



*Ilustración 2.12 DS1307*

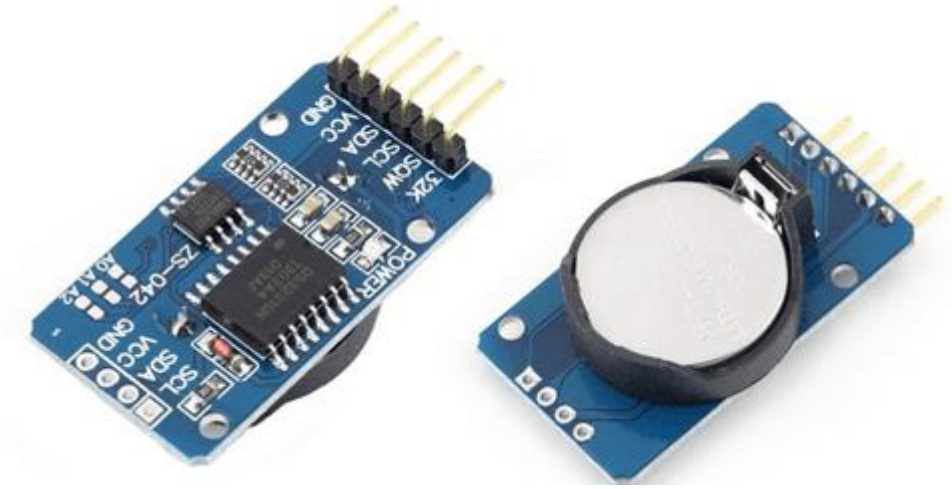



*Ilustración 2.13 DS3231*

En el modelo DS1307 las variaciones de temperatura afectan a la medición del tiempo de los cristales resonadores, lo cual, se traduce en errores de desfase acumulado. Esto hace que el DS1307 sufra de un desfase temporal, que puede llegar a ser 1 o 2 minutos al día.

Para solucionarlo, el DS3231 incorpora medición y compensación de la temperatura garantizando una precisión de al menos 2ppm, lo que equivale a un desfase máximo 172 ms/día o un segundo cada 6 días. En el mundo real normalmente consiguen precisiones superiores, equivalente a desfases de 1-2 segundos al mes.

La comunicación en ambos modelos se realiza a través del bus I2C, por lo que es sencillo obtener los datos medidos. La tensión de alimentación es 4.5 V a 5.5 V para el DS1307, y 2.3 a 5.5 V para el DS3231.

Frecuentemente estos módulos también incorporan una pequeña EEPROM AT24C32, que puede ser empleada para almacenar registros y mediciones. En el caso del DS3231, la medición de temperatura también está disponible, aunque tiene una precisión baja ±3ºC, y el tiempo de adquisición puede demorar hasta 1 segundo.

También incorporan una batería CR2032 para mantener el dispositivo a la hora al retirar la alimentación. Esta batería debería ser capaz de mantener al circuito energizado durante varios años en el caso del DS1307, y durante meses para el DS3231. La tensión de alimentación de batería es de 2.0 a 3.5 para el DS1307 y de 2.3 a 5.0 para el DS3231.

Los RTC son dispositivos ampliamente utilizados en electrónica. Todos los ordenadores personales, servidores, tabletas, y smartphones incorporan uno. También son muy frecuentes en sistemas embebidos y, en general, en multitud de dispositivos que requieren realizar un registro del tiempo.

Para hacer uso de este dispositivo es conveniente emplear una biblioteca descargable directamente del entorno de Arduino, llamada “Adafruit RTC”. Esta biblioteca facilita la lectura de las variables de tiempo y su programación, solo quedan por hacer las conexiones necesarias, dos pines de polarización y dos de control o datos como se puede ver en la ilustración 2.14 (Llamas, 2020).

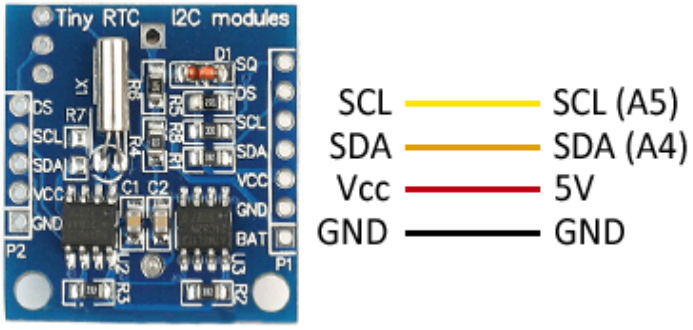



*Ilustración 2.14 RTC.*

# 2.5 LCD

Una LCD (Liquid Crystal Display, o lo que viene a ser Pantalla de Cristal Líquido) es una pantalla que se puede hallar en diversos dispositivos como despertadores, pantallas de relojes, calculadoras, celulares, etc., un sinfín de dispositivos electrónicos que se desarrollan día con día más, gracias a la combinación de una LCD (ilustración 2.15) y un microcontrolador.

*Ilustración 2.15 LCD.*

Para poder hacer uso de este tipo de pantallas con una tarjeta de la familia Arduino, es necesario conectar los respectivos pines, los cuales se observan en la tabla 2.1.

**Tabla 2.1 Pines de conexión de la LCD**

| **Interface pin functions** | | |
|---|---|---|
| **Pin no.** | **Symbol** | **Function** |
| 1 | $V_{SS}$ | Group |
| 2 | $V_{DD}$ | Supply voltage for logic + 5V |
| 3 | $V_0$ | Operating voltage for LCD |
| 4 | RS | H: Data, L: Instruction code |
| 5 | RW | H: Read (MPU- Module), L: Write (MPU – Module) |
| 6 | E | Chip enable signal |
| 7 | DB0 | Data bus line |
| 8 | DB1 | Data bus line |
| 9 | DB2 | Data bus line |
| 10 | DB3 | Data bus line |
| 11 | DB4 | Data bus line |
| 12 | DB5 | Data bus line |
| 13 | DB6 | Data bus line |
| 14 | DB7 | Data bus line |

| | | |
|---|---|---|
| 15 | A | LED + |
| 16 | K | LED - |

Tomando en cuenta las terminales necesarias que requiere una LCD, no queda más que seleccionar una pantalla para comenzar a desarrollar un proyecto, dependiendo de las necesidades que se exijan, podemos usar una LCD de las diversas dimensiones existentes, como pueden ser de 5”, 20”, con tamaños de caracteres de 16 x 2, 8 x 2, 5x2, por mencionar algunos, y en todos los casos, el cursor se encuentra ubicad como se muestra en la ilustración 2.16.

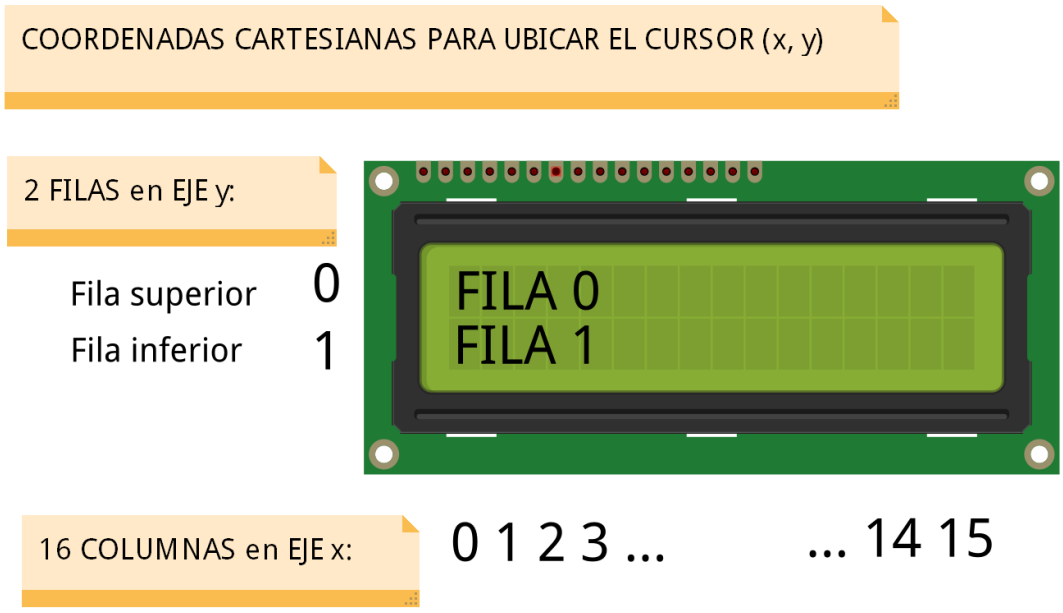



*Ilustración 2.16 Cursor del LCD.*

### 2.5.1 Tipos de LCD

Actualmente encontramos en el mercado los siguientes tipos de pantallas LCD:

- LCD de líneas.
- LCD por puntos.
- Display OLED.
- Display LED.
- Display TFT.

#### 2.5.1.1 LCD de líneas (Ilustración 2.17)

Es un tipo de pantalla que muestra la información a través de líneas. La información se sitúa en líneas y no podemos salirnos de ese marco. Este tipo es el más utilizado, económico y conocido, pero también es el que menos capacidades brinda, ya que solo muestra una determinada información y por lo general solo es texto.

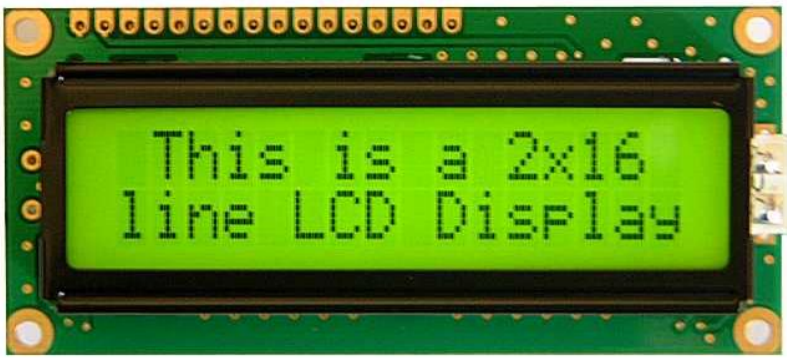



*Ilustración 2.17 LCD de líneas.*

#### 2.5.1.2 LCD por puntos (Ilustración 2.18).

Funciona casi igual que la anterior, pero la principal diferencia radica en que ésta muestra la información por una matriz de puntos. Así, en este tipo de pantalla podemos situar el texto e incluso imágenes en cualquier posición. Además, podemos tener varios tamaños de letra dentro de la misma pantalla, algo que no ocurre en la LCD de líneas, cuyo tamaño es siempre el mismo.

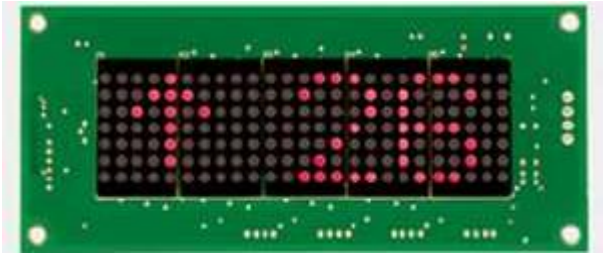


*Ilustración 2.18 LCD de puntos.*

#### 2.5.1.3 Display OLED

Es para muchos un tipo de display propio mientras que para otros está dentro de los tipos de LCD. El display OLED (Ilustración 2.19) es una pantalla que nos muestra información, pero su construcción es diferente al de la pantalla LCD, ya que utiliza diodos led con componentes orgánicos para su creación. A diferencia de los anteriores tipos, los displays OLED ofrecen una mayor resolución, color y un menor consumo energético. Al igual que los monitores de

ordenador o los LCD por puntos, las pantallas OLED utilizan la matriz de puntos o pixeles (ya que podemos utilizar varios colores en el mismo display) para mostrar el contenido.


*Ilustración 2.19 Display OLED.*

#### 2.5.1.4 El display LED

El también llamado LCD Led (ilustración 2.20) es similar al display OLED, pero los diodos led no contienen elementos orgánicos. Su rendimiento no es tan alto como el display OLED, pero si ofrece más resolución que la pantalla LCD por puntos y ofrece color.


*Ilustración 2.20 Pantalla LED.*

### 2.5.2 Display TFT

El display de la ilustración 2.21 es el tipo de LCD más reciente que existe en el mercado. Podemos decir que el display TFT utiliza pixeles como los monitores de los ordenadores o los televisores y que podemos emitir cualquier tipo de información a través de estas pantallas. Su consumo energético es mayor que cualquiera de los anteriores tipos de ahí que se utilicen

tamaños pequeños. El tamaño de estos displays se mide en pulgadas a diferencia de algunos de los otros tipos de displays que se miden por caracteres o por ancho de pantalla (García Cobo, 2020).


*Ilustración 2.21 Display TFT.*

## 2.6 Interruptores

Se conoce como interruptor a un elemento intercalado en cualquier circuito eléctrico, encargado de, como su nombre lo indica, interrumpir la circulación de la corriente eléctrica, por lo que tiene la capacidad de activar o desactivar al circuito o dispositivo en cuestión. Estos elementos son fabricados por medio de componentes conductores de la corriente eléctrica, los cuales se pueden colocar en circuito abierto o cerrado (como se ejemplifica en la ilustración 2.22) manualmente por medio de un elemento no conductor (plástico, cerámica, mica, etc.), para así evitar los contactos involuntarios.

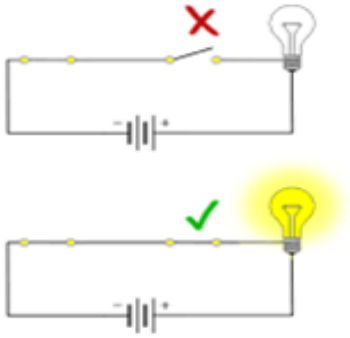


*Ilustración 2.22 Circuito abierto y circuito cerrado.*

La expresión más sencilla de los mismos consiste en 2 contactos de metal inoxidable, y el actuante. Normalmente, los contactos separados, se unen por medio de dicho actuante para

así poder permitir que circule la corriente. El actuante es conocido como la parte móvil, que en una de sus posiciones hace la presión sobre los contactos para así poder mantenerlos unidos y en su segunda posición se separa de uno de los dos contactos para abrir el circuito (ilustración 2.23).

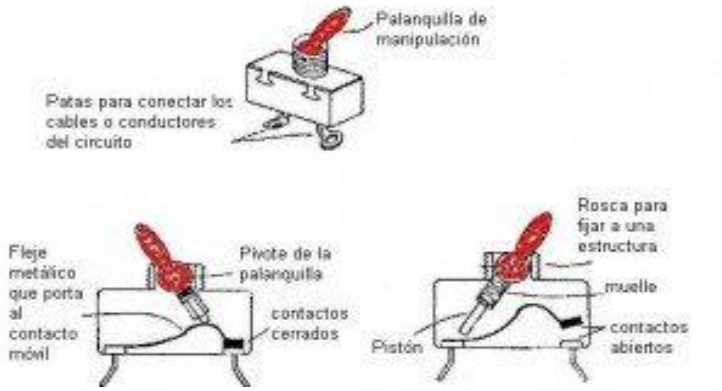



*Ilustración 2.23 Interruptor.*

### 2.6.1 Tipos de interruptores

Es posible encontrar diversos tipos de interruptor, cada uno con características propias, para poder determinar el adecuado, se deben evaluar las necesidades para cada caso y para cada uso, los más conocidos son los siguientes:

➢ Interruptor basculante (ilustración 2.24): Este tipo de interruptor cuenta o posee una palanca que actúa como miembro de actuación. La misma se debe movilizar hacia una determinada posición con la finalidad de que se observe una transformación o alteración en el estado del contacto.


*Ilustración 2.24 Interruptor basculante.*

➢ Interruptor de pulsador (ilustración 2.25): Como su nombre lo indica, este tipo de interruptor está conformado o constituido por un botón, el cual debe ser presionado o pulsado con la finalidad de que modifique o cambie el estado del contacto.

*Ilustración 2.25 Interruptor de pulsador.*

➢ Interruptor rotativo (ilustración 2.26): Los interruptores rotativos disponen de un eje, el mismo debe ser rotado o movido hacia una postura determinada, con la finalidad de que se observe una alteración y/o cambio en el estado del contacto.


*Ilustración 2.26 Interruptor rotativo.*

➢ Interruptor deslizante (ilustración 2.27): muy parecido al interruptor basculante cuenta con un actuador que al ser deslizado puede cerrar o abrir el circuito en los dos contactos de interés. Es el interruptor más común en aparatos electrónicos por su disposición de tamaño, facilidad de manejo y por contar con la posibilidad de manejar más de una línea de tensión, logrando así trabajar con diferentes voltajes usando un solo interruptor.

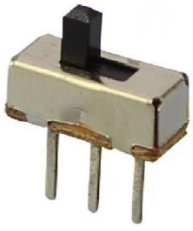


*Ilustración 2.27 Interruptor deslizante.*

➢ Interruptor automático o interruptor magneto-térmico (ilustración 2.28): Este tipo de interruptor posee la peculiaridad de tener la disposición de 2 métodos de resguardo. El primero se basa en que el interruptor automáticamente es apagado en caso de que se presente

un cortocircuito. El segundo se basa en que cuando se origina una sobrecarga de corriente eléctrica, el mismo se desactiva.


*Ilustración 2.28 Interruptor automático o interruptor magneto-térmico.*

➢ Reed switch (Ilustración 2.29): Este término se refiere o alude a un tipo de interruptor que se ubica en una capsula de vidrio. Y es activado solo cuando se descubre un campo magnético.

*Ilustración 2.29 Reed switch.*

➢ Interruptor centrifugo (ilustración 2.30): Este interruptor solo se activa o desactiva cuando es expuesto a una fuerza de carácter centrifuga.

*Ilustración 2.30 Interruptor centrifugo.*

➢ Interruptores DIP o Dual in line package (ilustración 2.31): Este tipo de interruptor consta de un conjunto de interruptores pequeños ligados entre ellos, constituyendo u estableciendo una línea doble de contactos (Anónimo, Paraquesirve.tv, 2020).

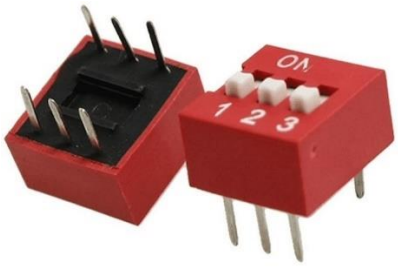



*Ilustración 2.31 Interruptores DIP.*

# 2.7 Pulsadores o botones

Un pulsador es un tipo de interruptor o switch, cuya función es permitir o interrumpir el paso de la corriente eléctrica de manera momentánea, a diferencia de un switch común, un pulsador solo realiza su trabajo mientras se le tenga presionado, es decir sin enclavamiento. Existen pulsadores NC (NC) y NA (NO), es decir normalmente cerrados y normalmente abiertos (Anónimo, SHOPTRONICA, 2020).

*Ilustración 2.32 Símbolo del pulsador.*

Como se puede observar en la ilustración 2.32, el símbolo del pulsador, nos da una idea muy clara de cómo funciona, ya que tendremos dos contactos, que serán unidos al oprimir el botón, permitiendo así el flujo de energía. En el mercado podemos encontrar un enorme catálogo de botones (ilustración 2.33) con diversas características, como el tamaño, la rigidez, altura del mecanismo, etc., para lograr cumplir con las necesidades de cada proyecto.


*Ilustración 2.33 Pulsadores.*

Dentro de esa variedad de pulsadores existen algunos con dos pines de conexión y otros tantos con 4 pines, en este segundo caso contienen dos pares, que cada uno se encuentra conectado internamente, por lo que se reduce como un botón común con dos terminales (ilustración 2.34).

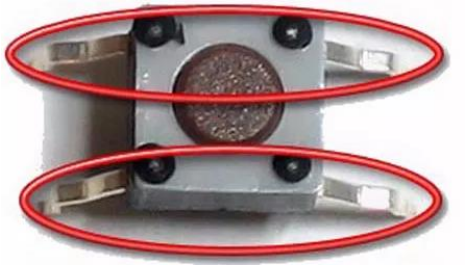


Ilustración 2.34 Configuración de pines del pulsador.

### 2.7.1 Resistencias de polarización

Cuando nos desenvolvemos en el entorno de los microcontroladores, nos encontramos con un término poco común, me refiero a la polarización de una E/S (Entrada o salida), debemos saber que hay dos tipos de polarización, polarización alta la resistencia (término inglés Pullup) va conectada a + (5 V) o polarización baja la resistencia (término inglés Pulldown) va conectada a masa – (0 V). Representado en los esquemas de la ilustración 2.35.

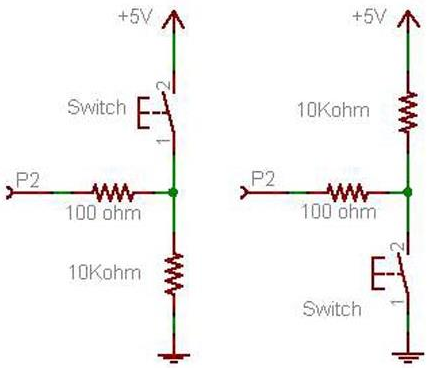



*Ilustración 2.35 Resistencias de polarización.*

## 2.8 Sistemas de control

En la sociedad actual nos encontramos rodeados todo el tiempo de sistemas de control, muchas veces sin darnos cuenta, debido a que es cotidiano el interactuar con ellos, hasta inconscientemente podemos estar usándolos sin percatarnos que son sistemas de control. Un ejemplo es al conducir un automóvil, cuando ejercemos control sobre la velocidad manteniéndola constante, mediante procesos electrónicos se hace variar las revoluciones por minuto a las que circulamos a partir de la fuerza ejercida sobre el pedal. En los hogares la mayoría de electrodomésticos están hechos a partir de un sistema de control, como el refrigerador, que mantiene una temperatura deseada, ajustable mediante perillas, botones o incluso con pantallas táctiles, el tostador de pan, lavadora, etc.

Por otra parte, todos formamos parte de un sistema. Por ejemplo: la naturaleza, el universo y el mismo ser humano, están compuestos por diversos sistemas. Un sistema está definido como el conjunto de elementos interrelacionados entre sí para cumplir un fin común.

Por lo tanto, teniendo en cuenta que un sistema es un conjunto de elementos relacionados entre sí para realizar un fin común y que control prácticamente significa la manipulación de ciertas variables para lograr satisfacer una necesidad, un sistema de control sería el conjunto de elementos interconectados entre sí para lograr la manipulación de ciertas variables con el fin de satisfacer una necesidad especifica.

### 2.8.1 Clasificación de los sistemas de control.

Los sistemas de control pueden ser clasificados como sistemas de control en lazo abierto y sistemas de control en lazo cerrado, según sea su diseño y su funcionalidad requerida.

### 2.8.1.1 Sistema de control en lazo abierto.

Al sistema de control en el cual la salida no afecta la acción de control se denomina sistema de control en lazo abierto. En otras palabras, en un sistema de control en lazo abierto no se mide la salida ni se realimenta para compararla con la entrada; por lo tanto, a cada entrada de referencia le corresponde una condición operativa fija; dando como resultado que la precisión del sistema depende de la calibración (ilustración 2.36).

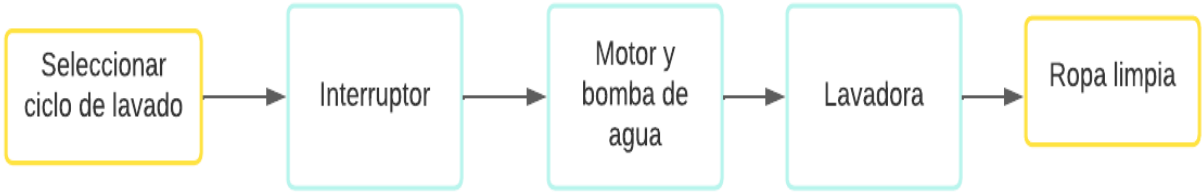



*Ilustración 2.36 Sistema de control de lazo abierto.*

La ilustración 2.36 muestra un sistema de control de lazo abierto, en cuyo diagrama se muestra cómo únicamente existen dos señales (de entrada y de salida) interrelacionados entre sí, pero sin haber un lazo de retroalimentación que compare la señal obtenida después del proceso con la señal de la entrada.

### 2.8.1.2 Sistema de control en lazo cerrado

En un sistema de control en lazo cerrado (ilustración 2.37), se alimenta al controlador con la señal de error de actuación, el cual es la diferencia entre la señal de entrada y la señal de realimentación (que puede ser la señal de salida misma o una función de la señal de salida y sus derivadas y/o integrales), a fin de reducir el error y llevar la salida del sistema a un valor conveniente.

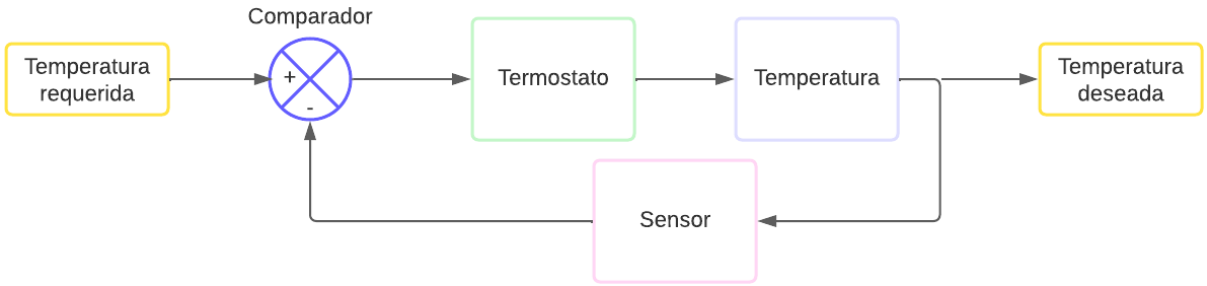



*Ilustración 2.37 Sistema de control de lazo cerrado.*

### 2.8.2 Componentes del Sistema de Control

En la industria ha tomado gran relevancia la aplicación del control automático de procesos ya que este permite mantener controladas ciertas variables como la temperatura, la humedad, la viscosidad, la presión, entre otras. Mantener estas variables estables es el objetivo del sistema de control. Cuyos elementos básicos son los siguientes:

- Transductor (Sensor/Transmisor).
- Controlador.
- Actuador.

La importancia de estos componentes radica en que estos realizan las tres operaciones básicas (ilustración 2.38) que deben estar presentes en todo sistema de control; estas operaciones, respectivamente, son:

**Medición:** la medición de la variable que se controla se hace generalmente mediante la, combinación de sensor y transductor.

**Decisión:** con base en la medición, el controlador decide qué hacer para mantenerla variable en el valor que se desea.

**Acción:** como resultado de la decisión del controlador debe efectuar una acción en el sistema, generalmente ésta es realizada por el elemento final de control.

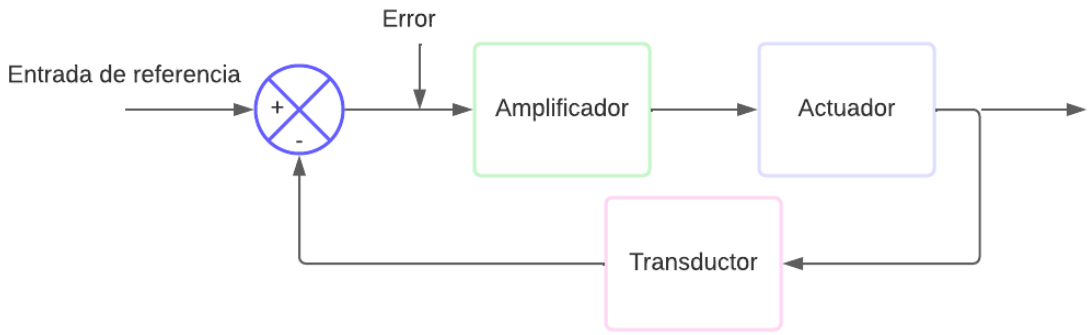



*Ilustración 2.38 Elementos de un sistema de control.*

Estas tres operaciones son forzosas para todos los sistemas de control. La toma de decisión puede realizarse con un sistema de control en lazo abierto o en lazo cerrado.

# 3 Desarrollo experimental

## 3.1 Diagrama a bloques

En el desarrollo de este proyecto, se utilizó para la programación el entorno de Arduino como una herramienta que facilitó el manejo de los diversos componentes, como lo son el RTC, el módulo para la memoria micro SD, etc. Para entender mejor la estructura del sistema y su implementación, se presenta el siguiente diagrama de bloques:

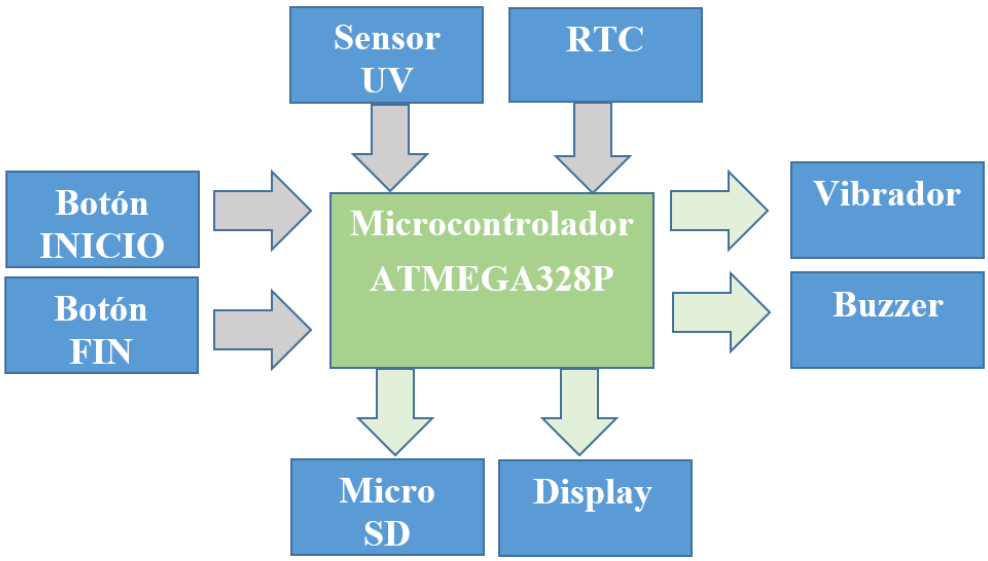



*Ilustración 3.1 Diagrama de bloques.*

Donde:

- *RTC.* - Es el Reloj, en el cual se apoya el microcontrolador para consultar la hora y minuto en tiempo real, así como la fecha con día, mes y año.
- *Sensor UV.* - Las mediciones del índice de radiación ultravioleta se realizan mediante este sensor, que entrega diferentes valores de voltaje dependiendo de la intensidad UV que detecte, para posteriormente ser evaluadas por el microcontrolador y determinar la lectura correcta a partir de promedios de las miles de lecturas que se toman por segundo.
- *Botón INICIO.* - Este botón es el medio por el cual el usuario le indica al dispositivo en qué momento se deben empezar a realizar las mediciones y cuándo dejar de hacerlo, ya que mientras se mantenga presionado tomará lecturas.

➢ *Botón FIN*. - Así como el botón de INICIO, este botón sirve para comunicarse con el microcontrolador, en este caso para indicar el momento en el cual se desea finalizar la sesión, para entonces realizar las actividades correspondientes.

➢ *Micro SD*. - Esta es la memoria extraíble, en la que se llevará el registro cotidiano de las sesiones, los datos se almacenarán en una hoja de cálculo en un formato que se mostrará posteriormente.

➢ *Display*. - Mediante este display de 8 x 2 caracteres, se facilitará la comunicación del usuario con el dispositivo, ya que se mostrarán diversos mensajes, como cuándo se están tomando lecturas, cuándo hay un trabajo en curso o no, incluso si el equipo se encuentra en espera, además tendrá una utilidad extra, ya que se mostrará en todo momento la hora actualizada.

➢ *Buzzer y vibrador*. - Son elementos utilizados para enviar alertas, a partir de la vibración del dispositivo y el sonido que se emitirá, el usuario debe finalizar los inicios de actividad, y estos elementos sirven como recordatorio, para evitar que olvide apagarlo o finalizar las operaciones; de igual manera, si quiere guardar o finalizar una actividad sin antes haber iniciado una medición, estos elementos y el display le alertarán de cualquier error en su manejo.

## 3.2 Diagrama esquemático.

El esquema expuesto en la ilustración 3.2 muestra las conexiones que se deben realizar entre los diferentes elementos para lograr el sistema funcional:

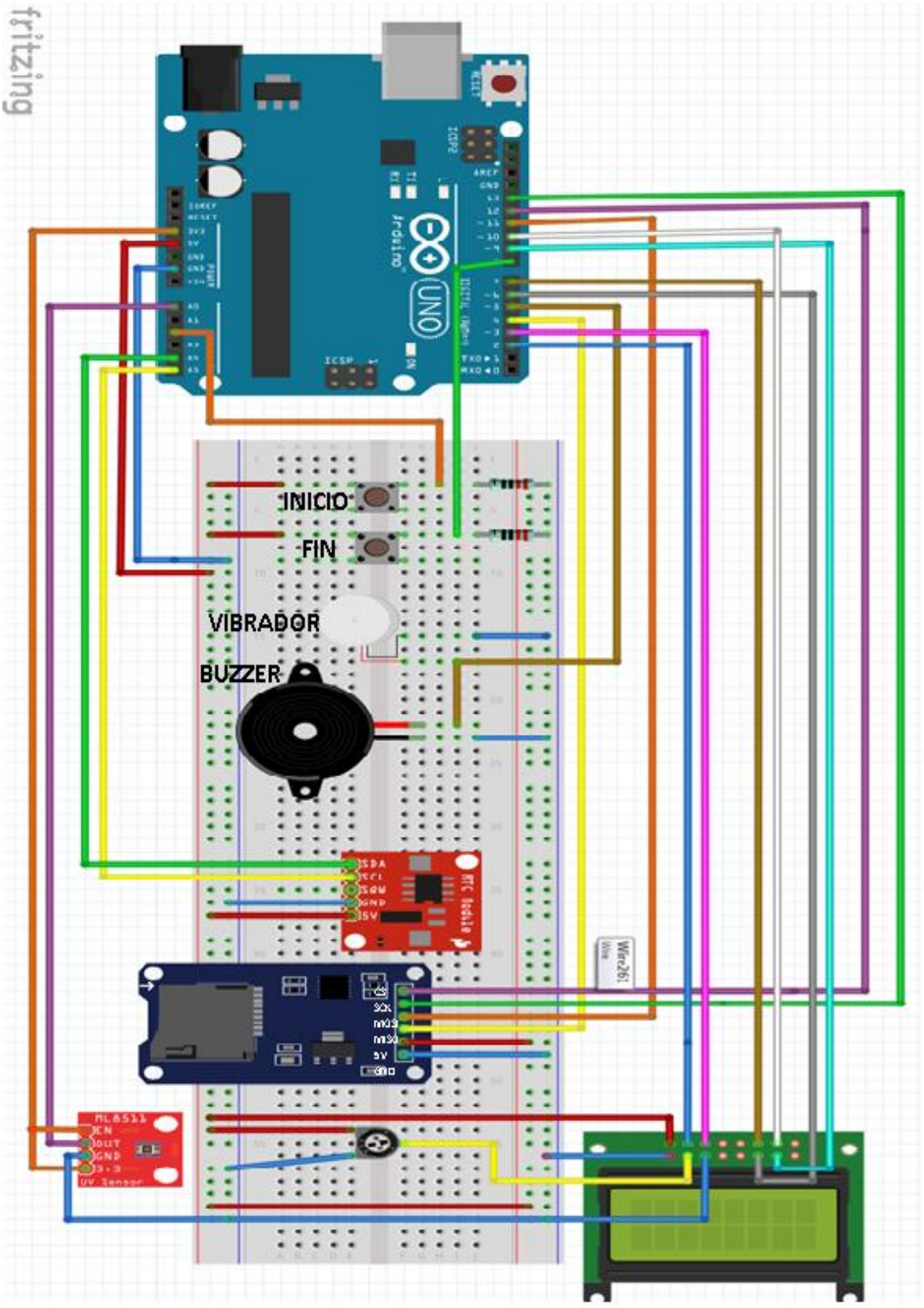


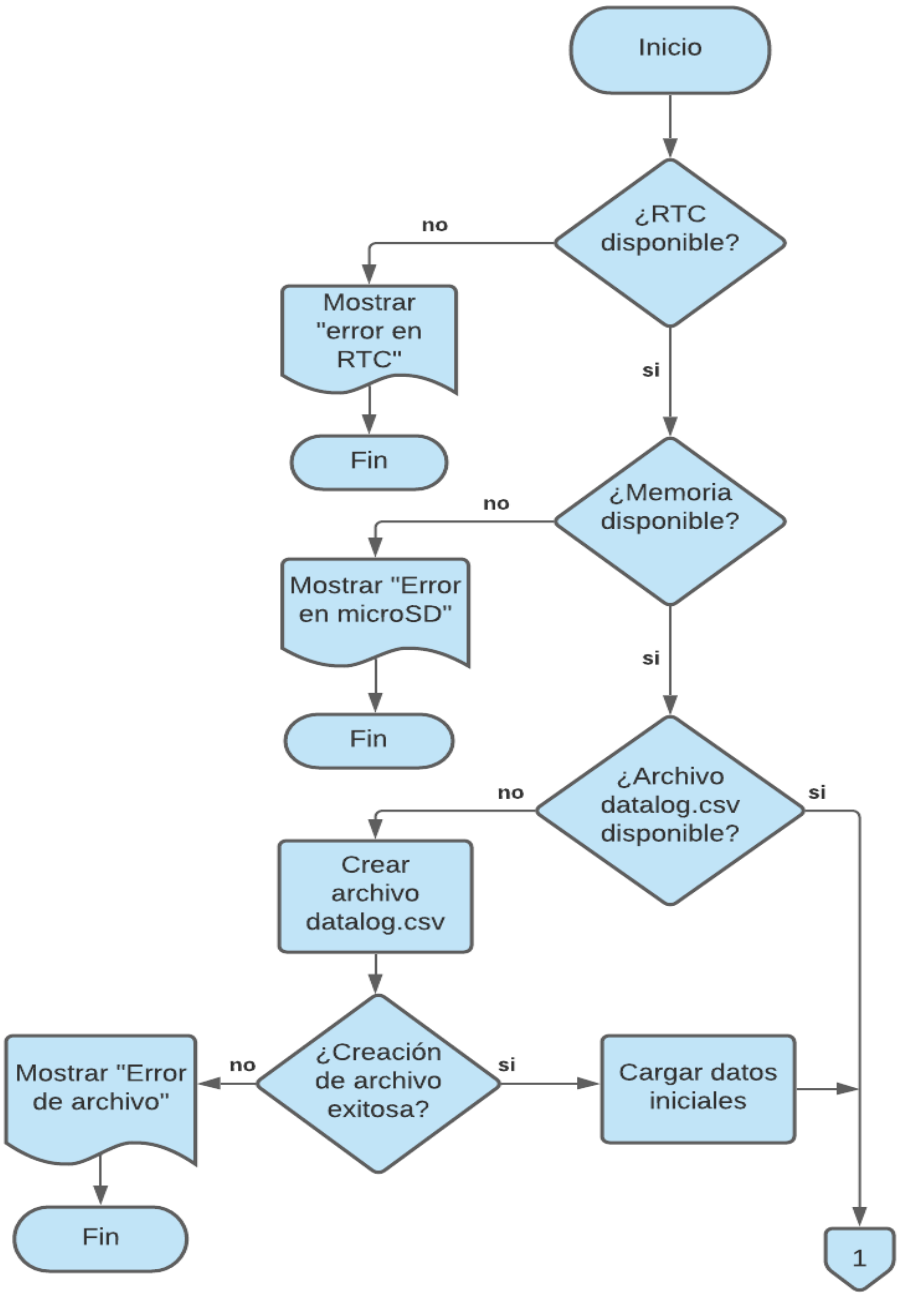

*Ilustración 3.2 Diagrama esquemático.*

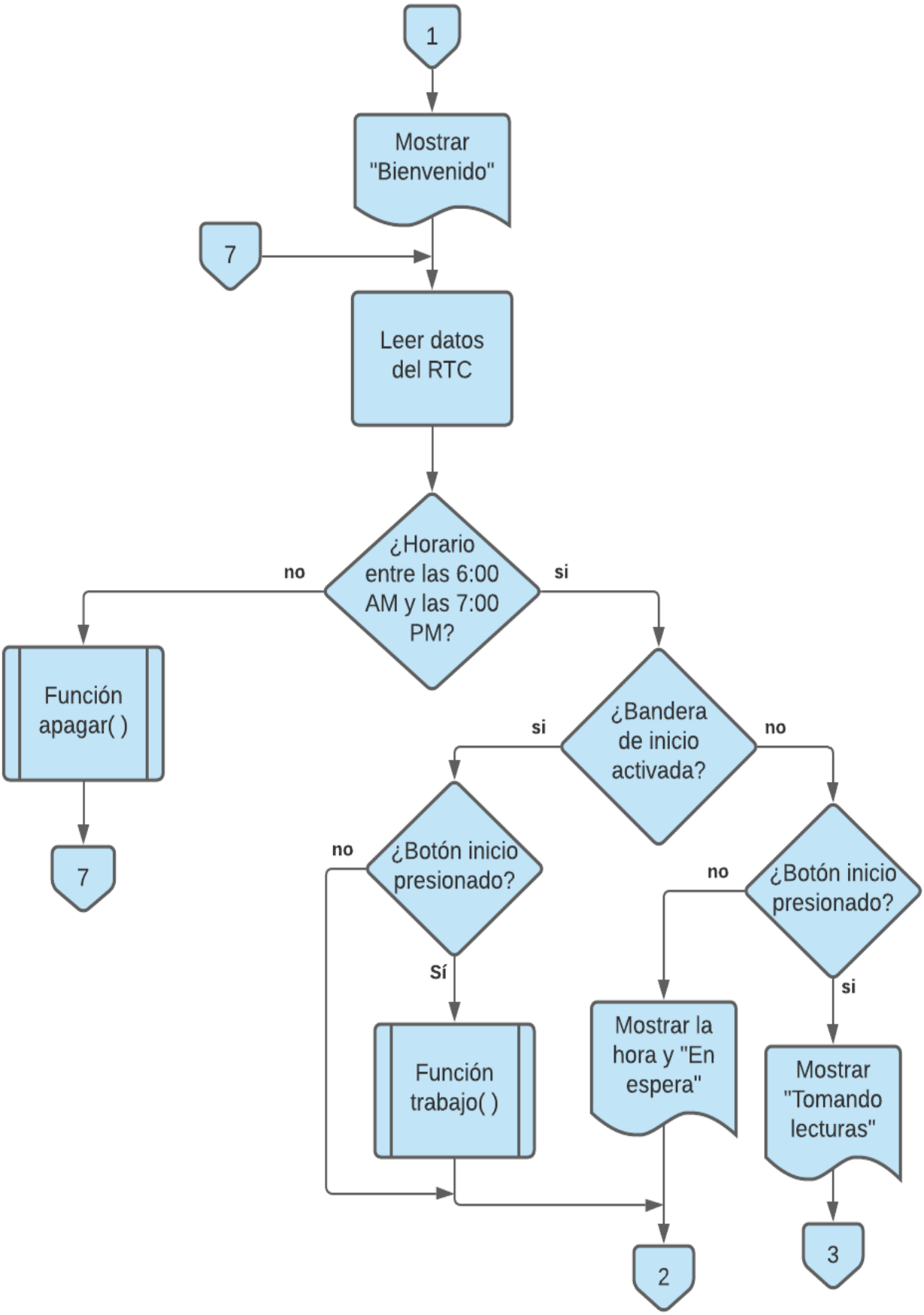
## 3.3 Diagramas de flujo

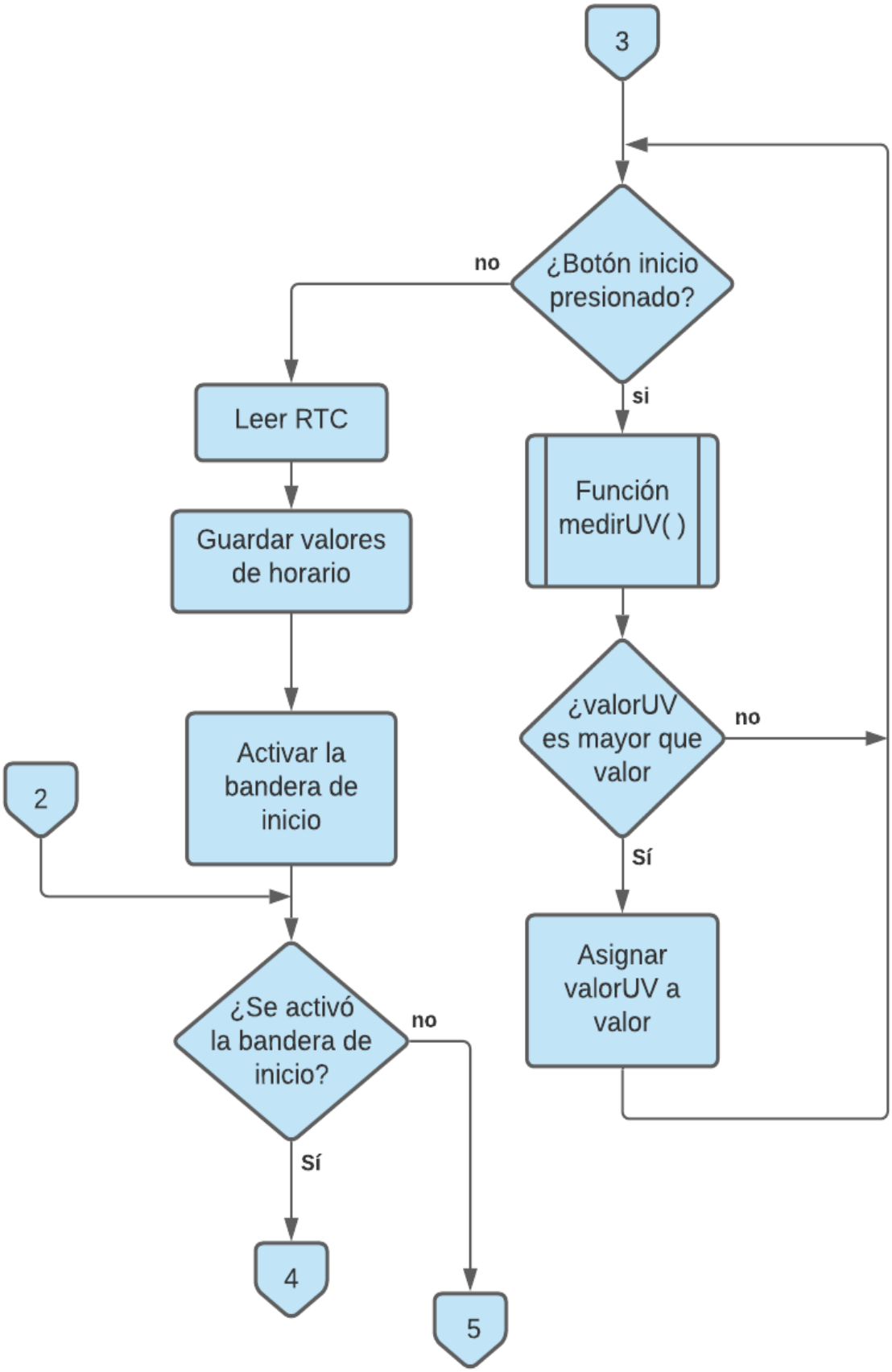
En este apartado se presentan los diversos algoritmos implementados en el sistema mediante sus diagramas de flujo.

### 3.3.1 Función void loop( )

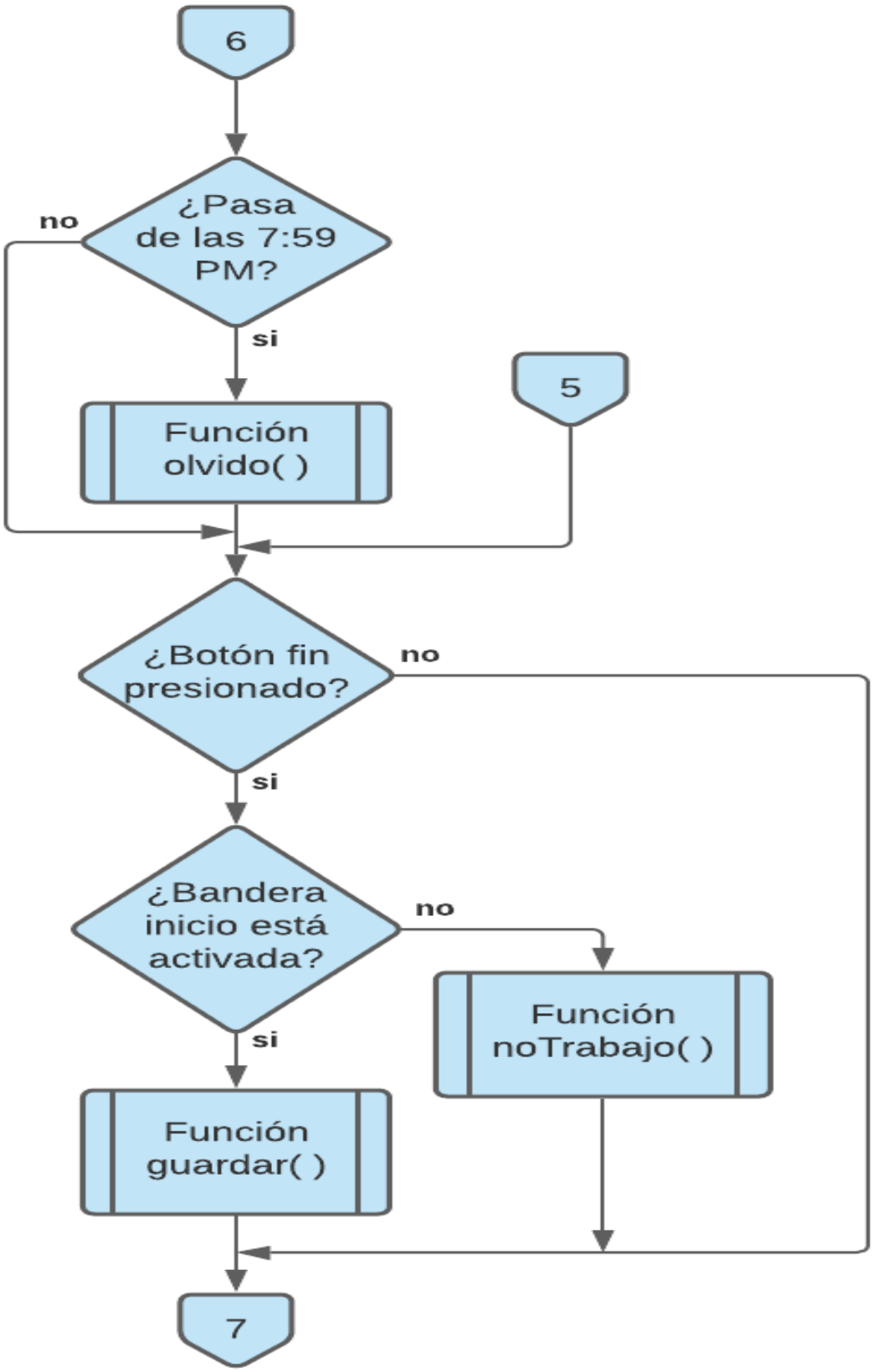
Dado que se realizó la programación en el entorno de Arduino, se tiene como función principal la denominada “loop” la cual se realiza de manera cíclica.




1
Mostrar "Bienvenido"
7
Leer datos del RTC
¿Horario entre las 6:00 AM y las 7:00 PM?
no
si
Función apagar( )
7
¿Bandera de inicio activada?
si
no
¿Botón inicio presionado?
no
Sí
Función trabajo( )
¿Botón inicio presionado?
no
si
Mostrar la hora y "En espera"
Mostrar "Tomando lecturas"
2
3

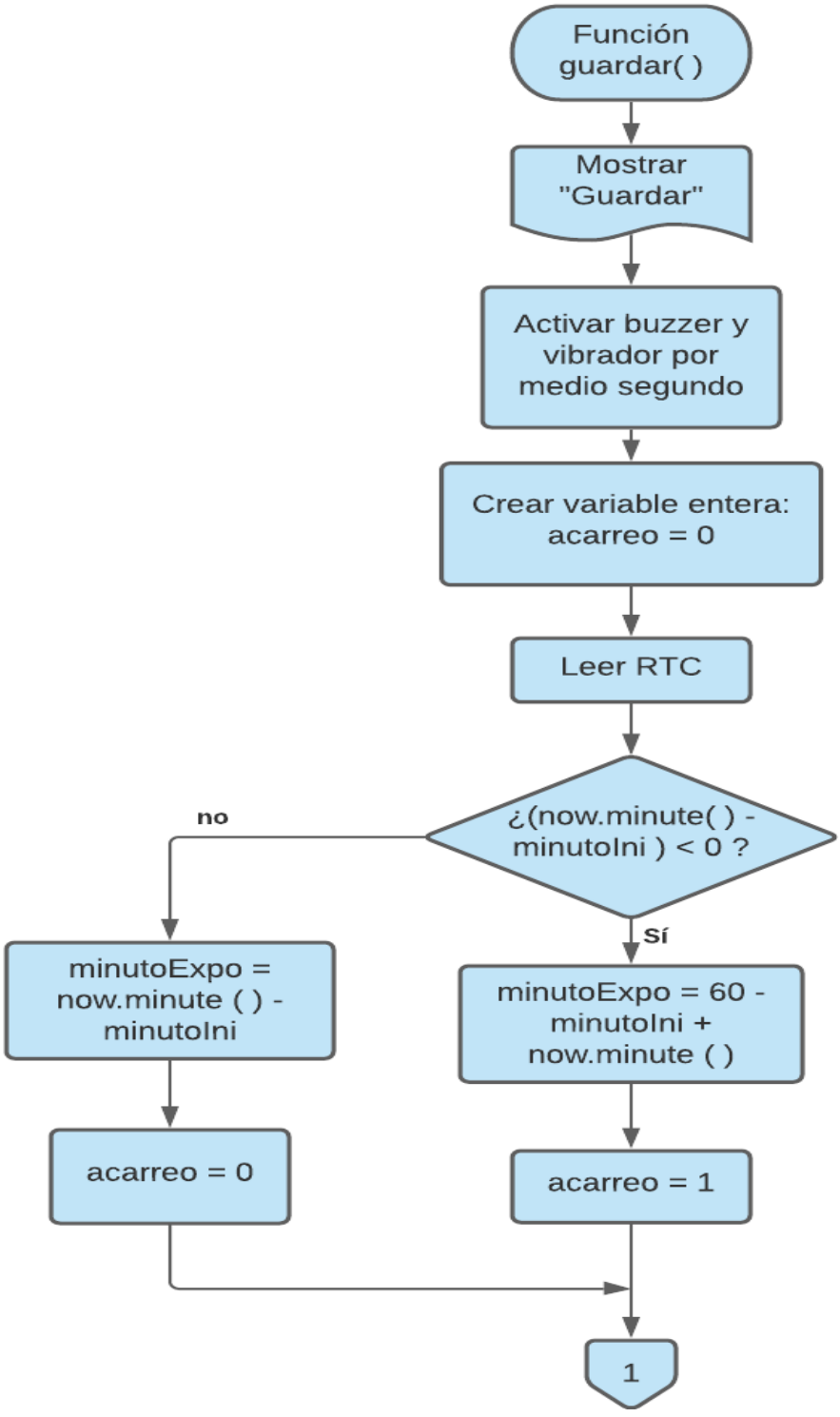

3
¿Botón inicio presionado?
no
si
Leer RTC
Guardar valores de horario
Activar la bandera de inicio
Función medirUV( )
¿valorUV es mayor que valor
no
Sí
Asignar valorUV a valor
2
¿Se activó la bandera de inicio?
no
Sí
4
5

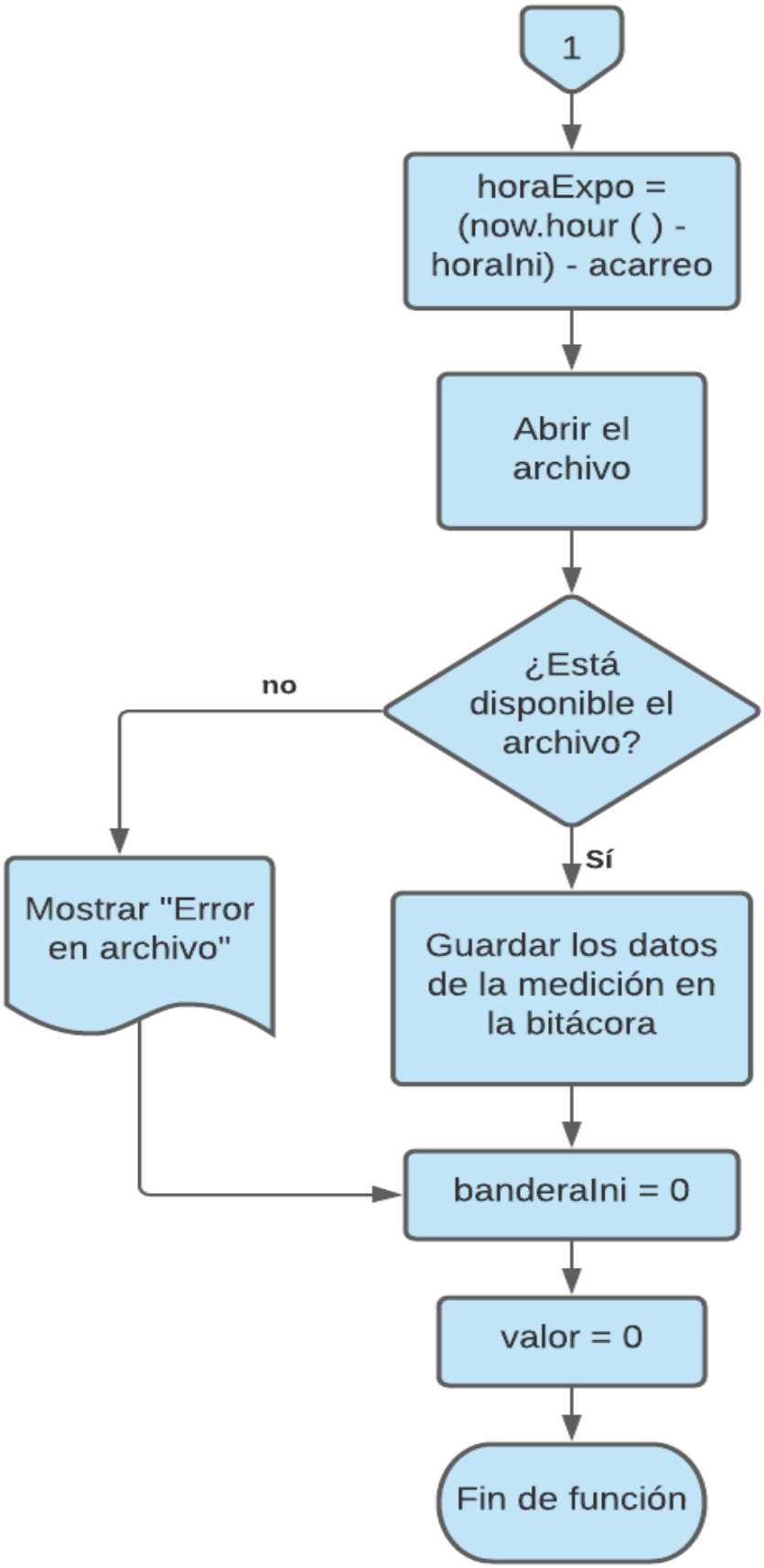

4
¿Transcurrió una hora?
Sí
Función alerta( )
¿Botón fin está presionado?
si
no
Mostrar "Operando"
Leer RTC
Mostrar la hora
6


6
¿Pasa de las 7:59 PM?
no
si
Función olvido( )
5
¿Botón fin presionado?
no
si
¿Bandera inicio está activada?
no
si
Función noTrabajo( )
Función guardar( )
7

### 3.3.2 Función guardar( )

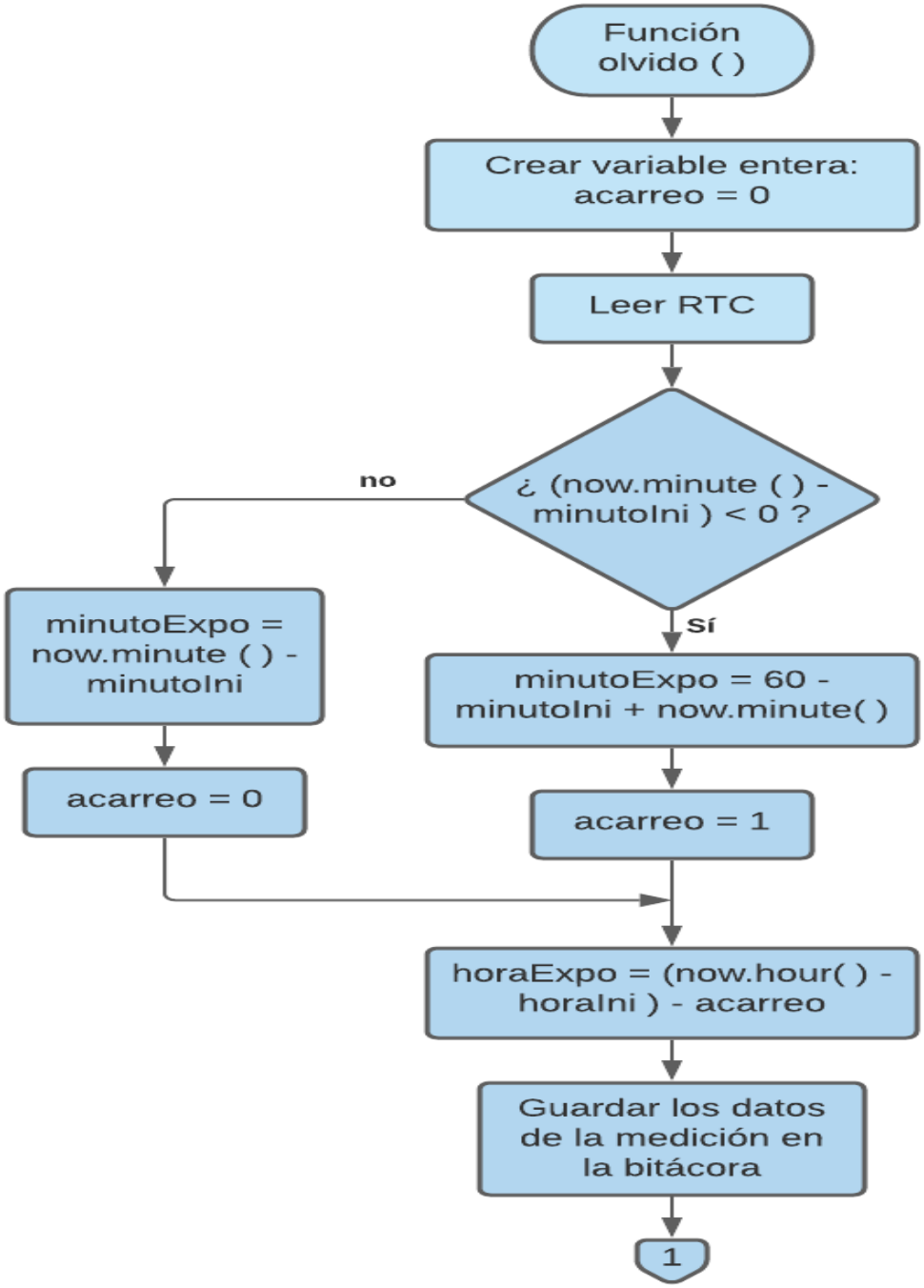
Cuando el botón FIN es presionado, se llama a una función llamada guardar( ), misma que realizará el guardado de la medición y su respectiva información, siempre y cuando se tenga una sesión previamente iniciada.

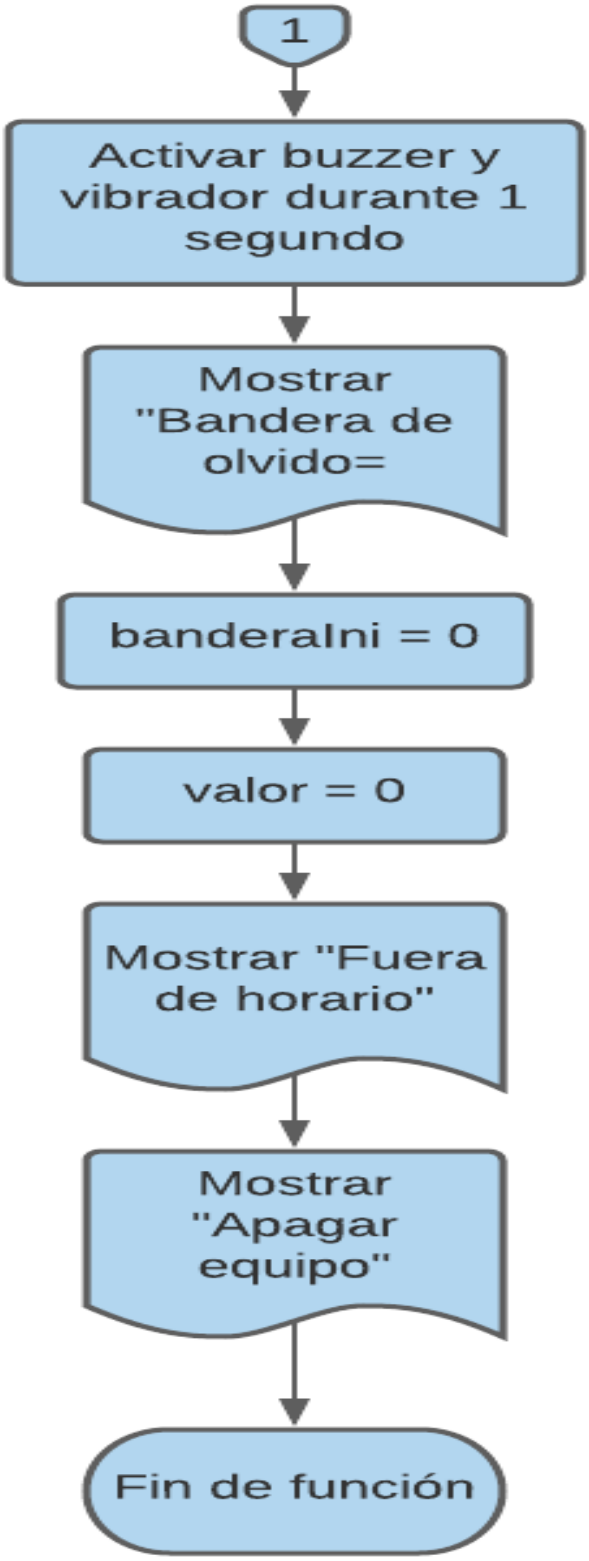



1
horaExpo = (now.hour ( ) - horaIni) - acarreo
Abrir el archivo
¿Está disponible el archivo?
no
Sí
Mostrar "Error en archivo"
Guardar los datos de la medición en la bitácora
banderaIni = 0
valor = 0
Fin de función

### 3.3.3 Función olvido( )

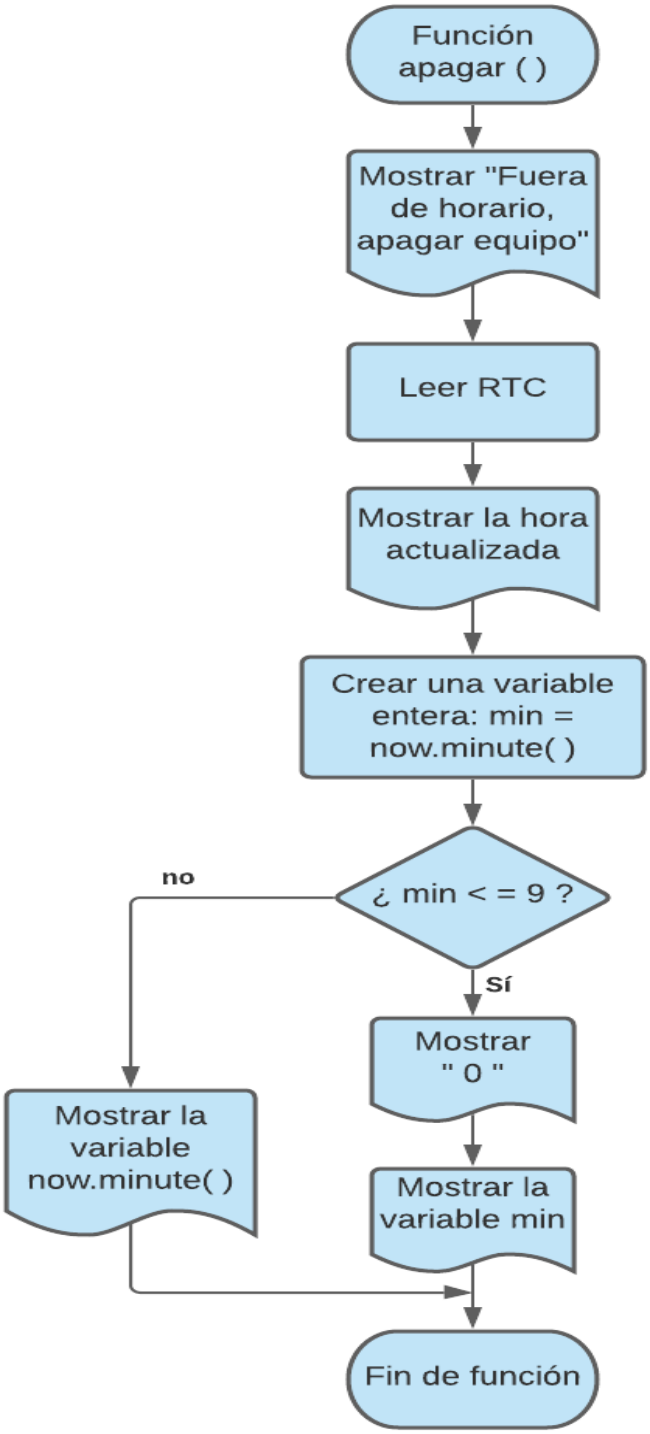
El sistema contempla como horario adecuado para funcionar de 6:00 a 20:00 horas, de tal manera que si son las 19:59 y el usuario no ha finalizado la sesión en curso, automáticamente el programa guardará la información y en la bitácora indicará dicha situación mediante un "1" en la fila titulada "Olvido".




1
Activar buzzer y vibrador durante 1 segundo
Mostrar "Bandera de olvido=
banderaIni = 0
valor = 0
Mostrar "Fuera de horario"
Mostrar "Apagar equipo"
Fin de función

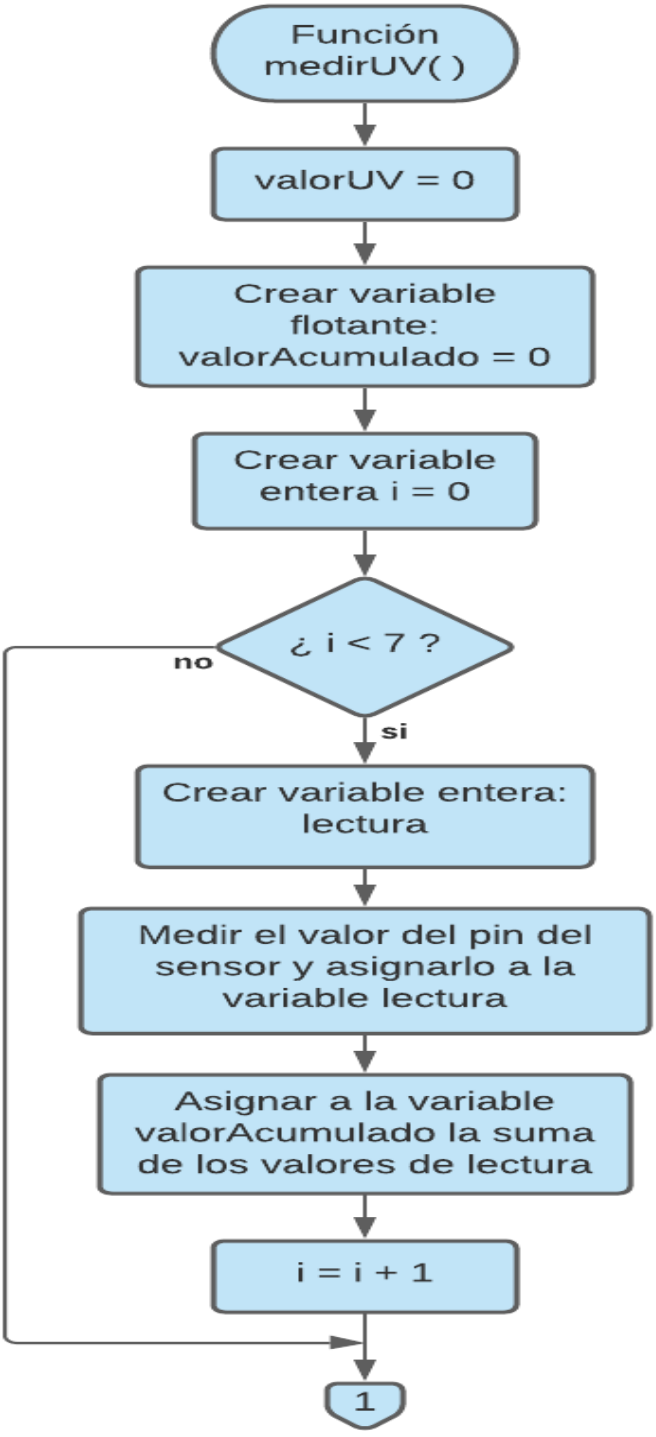
### 3.3.4 Función apagar( )

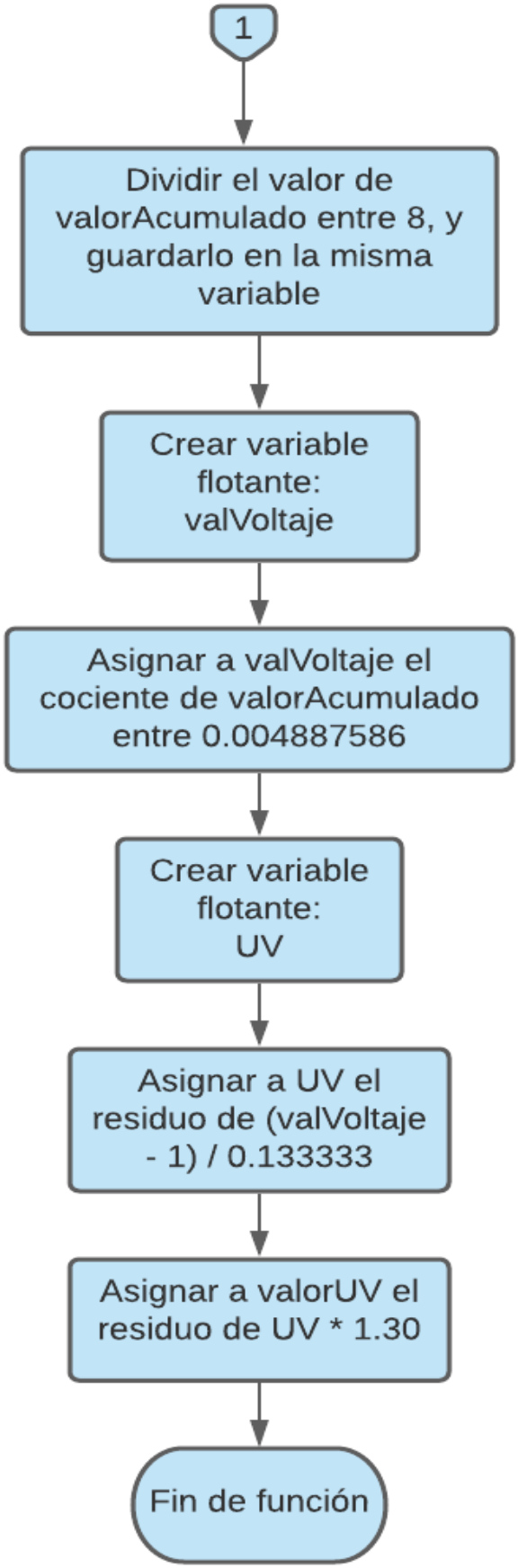
Cuando el radiómetro se inicia antes de las 6:00 o después de las 20:00 horas, no se podrá realizar una medición, y el dispositivo lo notificará al usuario mediante esta función.



### 3.3.5 Función medirUV( )

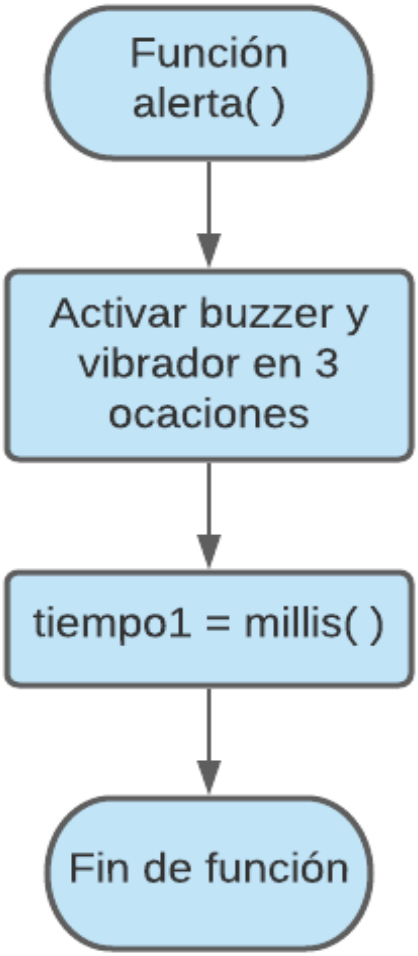
Mientras el botón inicio se encuentre presionado, se estarán tomando reiteradas mediciones a través de la función medirUV( ), como lo muestra el siguiente algoritmo:




1
Dividir el valor de valorAcumulado entre 8, y guardarlo en la misma variable
Crear variable flotante: valVoltaje
Asignar a valVoltaje el cociente de valorAcumulado entre 0.004887586
Crear variable flotante: UV
Asignar a UV el residuo de (valVoltaje - 1) / 0.133333
Asignar a valorUV el residuo de UV * 1.30
Fin de función

### 3.3.6 Función alerta( )

Con la finalidad de que el usuario no olvide que el dispositivo se encuentra encendido se agregó esta función, para enviar una alerta cada hora aprovechando elementos como el buzzer y el vibrador que son fáciles de percibir.



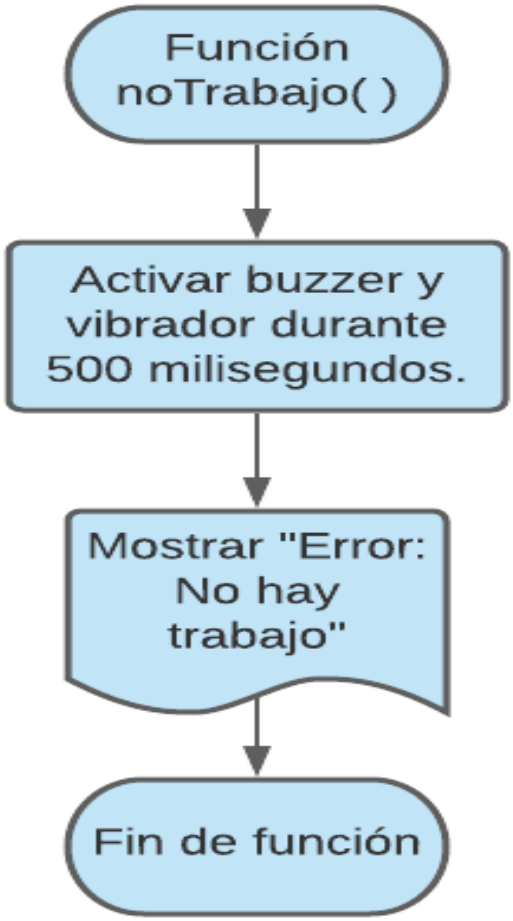

### 3.3.7 Función trabajo( )

Cuando haya una sesión iniciada y el usuario intente realizar una nueva medición sin antes haber finalizado la existente, el sistema debe alertarlo de la situación, tanto en los elementos de alerta como en el display.


Función trabajo( )
Activar buzzer y vibrador durante 500 milisegundos.
Mostrar "Error: Trabajo en curso"
Fin de función

### 3.3.8 Función noTrabajo( )

Esta función se ejecuta cuando el usuario intenta finalizar una actividad, pero que en realidad no se inició ninguna, presentando un mensaje con dicha situación, así como las alertas sonoras y de vibración.




## 3.4 Programación en Arduino

Una vez teniendo los algoritmos correctamente estructurados se procede a realizar su programación, que se explica a continuación para cada elemento.

### 3.4.1 RTC

El RTC contiene una batería propia, como se mencionó anteriormente, de manera que aun teniendo apagado el radiómetro, se sigue contando con la hora y la fecha actualizada, con el

mínimo índice de error (dos minutos por cada año). Su configuración es sencilla, pues se puede aplicar al entorno de Arduino una biblioteca descargable de Adafruit[1], que es compatible para los modelos de RTC, en esta propuesta se empleó el DS1307, por ser de un tamaño reducido, tener una gran exactitud y además ser de un bajo costo. Una vez descargada la biblioteca se puede comenzar a trabajar con el dispositivo.

```
#include <Wire.h>  // Biblioteca para la comunicaci□n I2C (SD y la RTClib)
#include "RTClib.h"  //biblioteca RTC de Adafruit
```

Es posible revisar si el microcontrolador tiene comunicación con el RTC, de la siguiente manera:

```
void setup()
{
if (! rtc.begin())  // Comprobamos que se reconozca al reloj de tiempo real
{
  lcd.setCursor(0, 0);
  lcd.print("Error:  ");
  lcd.setCursor(0, 1);
  lcd.print("en RTC  ");
  delay(200);
  while (1)
  {
  }
}
```

El ajuste a los datos que nos proporcionará el reloj, se realiza de la siguiente manera:

```
// Ponemos en hora, solo la primera vez, luego comentar y volver a cargar.

 rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
```

Así se aplican las referencias del sistema.

La biblioteca instalada del RTC, posee instrucciones que nos brindan la información necesaria. Para asegurarse de que ésta sea correcta, se puede realizar el siguiente ejemplo:

[1] https://github.com/adafruit/RTClib

```
void loop () {
 DateTime now = rtc.now();

 Serial.print(now.day());
 Serial.print('/');
 Serial.print(now.month());
 Serial.print('/');
 Serial.print(now.year());
 Serial.print(" ");
 Serial.print(now.hour());
 Serial.print(':');
 Serial.print(now.minute());
 Serial.print(':');
 Serial.print(now.second());
 Serial.println();
 delay(3000);
}
```

Y se observa en el monitor serial el día, mes, año, hora, minuto y segundo actual.

### 3.4.2 Micro SD

En el caso del módulo de la memoria micro SD, no es necesario descargar ninguna biblioteca, pues las funciones relacionadas a su manejo ya vienen incluidas en el entorno, únicamente será requerido invocar la biblioteca en el programa:

```
#include <SD.h>
```

Para el caso de este proyecto respecto a las necesidades que se deben cumplir, se emplean las instrucciones dedicadas a comprobar si la memoria está correctamente instalada, ya que, de no tener comunicación con ella, es indispensable saber dicha situación, para evitar perder mediciones, para ello se realizó el siguiente programa:

```
if(!SD.exists("datalog.csv"))  //Comprobamos exitencia de archivo
{
  myFile = SD.open("datalog.csv", FILE_WRITE); // Creacion del archivo
  if (myFile)                  //Comprobamos creaci□n del archivo
  {
    myFile.print("Olvido");
    myFile.print(',');
    myFile.print("IUV");
    myFile.print(',');
    myFile.print("A&o");
    myFile.print(',');
    myFile.print("Mes");
    myFile.print(',');
    myFile.print("Dia");
    myFile.print(',');
    myFile.print("Hora inicial");
    myFile.print(',');
    myFile.print("Minuto inicial");
    myFile.print(',');
    myFile.print("Hora final");
    myFile.print(',');
    myFile.print("Minuto final");
    myFile.print(',');
    myFile.print("Horas expo");
    myFile.print(',');
    myFile.print("Minutos expo");
    myFile.println("");
    myFile.close();
  }
  else
  {
    lcd.setCursor(0, 0);
    lcd.print("Error de");
    delay(2000);
    lcd.setCursor(0, 1);
    lcd.print("archivo ");
    while (1)
    {
    }
  }
}
```

Si no hubiera comunicación con la micro SD, se observa en el display un mensaje de error en MicroSD.

### 3.4.3 Sensor UV.

El sensor que se empleó en este proyecto fue el ML8511, por las ventajas de tamaño, exactitud, precisión y bajo costo. La salida siempre tiene la forma de voltaje analógico que se procesa mediante el ADC del ATmega328p. Para entender el comportamiento del sensor

y la relación intensidad UV – voltaje de salida, es necesario apoyarse de la gráfica de la ilustración 2.7.

Como se puede observar en la gráfica, el voltaje que será entregado al microcontrolador dependerá de la intensidad de radiación que reciba el sensor, las rectas de colores nos indican el comportamiento del sensor sometido a diferentes temperaturas y es notable que la variación es mínima, notándose a muy altos niveles de radiación.

Para tener un valor aún más preciso se realizan promedios de 8 lecturas, quedando esa parte del programa de la siguiente manera:

```
void medirUV()
{
    valorUV = 0;
    float valorAcumulado = 0; °
    for(int x = 0 ; x < 8 ; x++)
    {
      int lectura = analogRead(A0);
      valorAcumulado += lectura;
    }
    valorAcumulado /= 8;
    float valVoltaje = valorAcumulado * 0.004887586;
    float UV = (valVoltaje-1)/.133333;
    float ajuste = 1.30;
    valorUV = UV * ajuste;
}
```

El promedio obtenido es el voltaje analógico que está relacionado linealmente con la intensidad UV medida (mW / cm 2), como se observa en la ilustración 2.7, de tal manera que si se digitaliza es posible encontrar el valor de intensidad UV. El microcontrolador empleado tiene incorporado un ADC con resolución de 10 bits, por ello no se requiere ningún elemento de hardware adicional para realizar la digitalización al valor que arroja el sensor analógico, considerando el valor de voltaje en 5 V y los 10 bits de resolución, se sabe que la tensión analógica de entrada es convertida en un valor numérico de 0 a 1023, 5 V / 1023 = 0.004887586, valor que se debe multiplicar por el promedio obtenido para obtener la digitalización:

```
float valVoltaje = valorAcumulado * 0.004887586;
```

Una vez digitalizando la lectura de voltaje, se procede a obtener la equivalencia en intensidad UV, haciendo uso de la ecuación de la recta que se presenta en la ilustración 2.7, se emplea la ecuación ordinaria y = mx + b, donde "y" es el valor conocido de la digitalización, "b" es la ordenada al origen, la cual se identifica fácilmente en la gráfica (b = 1), "m" es la pendiente, para obtenerla se necesitan dos puntos de la recta, tomando en cuenta (1,0) y (2.5,11.25), por ser los más exactos en la gráfica, se obtiene que m = 0.1333, el valor de la intensidad UV se localiza en el eje horizontal, y en la ecuación está representado por la variable "x", al despejarla se tiene x = (y – 1) / (0.1333) y nombrando a las variables como fueron declaradas en nuestra programación se obtiene:

```
float UV = (valVoltaje-1)/.133333;
```

Finalmente se realizó un ajuste a la medición obtenida, multiplicándola por 1.30, valor que se obtuvo comparando las mediciones con las que arroja un radiómetro comercial certificado, para llegar a los mismos valores en ambos dispositivos.

### 3.4.4 Display.

El display más cómodo para el dispositivo es el LCD de 8x2 caracteres, con él es posible mostrar los mensajes necesarios en un pequeño tamaño, utilizando muy pocos recursos en el microcontrolador. Para trabajar con este LCD se agrega una biblioteca al entorno de Arduino, con la siguiente línea de código:

*#include <LiquidCrystal.h>*

Posteriormente es necesario definir los pines con los que trabajará el intercambio de información, para el caso de este proyecto fueron los siguientes:

*LiquidCrystal lcd(2, 3, 6, 7, 9, 10); // const int rs = 2, en = 3, d4 = 6, d5 = 7, d6 = 9, d7 = 10;*

Como se puede observar en el diagrama esquemático, son pocos los pines del microcontrolador que se ocuparán para el display, una ventaja más para trabajar con este modelo de 8x2.

Antes de comenzar a mandar mensajes en la pantalla, se debe inicializar con el siguiente comando (donde indicaremos al microcontrolador la dimensión de la misma):

*lcd.begin(8, 2);  //Inicializamos el LCD*

Una vez hecho esto, es posible mandar mensajes, recordando que éste debe ser de 8 caracteres por fila, por lo cual resulta conveniente usar retardos para mostrar una parte del mensaje por algunos segundos y después mostrar lo faltante.

Si se quisiera mostrar en el display el mensaje "BIENVENIDO" se requiere usar las dos filas, debido a que esa palabra contiene 10 caracteres, eso resultaría en escribir "BIEN-  " sobre la fila 1 y "VENIDO " en la segunda, las instrucciones para lograrlo son las siguientes:

*lcd.setCursor(0, 0)*

*lcd.print("BIEN-  ");*

*lcd.setCursor(0, 1);*

*lcd.print("VENIDO");*

*delay(3000);*

El comando "*lcd.setCursor"* sirve para posicionar el cursor en la parte que se desea, el primer valor entre paréntesis se refiere a la fila, y el segundo  a la columna, como lo muestra la ilustración 2.16.

### 3.4.5 Codificación de los diagramas de flujo

Para realizar la programación del diagrama de flujo del programa principal, mismo que se mostró anteriormente, es necesario comenzar incluyendo las bibliotecas con las que se trabajará. Para el caso particular de este sistema se empleó un LCD de 8 x 2 caracteres, con el que se mostrará la información al usuario, y para trabajar con él se debe incluir también la biblioteca correspondiente:

```
#include <Wire.h>  // Biblioteca para la comunicacion I2C (SD y la RTClib)
#include "RTClib.h"  //Biblioteca RTC de Adafruit
#include <SD.h>      //Biblioteca para memoria SD
#include <LiquidCrystal.h>   //Biblioteca para trabajar con LCD
```

Teniendo incluidas las bibliotecas se procede a la asignación de nombres para los elementos que lo requieren, en el caso del reloj de tiempo real se nombra "rtc", para la pantalla LCD

como “lcd”, se trabajará con un archivo guardado en la memoria microSD sobre el cual se registran las bitácoras, para este caso se nombró “myFile”:

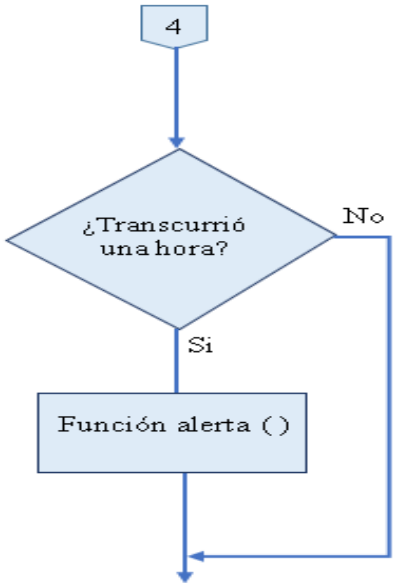
```
LiquidCrystal lcd(2, 3, 6, 7, 9, 10);  // const int rs = 2, en = 3, d4 = 6, d5 = 7, d6 = 9, d7 = 10;
RTC_DS3231 rtc;  // Se define nombre para los metodos del RTC DS3231
File myFile;   //Declaramos archivo
```

En la asignación de nombre del LCD se definen los pines con los que se trabajará, como se puede observar se usaron los pines 2, 3, 6, 7, 9 y 10 del Arduino conectados a los pines 4, 6, 11, 12, 13 y 14 del display, apoyándose de la tabla 2.1.

Para realizar las diferentes funciones del sistema fue necesario usar banderas indicadoras, mismas que apoyarán en el manejo de la información, además se deben definir variables globales debido a que son necesarias en diferentes partes del programa, tanto en el loop (programa principal) como en las funciones creadas.

```
unsigned long  tiempo1 = 0;
unsigned long tiempo2 =0 ;
float valorUV;
int valor = 0;
int horaIni = 0;
int minutoIni = 0;
int horaExpo = 0;
int minutoExpo = 0;
int banderaIni = 0;
```

Como se mencionó anteriormente una función de alerta se activa cada hora con la finalidad de recordar al usuario que el dispositivo se encuentra encendido, representado en el siguiente extracto del diagrama de flujo:




Para ello fueron creadas las variables *tiempo1* y *tiempo2*, las cuales van a ser comparadas para revisar si al realizar su diferencia ha transcurrido 1 hora.

El uso de las otras 7 variables se puede observar en la siguiente tabla:

**Tabla 3.1 Variables globales del programa**

| NOMBRE | TIPO DE VARIABLE | SIGNIFICADO |
|---|---|---|
| valorUV | Float | Valor del índice UV medido |
| valor | Int | Valor más grande medido |
| horaIni | Int | Hora de inicio de la medición |
| minutoIni | Int | Minuto de inicio de la medición |
| horaExpo | Int | Horas de exposición de la sesión |
| minutoExpo | Int | Minutos de exposición de la sesión |
| banderaIni | Int | Bandera que nos indicará hay un proceso iniciado |

Una vez declaradas las variables globales a utilizar, se puede comenzar a programar la función setup( ).

Es en esta parte del programa donde se van a definir como entrada o salida los pines del microcontrolador con los que funcionan los botones, el sensor y los elementos de alerta (un vibrador y un buzzer que trabajan a 5 volts), además se inicializa el LCD.

Para realizar el control de las alertas que se mandarán cada hora, se puede hacer uso de una función de Arduino llamada millis( ), la cual regresa el valor de los milisegundos que lleva activado el microcontrolador, de esta manera al estar monitoreando regularmente dos variables diferentes en las cuales se guarden los resultados de millis( ), de logra saber si han transcurrido 3600000 milisegundos que equivalen a una hora, para ello se asigna el primer valor a la variable "tiempo1" en esta sección del programa.

```
void setup()
{
  tiempo1 = millis();
  pinMode(5, OUTPUT); //Declaramos el pin de alertas como salida
  pinMode(A0, INPUT); //Declaramos el pin del sensor como entrada
  pinMode(A2, INPUT); //Declaramos el pin del boton inicio entrada
  pinMode(8 , INPUT); //Declaramos el pin del boton fin como entrada
  lcd.begin(8, 2);    //Inicializamos el LCD
```

En la primera sección del diagrama de flujo se observa que para asegurar el correcto funcionamiento de los elementos del sistema y su comunicación con el microcontrolador se emplea la estructura de control *if*, de manera que si no son detectados se enviará un mensaje de error en la pantalla y el proceso será detenido, únicamente estando todo en orden el programa podrá seguir con su curso, dando el mensaje de bienvenida antes de ingresar a la función loop(), quedando de la siguiente manera:

```
if (! rtc.begin())  // Comprobamos conexion con el RTC
{
    lcd.setCursor(0, 0);
    lcd.print("Error:  ");
    lcd.setCursor(0, 1);
    lcd.print("en RTC  ");
    delay(200);
    while (1)
    {
    }
}

if (!SD.begin(4))  //Se comprueba conexión con la memoria microSD
{
    lcd.setCursor(0, 0);
    lcd.print("Error en");
    lcd.setCursor(0, 1);
    lcd.print("MICRO SD");
    delay(200);
    while (1)
    {
    }
}
```

```
if(!SD.exists("datalog.csv"))  //Comprobamos exitencia de archivo
{
    myFile = SD.open("datalog.csv", FILE_WRITE); // Creacion del archivo
    if (myFile)                    //Comprobamos creaciÓn del archivo
    {
        myFile.print("Olvido");
        myFile.print(',');
        myFile.print("IUV");
        myFile.print(',');
        myFile.print("A&o");
        myFile.print(',');
        myFile.print("Mes");
        myFile.print(',');
        myFile.print("Dia");
        myFile.print(',');
        myFile.print("Hora inicial");
        myFile.print(',');
        myFile.print("Minuto inicial");
        myFile.print(',');
        myFile.print("Hora final");
        myFile.print(',');
        myFile.print("Minuto final");
        myFile.print(',');
        myFile.print("Horas expo");
        myFile.print(',');
        myFile.print("Minutos expo");
        myFile.println("");
        myFile.close();
    }

        else
        {
            lcd.setCursor(0, 0);
            lcd.print("Error de");
            delay(2000);
            lcd.setCursor(0, 1);
            lcd.print("archivo ");
            while (1)
            {
            }
        }
    }

    lcd.setCursor(0, 0);     //Mensaje indicando que los componentes fueron reconocidos
    lcd.print("BIEN-");
    lcd.setCursor(0, 1);
    lcd.print("VENIDO");
    delay(3000);
}
```

Todo este proceso se realiza solo en una ocasión cada que se enciende el dispositivo, si todos los componentes del sistema se encuentran funcionando de manera normal se podrá dar inicio a una sesión.

Cuando el radiómetro comienza a funcionar entra en un bucle que se repite una y otra vez hasta que el interruptor corte el suministro de alimentación, apagando el sistema, es importante recordar que el microprocesador tiene un funcionamiento secuencial, las acciones que realice dependerán del orden en el que se hayan programado, por lo tanto, es necesario detallar el correcto orden de las instrucciones para asegurar su óptima operación.

El inicio de este proceso se identifica en la siguiente parte del programa:

```
void loop()
```

La primera acción a realizar es leer los datos del reloj de tiempo real, la instrucción correspondiente será la expuesta en el marco teórico:

```
DateTime now = rtc.now();
```

Ello con la finalidad de verificar que el horario es el adecuado para iniciar, tal como se mostró en el diagrama de flujo, se tienen dos opciones en esa toma de decisión, si se encontrará en el horario adecuado, el programa seguirá con la secuencia de operación, de lo contrario se invocará la función apagar( ), y resultará imposible iniciar una medición. En el caso de que se verifique un horario correcto se comenzará a revisar la bandera de inicio, una variable auxiliar con la que es posible apoyarse para evitar un intento de tomar dos mediciones en una misma sesión, ya que, si se quiere intentar una segunda medición, se desea que el radiómetro envíe un error en el display, notificando que ya se encuentra en curso un trabajo y será necesario finalizarlo con el botón correspondiente, el cual se encarga de guardar los datos en la memoria externa. En cualquiera de los dos posibles estados de esa bandera de inicio, el sistema monitoreará si el botón de inicio de medición se encuentra activado, si aún no hay una medición previa, el tiempo que este botón sea presionado, se estarán tomando reiteradas mediciones, para ser promediadas y comparadas entre sí buscando el valor mayor, al mismo tiempo se estará enviando el mensaje “Tomando lecturas”, en el momento en el que se deje de presionar el botón se guardarán los datos de fecha, hora e IUV, para posteriormente ser enviados a la bitácora, además se tomará el valor de millis( ), para realizar el proceso de

medición del tiempo que enviará la alerta cada 60 minutos y se activará la bandera de inicio, ello se programó de la siguiente manera:

```
if((now.hour()<=6) || (now.hour()>=20))
{
    apagar();
}
else
{
    if (banderaIni == 0 )
    {
        if (digitalRead(A2) == 1)   //revision de estado del boton de inicio de lectura
        {
            lcd.setCursor(0, 0);
            lcd.print("Tomando ");
            lcd.setCursor(0, 1);
            lcd.print("Lecturas");

            while (digitalRead(A2) == HIGH)
            {
                medirUV();
                if ( valorUV > valor )
                {
                    valor = valorUV;
                }
            }

            DateTime now = rtc.now();
            horaIni = now.hour();
            minutoIni = now.minute();
            tiempo1 = millis();
            banderaIni = 1;
        }
```

Mientras no haya un trabajo iniciado y el botón para lecturas no sea presionado, el sistema informará que se está esperando por la medición; además, se dará la hora actualizada como una utilidad pensada para el usuario, el RTC va a proporcionar esa información, de manera que si enviamos los datos directamente al display, si el horario contempla un valor entre el minuto 1 y 9, se recibiría inmediatamente a la derecha el último carácter mostrado en esa posición en un mensaje anterior, por lo tanto, para ese caso, se debe imprimir un “0” y después el minuto, cuando el dato sea igual o mayor a 10, si se debe enviar directamente, pues se estarían llenando los dos caracteres dedicados a dicha información, quedando así su programación:

```
else
{
 lcd.setCursor(0, 0);
 lcd.print("En       ");
 lcd.setCursor(3, 0);
 lcd.print(now.hour(), DEC);
 lcd.print(":");
 int minut = now.minute();
  if (minut <= 9)
    {
     lcd.print(0);
     lcd.print(minut);
    }
   else
    {
     lcd.print(now.minute(), DEC);
    }
 lcd.setCursor(0, 1);
 lcd.print("espera   ");
 delay(170);

}
```

Una vez tomada una lectura, la bandera de inicio será igual a 1, es decir, se activará, para así enviar el mensaje de error al presionar otra vez el botón de inicio sin antes finalizar, esto se incluye en una función llamada trabajo( ), y se observa en el siguiente extracto de programa:

```
else //Si hay un trabajo iniciado
{
    if (digitalRead(A2) == 1)
    {
    trabajo();
    }
}
```

Así como se revisa permanentemente el estado del botón de inicio, de igual manera es necesario saber cuándo el botón fin se presiona, ya que debe haber acciones dedicadas a cada estado de ese botón, si no es presionado, el microcontrolador deberá mostrar el mensaje "Operando" para que el usuario sepa que se está esperando la finalización de dicha sesión, en el caso de que no se concluya la actividad antes de las 8:00 p.m., horario en que ya no hay un índice UV, se deberá implementar la función de olvido, que consiste en realizar el autoguardado de los datos, programado de la siguiente manera:

```
if (digitalRead(8) == LOW) //Si el boton fin no es presionado
 {
  lcd.setCursor(0, 0);
  lcd.print("Operando");
  DateTime now = rtc.now();
  lcd.setCursor(0,1);
  lcd.print("  ");
  lcd.setCursor(6,1);
  lcd.print("  ");
  lcd.setCursor(2,1);
  lcd.print(now.hour(), DEC);
  lcd.print(":");
  int minut = now.minute();
   if (minut <= 9)
    {
     lcd.print(0);
     lcd.print(minut);
    }
   else
    {
     lcd.print(now.minute(), DEC);
    }
   delay(50);
 }

 if((now.hour()>=19) && (now.minute() >=59))
 {
     olvido();
 }
```

Al final del "loop", se revisará el estado del botón "fin" una vez más; de manera que si es presionado, se revisará si la bandera de inicio está activada, de ser así se realizará el guardado de los datos mediante la función guardar( ), en caso contrario se estará notificando al usuario del error, pues no habría datos qué guardar en la memoria externa, el mensaje que se mandaría al display es "Error: no hay trabajo".

```
if (digitalRead(8) == 1) //Si el boton fin es presionado
{
    if (banderaIni == 1)
    {
        guardar();
    }
    else
    {
        noTrabajo();
    }
```

Como se puede observar en los diagramas de flujo, cada función o subrutina cumple diferentes tareas, se separan del programa principal para facilitar su manejo, ya que, de no separarse en subrutinas, se tendría un programa muy grande y su análisis se complicaría, al presentarse algún error se tendrían que revisar todas las líneas de código para identificar el problema. Un simple bloque representa varias líneas de código para lograr obtener una tarea, por ejemplo, el más repetido, "Mostrar Error: no hay trabajo" se logra con la siguiente fracción del programa:

```
lcd.setCursor(0, 0);
lcd.print("Error:   ");
lcd.setCursor(0, 1);
lcd.print("          ");
delay(2000);
lcd.setCursor(0, 0);
lcd.print("No hay   ");
lcd.setCursor(0, 1);
lcd.print("Trabajo ");
delay(2500);
```

## 3.5 Diseño de PCB

La tarjeta de circuito impreso brinda la seguridad de no tener problemas de funcionamiento por alguna desconexión de componentes, además, permite distribuir correctamente los diferentes componentes del sistema para reducir el tamaño en la mayor medida posible, por ello es indispensable lograr un buen diseño. Tomando en cuenta que el dispositivo será empleado por el usuario de manera cotidiana, se debe considerar que el radiómetro tiene que ser lo compacto y fácil de emplear, los elementos necesitan estar distribuidos de la mejor manera posible, el cargador y la memoria externa requieren un espacio suficiente y facilidad de conexión, el sensor debe apuntar directamente al sol, y mientras eso sucede el botón dedicado a la toma de lecturas ser pulsado por el usuario cómodamente.

Para diseñar la placa de circuito impreso se utilizó el software EAGLE (ilustración 3.3):


*Ilustración 3.3 AUTODESK EAGLE.*

Como se mencionó en el marco teórico, en el proyecto se empleó la placa experimental de Arduino para realizar las pruebas, toda la experimentación se llevó a cabo con el uso de dicha tarjeta y protoboard, pero en la tarjeta de circuito impreso se empleó únicamente el microcontrolador ATMEGA328p, y los componentes necesarios para su correcto funcionamiento, que se muestran en la ilustración 3.4.

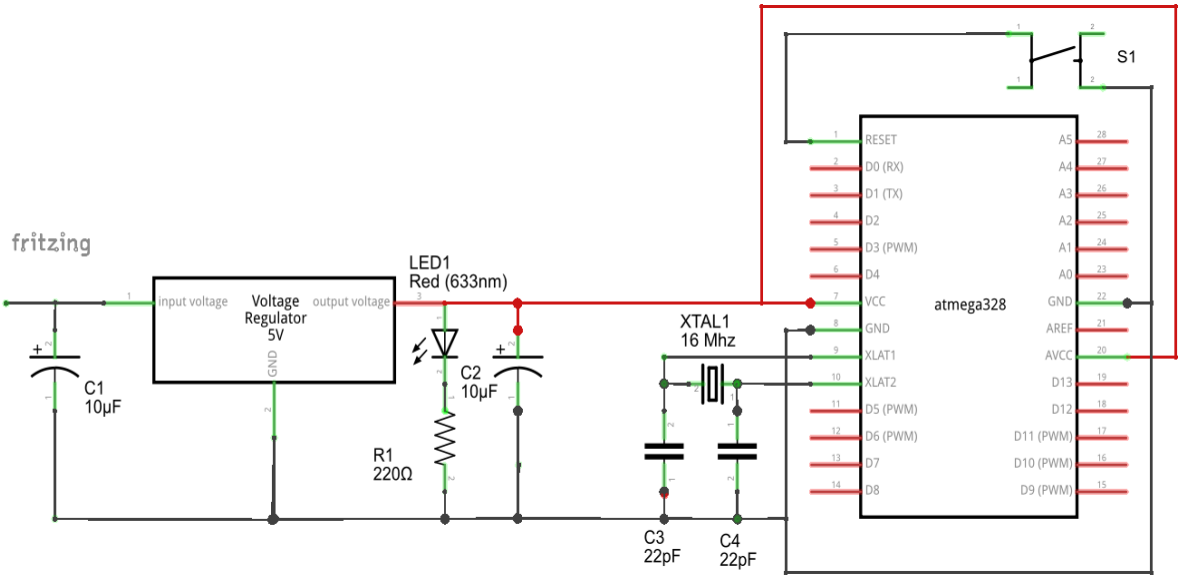


*Ilustración 3.4 ATMEGA328p*.

Para poder cargar un programa en el microcontrolador se requiere del programador USBasp. El programa debe ser cargado al microcontrolador antes de ser instalado, por lo cual éste debe ser removible de la placa para poder realizar algún cambio en el código, por ello debe colocarse en la placa PCB una base socket o zócalo como el de la ilustración 3.5, además de las posibles complicaciones que conlleve soldar directamente el ATMEGA328p:

*Ilustración 3.5 Base zócalo para C.I. de 28 pines*.

Teniendo el programa que hará funcionar el sistema de manera correcta, ya probado perfectamente en la experimentación con protoboard, es posible obtener el archivo con extensión .hex, mismo que podrá ser cargado con las herramientas descritas anteriormente, para ello existen diversas aplicaciones, como puede ser USBasp, un software sencillo de usar, que posee una interfaz muy cómoda y además facilita la instalación de los controladores que permitirán a nuestra PC reconocer el grabador, ahorrando el tiempo de búsqueda de dichos drivers en la red, como es el caso de otros software con los que se debe identificar el modelo del grabador y localizar el controlador correcto.

Considerando la totalidad de componentes que serán empleados, los cuales se muestran en la ilustración 3.6, se comienza a identificar el tipo de conexión de cada uno de ellos, pues como es el caso del microcontrolador, algunos deben ser removibles de la placa y otros deben ser fijos.

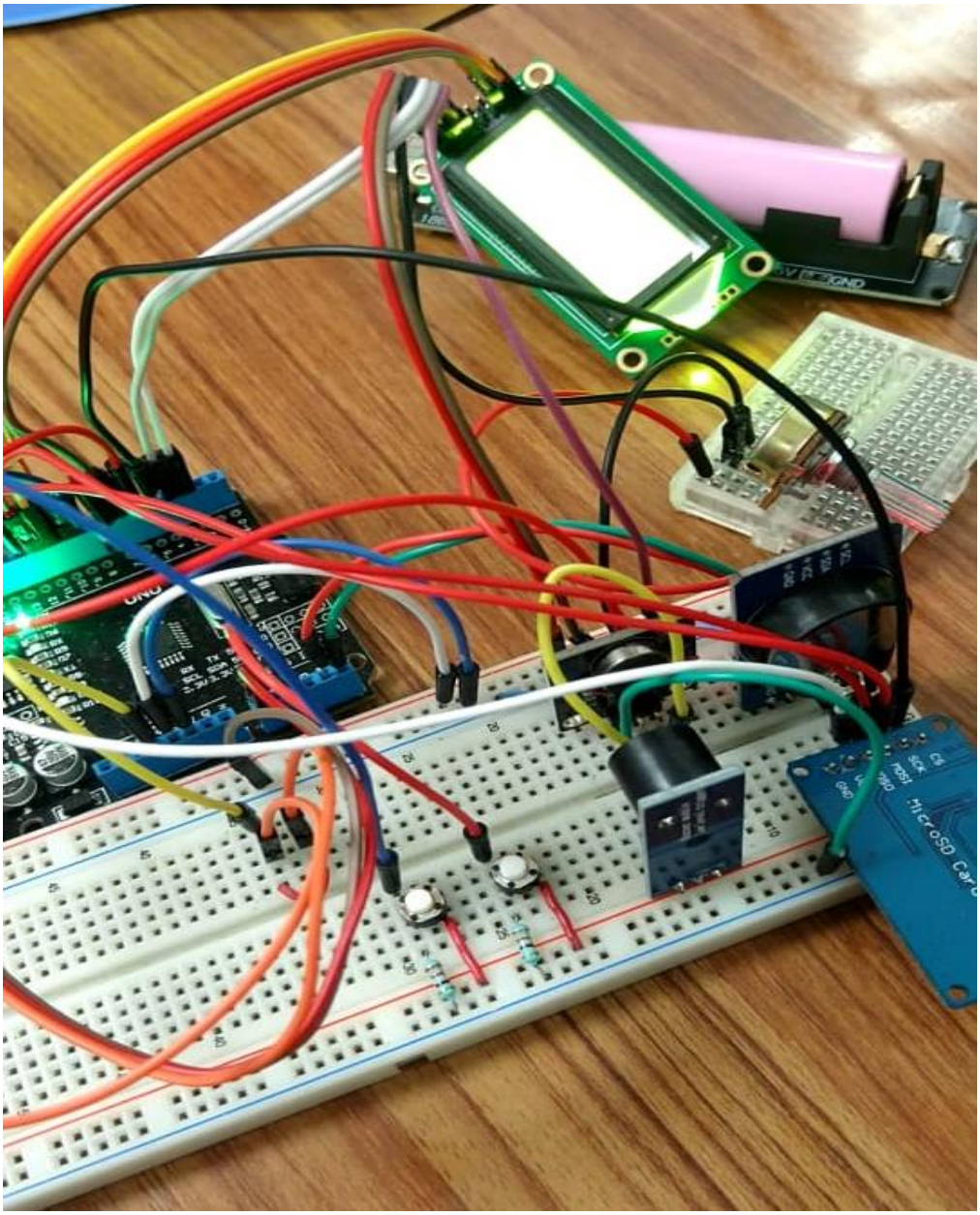



*Ilustración 3.6 Elementos del sistema.*

Los elementos y su conexión se enlistan en la tabla 3.2.

**Tabla 3.2 Elementos del sistema y conexión**

| NOMBRE | TIPO DE CONEXIÓN |
|---|---|
| LCD | Extraíble |
| Microcontrolador | Extraíble |
| Módulo de alimentación | Extraíble |
| Vibrador | Fijo |
| Buzzer | Fijo |
| RTC | Extraíble |
| Botones | Extraíble |
| Módulo de memoria externa | Fijo |

Los elementos de conexión fija son soldados directamente a la placa, mientras que los que serán extraíbles se conectan mediante pines header hembra o macho, como los que se muestran en la ilustración 3.7, mismos que aseguran una correcta fijación.

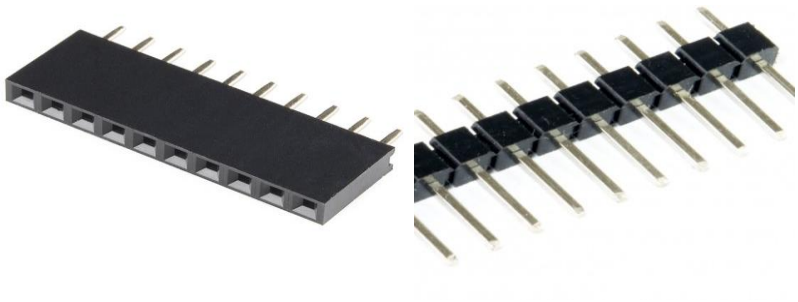


*Ilustración 3.7 Pines header.*

Esto se debe considerar para realizar el diseño de la placa de circuito impreso, en el software utilizado es muy sencillo localizar los elementos, por ejemplo, estos conectores se encuentran con su mismo nombre, pin header, tal como se puede observar en la ilustración 3.8:

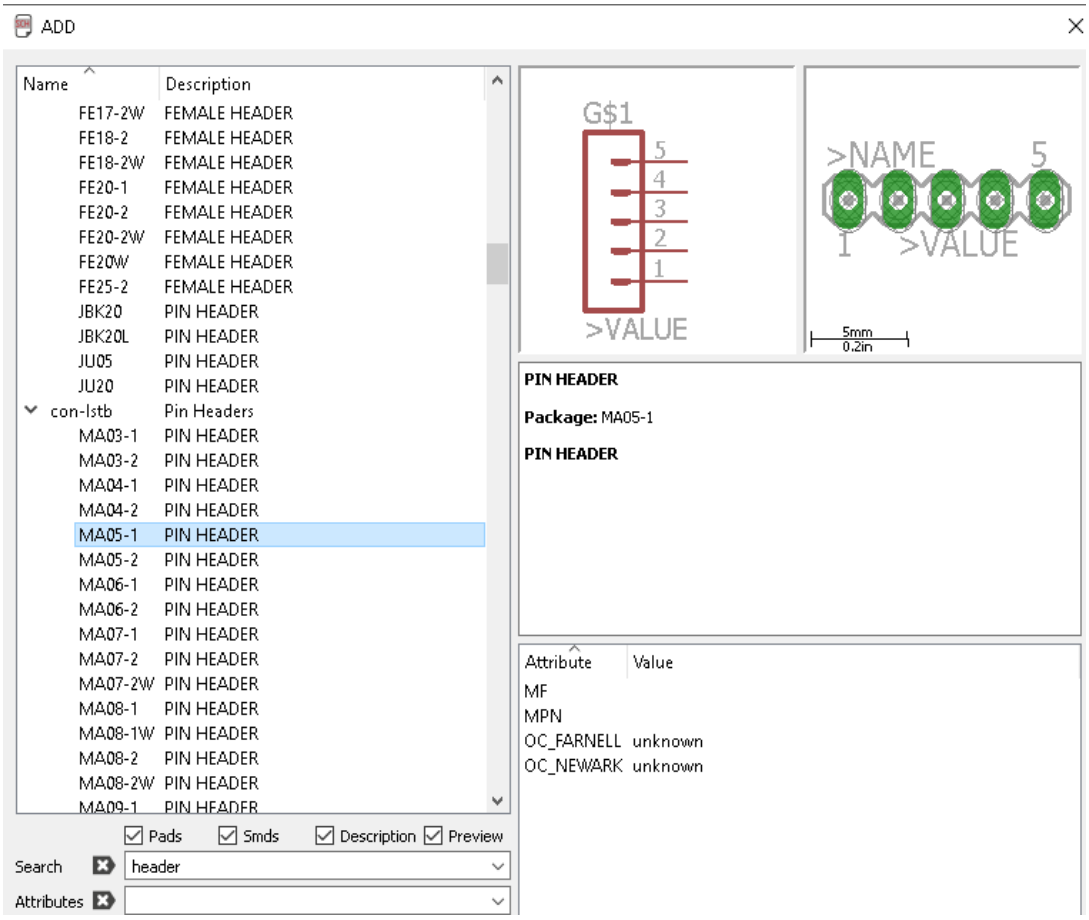



*Ilustración 3.8 Pin header en EAGLE.*

En la ilustración 3.6 hay un potenciómetro conectado entre la tarjeta Arduino y el LCD, éste no se incluye en la PCB para este caso, debido a que la función que cumple es regular el contraste de la pantalla, conectado al pin 3, marcado como $V_0$ en la tabla 2.1, en las pruebas con un potenciómetro se determinó que la resistencia para una luz posterior adecuada fue de 1KΩ, por lo que se usó un resistor de ese valor.

La fuente de alimentación es una tarjeta prefabricada que cuenta con un puerto para batería Li-ion modelo 18650, reguladores de voltaje a 5 V y 3.3 V e incorpora pines headers para la conexión de ambos voltajes, debido a eso será conectada a la PCB con cables.

Los botones no pueden ser conectados directamente a la placa, ya que no serían fáciles de presionar, necesitan estar en la carcasa donde se instalará el circuito, para ser cómodamente presionados al estar en la misma superficie, motivo por el cual también deberán ser conectados mediante cable.

Los elementos incorporados en el proyecto creado en Eagle se muestran en el diagrama esquemático de la ilustración 3.9.

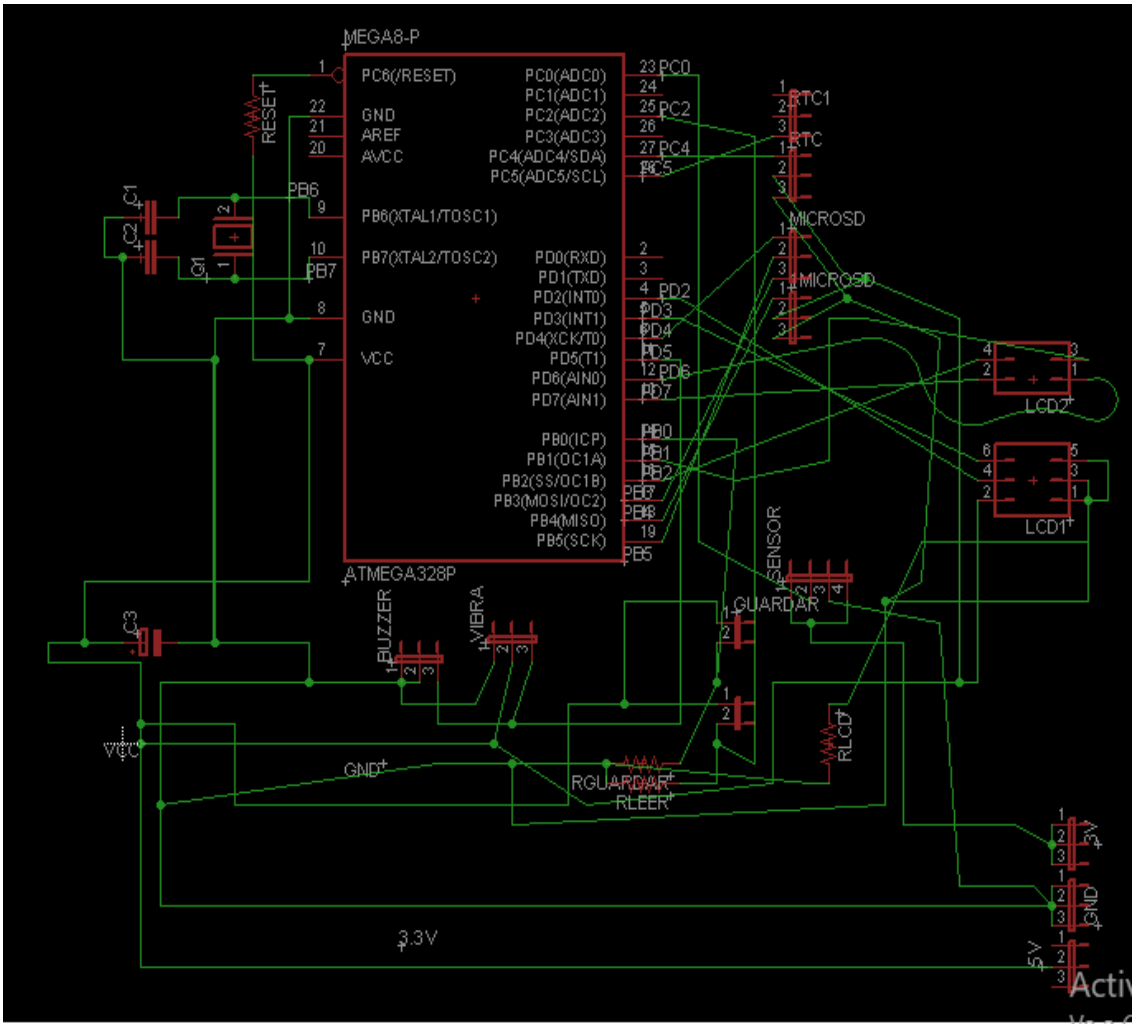



*Ilustración 3.9 Diagrama esquemático en Eagle*.

Se pueden visualizar los diversos elementos con su nombre para identificar cada uno de ellos, los que van soldados directamente a la placa se encuentran por su símbolo, mientras que los que son extraíbles de la placa se localizan por pines header y su respectivo nombre. Después de numerosos intentos por diseñar la placa se decidió separar en dos conjuntos de pines el LCD, por ello se encuentra LCD1 y LCD2, debido a que un conjunto de pines son conexiones a la fuente de alimentación y el otro conjunto son pines de conexión, que se encuentran en lados opuestos del microcontrolador y se crean muchos cruces de pistas complicando bastante hacer el acomodo de las mismas considerando placa de una solo cara.

El circuito final, se muestra en la ilustración 3.10.

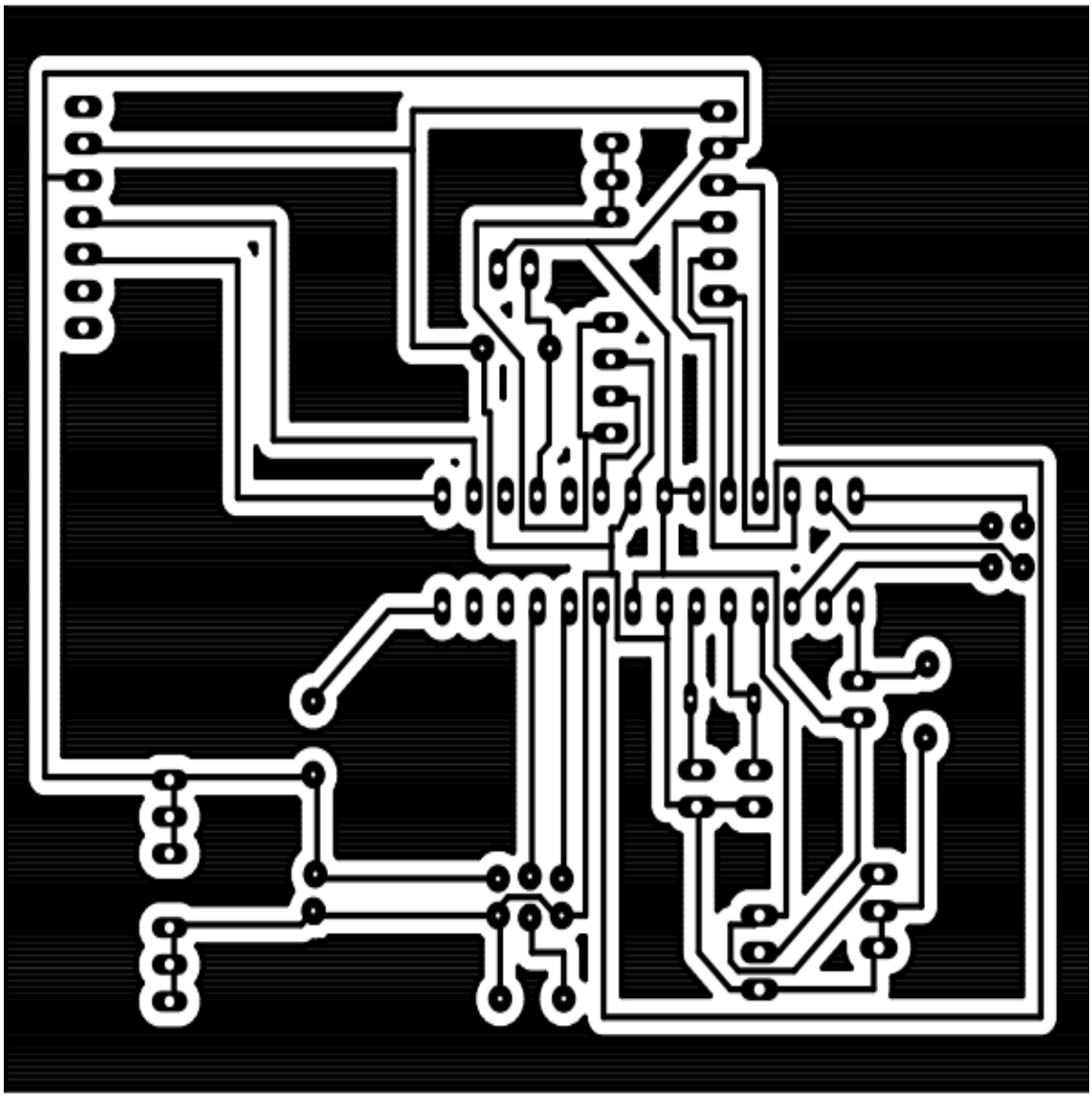


*Ilustración 3.10 Diseño final de la PCB.*

## 3.6 Fabricación de PCB

Con el diseño de la PCB final se procedió a la fabricación, recordando que algunos elementos serán soldados directamente a la placa y otros requieren de headers para su conexión, el software contempla las medidas exactas de los componentes, por lo cual no se tiene problemas a la hora de soldar por algún fallo en el espacio, de cualquier manera, es conveniente medir los elementos en alguna impresión del diseño, revisando que todo coincida

perfectamente. Asegurando el correcto acomodo de los componentes en el circuito, dentro de los diversos métodos de fabricación, como lo son: impresión serigráfica, impresión con insoladora, impresión con fresadora CNC, Solder Mask UV, entre otros, se decidió llevarlo a cabo mediante el método del planchado, ya que es una manera sencilla y económica de obtener una buena PCB, con excelente definición en las pistas y cada elemento del diseño, como se muestra en la ilustración 3.11, sin necesidad de herramientas especiales, aprovechando que la PCB será de una cara, ya que de lo contrario este método no es útil.

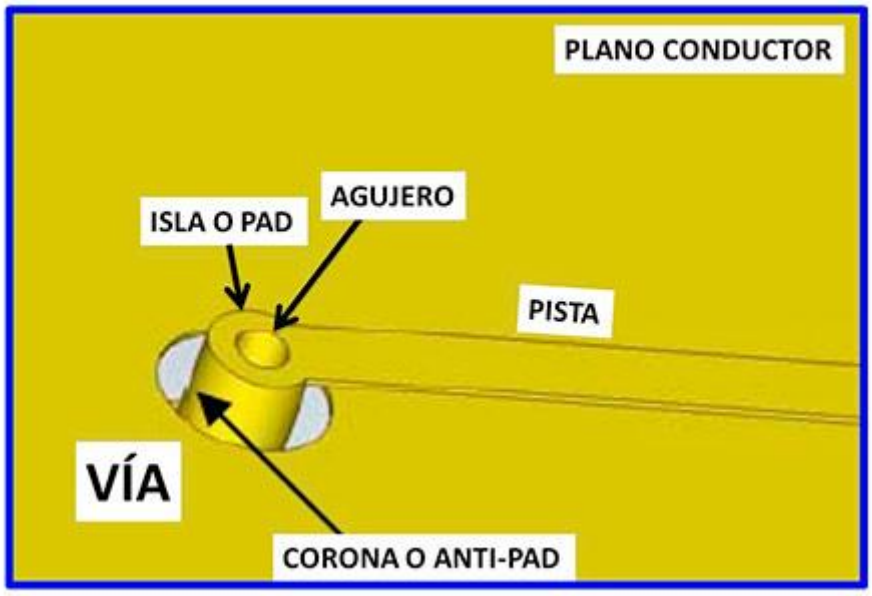



*Ilustración 3.11 Elementos del diseño.*

realizando el proceso de manera correcta se podrá obtener un resultado como el de la ilustración 3.12:

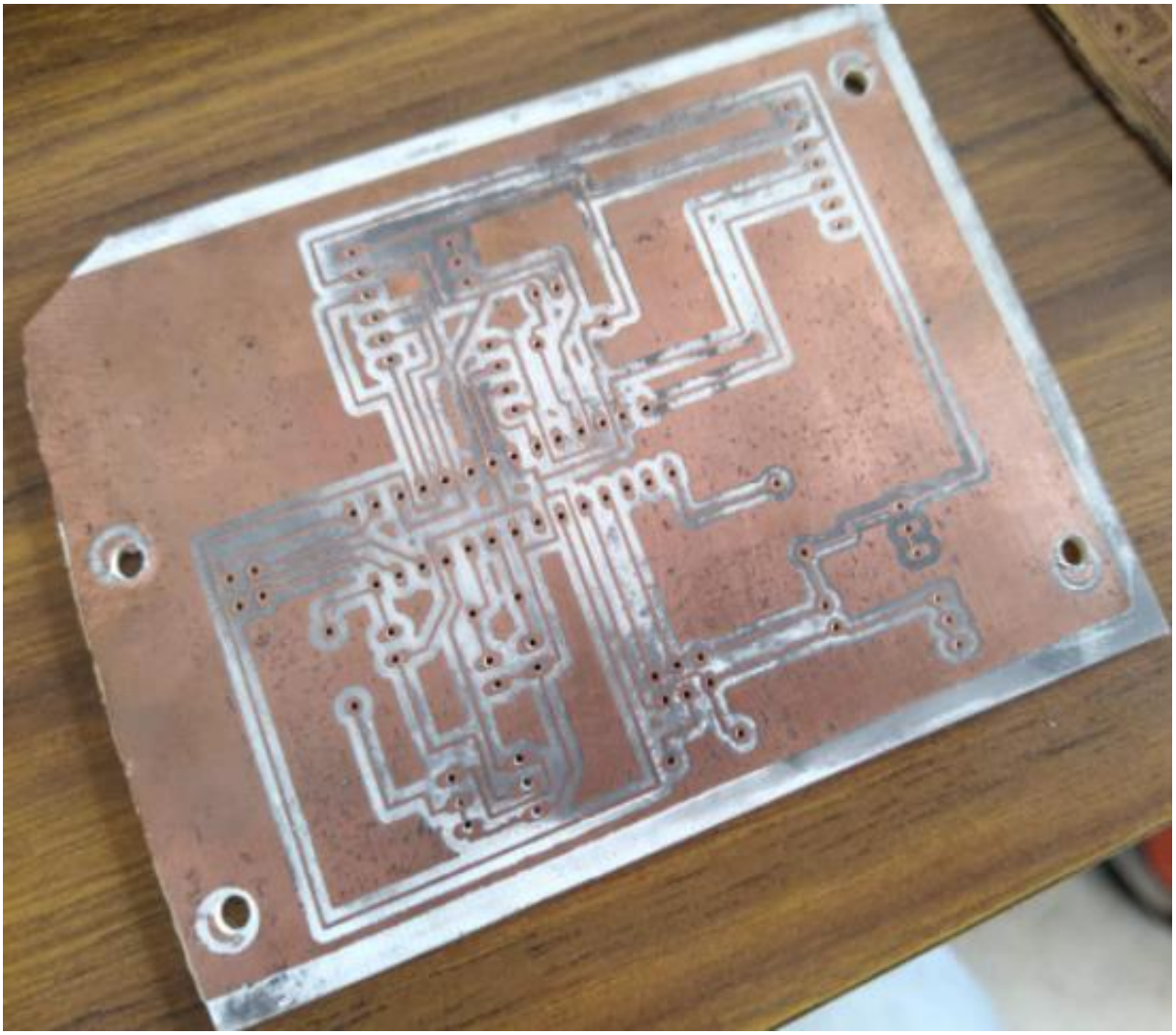


*Ilustración 3.12 PCB.*

Como se puede observar en la ilustración anterior hay unos orificios, mismos que se realizan en los pads (observar la ilustración 3.11) con una herramienta rotativa, como lo es un mini taladro, o un mototool con base, para tener un mejor soporte en la placa y asegurar orificios correctos.

En los orificioss realizados se tienen que introducir los componentes del sistema para poder ser soldados, y obtener una placa finalizada, que resulta como la que se expone en la ilustración 3.13:

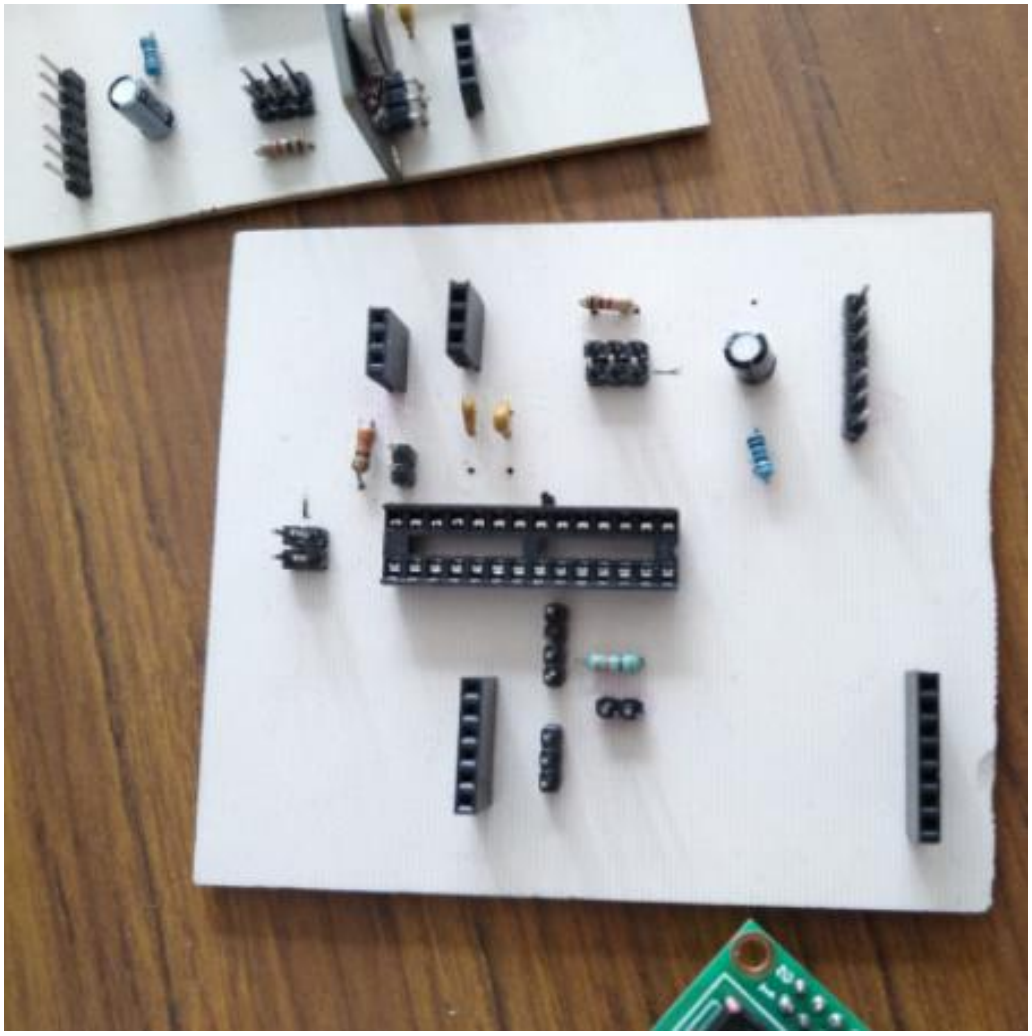


*Ilustración 3.13 PCB y componentes soldados.*

Los botones, el interruptor y el display no serán colocados en la PCB, a diferencia de los demás componentes, por esa razón es recomendable realizar placas pequeñas para soldar ahí los elementos mencionados y tener una correcta fijación a la carcasa, agregando un adhesivo como silicón o algún otro material que facilite la tarea asegurando rigidez, dichas PCB comprenderán únicamente de el botón y dos pines header para poder hacer la conexión con cable.

El display no requiere fabricación de una PCB, debido a que éste ya la trae incorporada, de manera que se puede instalar la totalidad de elementos como se muestra en la ilustración 3.14, para proceder a probar el circuito finalizado y asegurar un correcto funcionamiento.

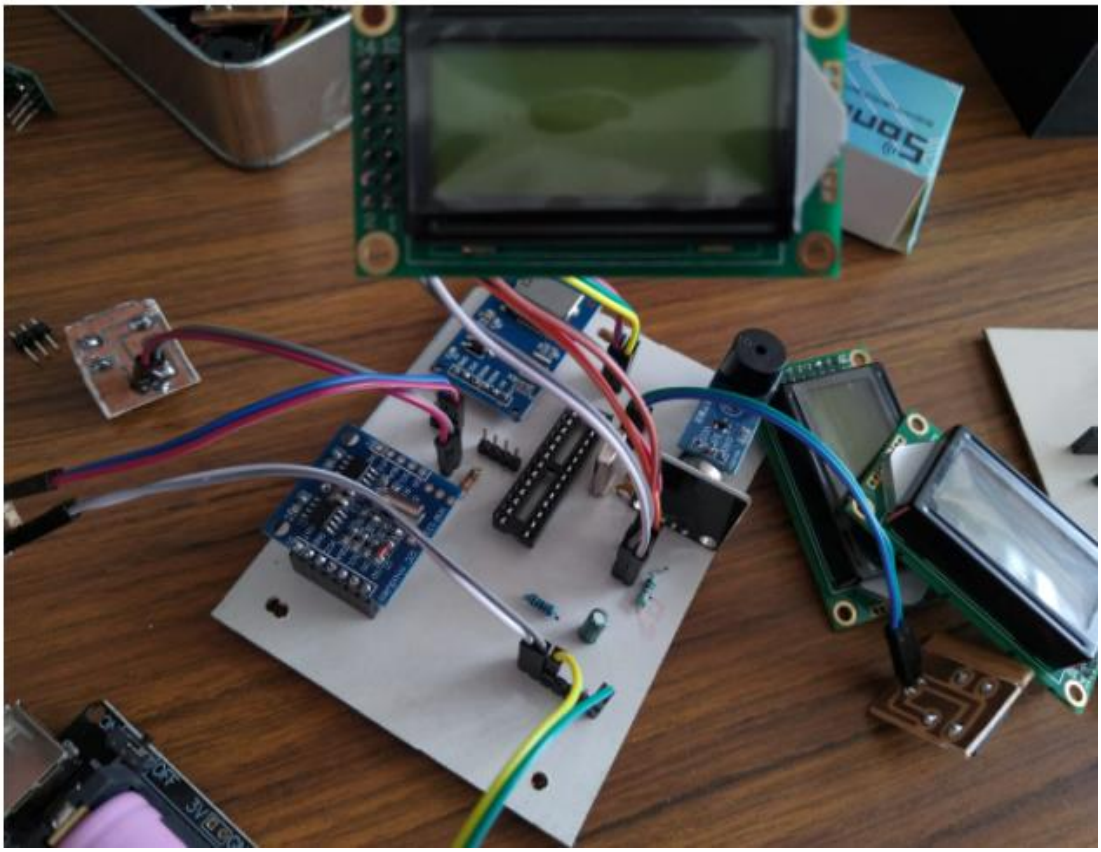


*Ilustración 3.14 Circuito final*.

## 3.7 Diseño de carcasa.

La carcasa protectora o armazón del sistema debe cumplir una serie de requerimientos, estrictamente denotados por la funcionalidad del dispositivo, necesita contar con orificios para la extracción y conexión de los elementos que así lo requieran, el radiómetro va a estar en contacto con la intemperie, se expondrá directamente al sol, puede llegar a estar bajo la lluvia si las condiciones climáticas llegaran a encontrarse así, por lo cual es indispensable proteger la circuitería casi herméticamente, de manera que por la parte superior únicamente puedan ser pulsados los botones, cambiar de posición el interruptor y ser visible el LCD, para ello se deben dejar los orificios exactos para incluir dichos elementos en el material con el que se fabricará la protección.

El diseño 3D consiste en utilizar un software para crear una representación matemática de un objeto o una forma tridimensional.

El diseño se imprime comúnmente con un material llamado filamento PLA en una impresora 3D, la impresión finalizada resulta en un modelo sumamente resistente, bien definido y completamente idéntico al que se logra en el software.

Como se puede observar en la ilustración 3.15, se dejaron los orificios correspondientes a la entrada del cargador y de la memoria MicroSD en los costados de la caja.

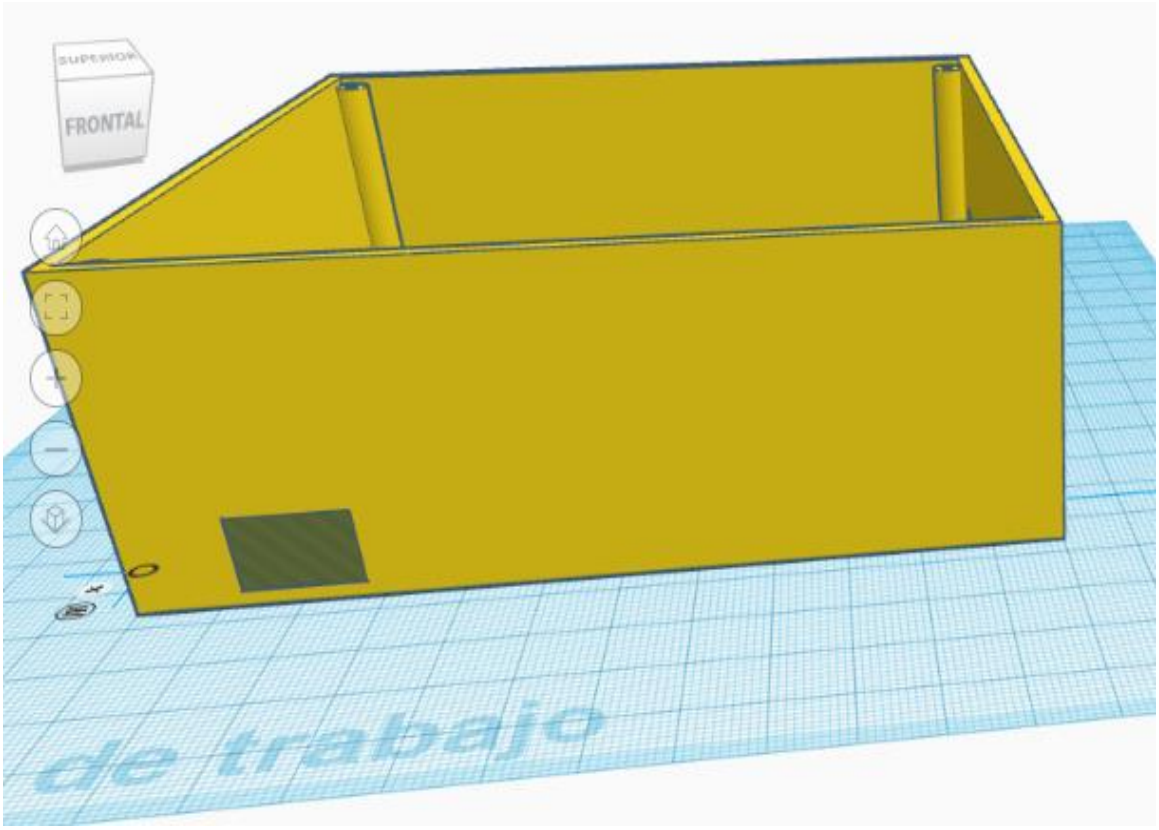



*Ilustración 3.15 Diseño de carcasa.*

Además, se incluyeron en las esquinas cuerpos cilíndricos (observar la ilustración 3.16) donde entran los tornillos que sujetan la tapa, de la misma manera son asegurados la PCB y el módulo de carga de la batería, para no tener ningún juego dentro de la carcasa.

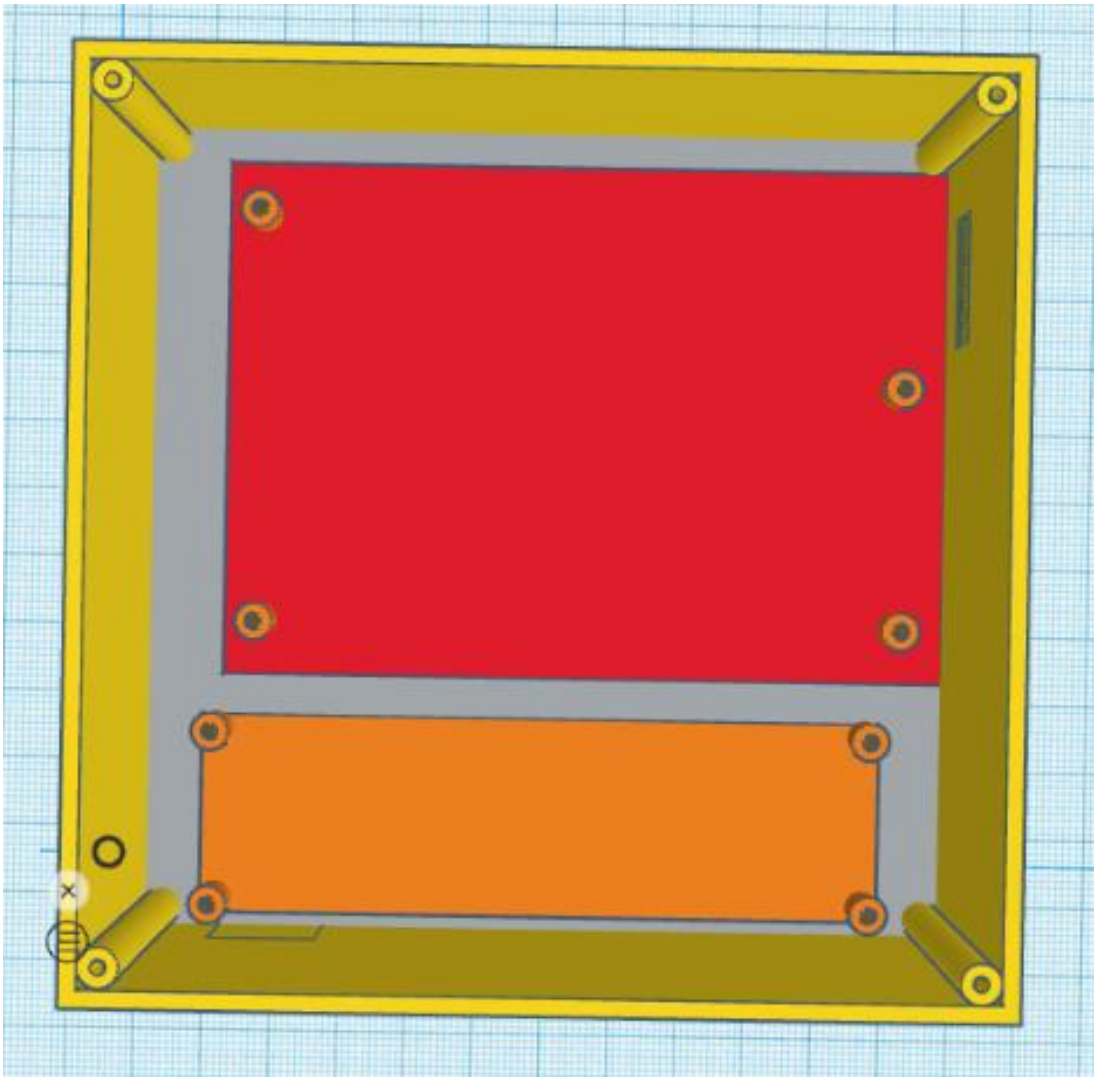


*Ilustración 3.16 Orificios en el armazón.*

La tapa de la ilustración 3.17 contempla la instalación de los elementos con los que el usuario interactúa directamente, por ello se observan los espacios correspondientes donde serán insertados, para estar asegurados y no correr el riesgo de ser arrojados al interior del armazón al ejercer presión sobre ellos se emplea silicón termofusible, pues tiene mucha adherencia con el material utilizado para la impresión. De igual manera en las 4 esquinas se localizan los orificios para la instalación de los tornillos de fijación.

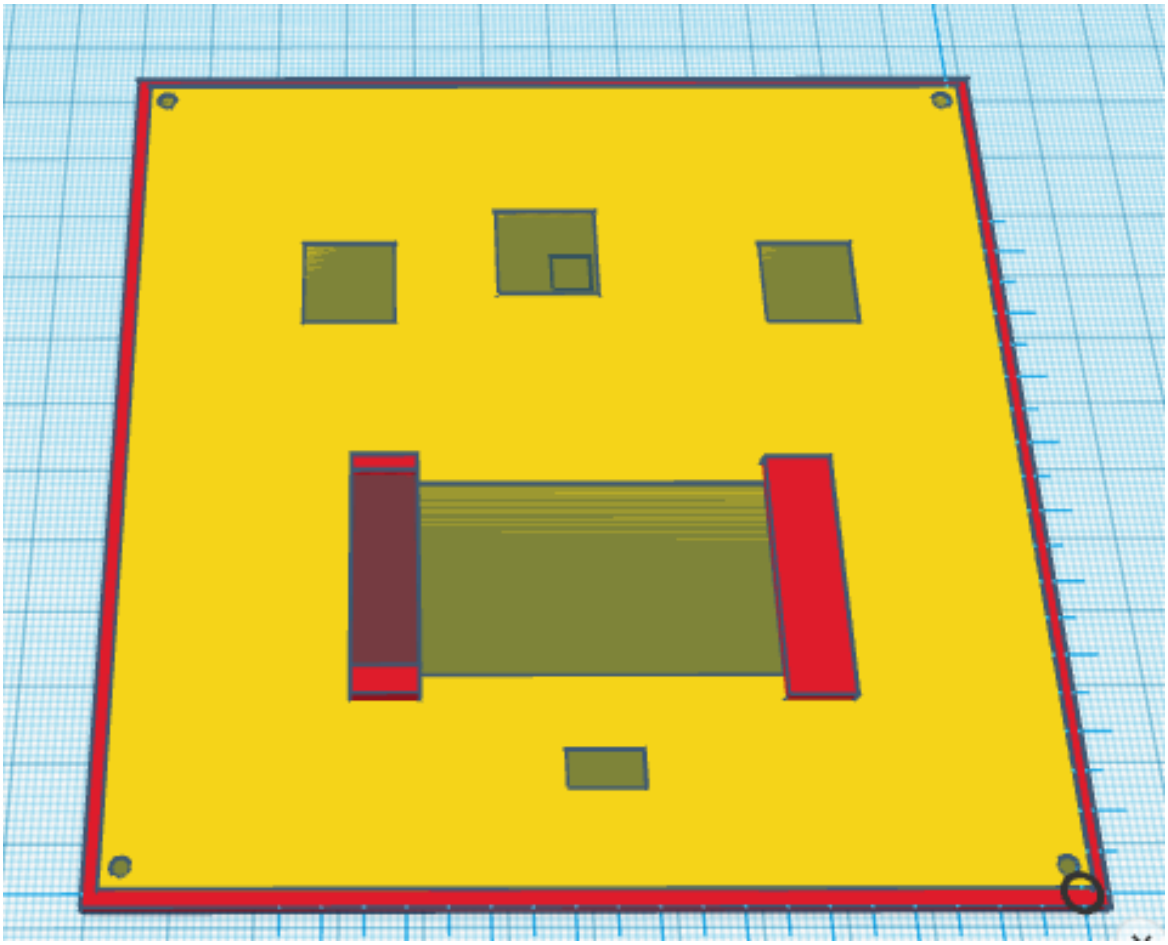


*Ilustración 3.17 Diseño de tapa.*

Se incluyeron también las palabras que identifican a cada botón, “INICIO” y “FIN”, para facilitar el manejo del dispositivo, se muestra en la ilustración 3.18.

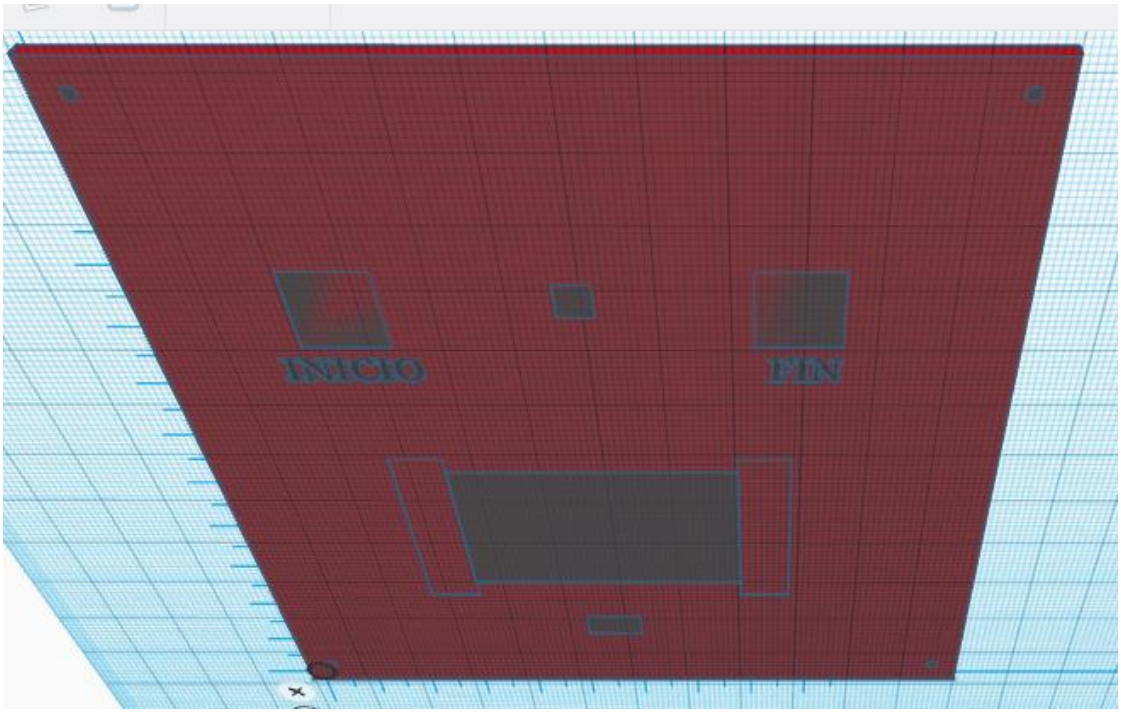



*Ilustración 3.18 Palabras identificadoras de los botones.*

La impresión finalizada se muestra en la ilustración 3.19 y al instalar los diferentes elementos del sistema para poder cerrar la carcasa, se obtiene un radiómetro concluido como el de la ilustración 3.20.

+

*Ilustración 3.19 Carcasa impresa.*

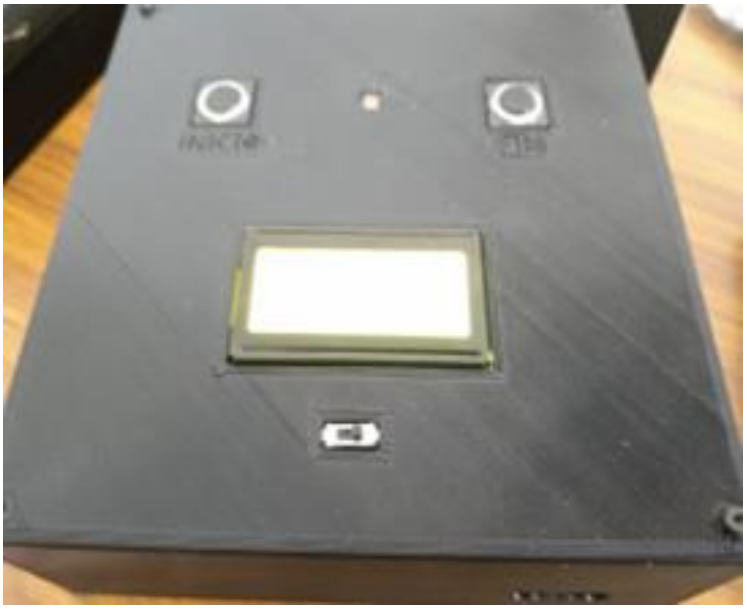


*Ilustración 3.20 Radiómetro concluido.*

# 4 Pruebas y resultados

## 4.1 RTC

Se encontró en repetidas ocasiones un problema con el RTC, a la hora de cargar el programa con la fecha y la hora actual, y al incorporarlo al radiómetro se perdía dicha información, retornando a su valor por default. Se procedió a cambiar las pilas y se probó de manera independiente el módulo RTC; sin embargo, el problema persistía, después se encontró que la causa de ese desajuste ocurría cuando tras cargar nuevamente la fecha y hora actual al módulo, se conectaba al sistema sin quitar la polarización de la tarjeta, por lo cual es importante que cada vez que se inserte un módulo RTC al sistema, asegurarse que no se encuentre polarizado.

## 4.2 Módulo de memoria MicroSD

Al igual que en el caso del RTC, la desconexión del módulo al tener el sistema polarizado provocaba la pérdida de la información en la memoria externa.

## 4.3 LCD

El display empleado es cómodo y sencillo de usar; sin embargo, se debe contemplar el número de caracteres, para evitar tener palabras incompletas, al igual que cuidar realizar el borrado de caracteres antes de escribir una nueva palabra, por ello se realizó la configuración correspondiente antes de mostrar la hora, ya que, si el número de minutos es inferior a 10, quedaría un carácter del mensaje previo. Contemplando dichas observaciones fue posible mandar todos y cada uno de los mensajes de manera correcta.

## 4.4 Funcionamiento del sistema

Las funciones incluidas en el programa presentaron diversas fallas a lo largo de la programación en las pruebas, que se fueron solucionando gracias a la modificación de los diagramas de flujo hasta conseguir un correcto funcionamiento del programa, el cual se define en los diagramas antes expuestos, el resultado final, fue el que se grabó en los microcontroladores de los radiómetros realizados, debido a su funcionamiento adecuado, además no se encontró algún error.

Al iniciar el radiómetro se verifica el correcto funcionamiento de los elementos, en caso de fallar alguno de ellos se muestra en la pantalla el aviso correspondiente, por ejemplo, si la memoria no se detecta se mostrará el mensaje de la ilustración 4.1.

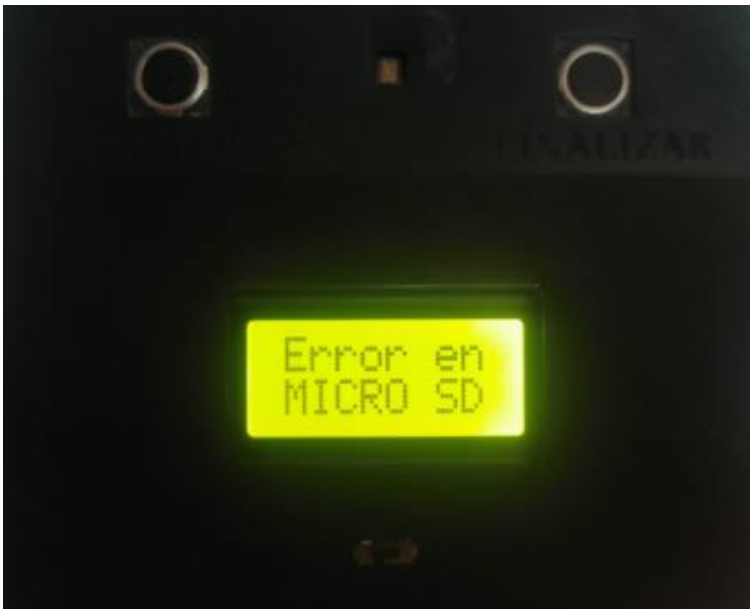



*Ilustración 4.1 Error en MicroSD.*

De igual manera se verifica el horario para saber si se encuentra dentro del rango establecido, en caso de no ser así se notificará, como en la ilustración 4.2a y 4.2b.


*Ilustración 4.2a y 4.2b Mensajes de la Función olvido.*

Si el microcontrolador detecta que todo se encuentra en orden dará un mensaje de bienvenida, como en la ilustración 4.3.


*Ilustración 4.3 Inicio correcto*.

Posterior a ello, es posible iniciar una medición, por lo tanto, se muestra el mensaje que indica que se está en espera de la medición, como se ve en la ilustración 4.4, y al presionar el botón inicio se observa el mensaje de la ilustración 4.5.


*Ilustración 4.4 En espera.*


*Ilustración 4.5 Medición.*

Una vez tomada la medición, el sistema estará en espera de que finalice para guardar la información correspondiente, y eso se sabe al observar el mensaje de la ilustración 4.6, pues se está realizando una operación.


*Ilustración 4.6 Trabajo iniciado.*

Cuando se finaliza la sesión, el sistema guarda en la bitácora el registro de la información, y se muestra un mensaje como el de la ilustración 4.7.


*Ilustración 4.7 Finalización del trabajo.*

Si se presiona el botón inicio por segunda vez sin antes haber finalizado una primera medición con el botón fin, se estará cometiendo un error, y eso invocaría la función trabajo( ), avisando la existencia de un trabajo en curso, como se muestra en la ilustración 4.8, y si se presiona el botón fin sin tener una medición previamente tomada se llamará a la función noTrabajo( ), como se puede observar en la ilustración 4.9.


*Ilustración 4.8 Trabajo en curso.*


*Ilustración 4.9 No hay trabajo.*

## 4.5 Mediciones

Se realizaron pruebas comparativas entre las mediciones del IUV del sistema propuesto y el sensor comercial Sentry ST513. La tabla 4.1 y la ilustración 4.10, son una muestra de las mediciones realizadas -de manera simultánea con ambos sistemas- antes de incorporar las rutinas de programación para mejora de precisión.

**Tabla 4.1 Comparación de mediciones sin ajuste.**

| MUESTRA | ML8511 | SENTRY | DIFERENCIA |
|---|---|---|---|
| 1 | 3.1 | 1.8 | 1.3 |
| 2 | 3.3 | 1.9 | 1.4 |
| 3 | 4.4 | 3 | 1.4 |
| 4 | 4.4 | 3.5 | 0.9 |
| 5 | 4.6 | 3.6 | 1 |
| 6 | 4.6 | 3.6 | 1 |
| 7 | 4.6 | 3.7 | 0.9 |
| 8 | 4.7 | 3.7 | 1 |
| 9 | 4.7 | 3.9 | 0.8 |
| 10 | 4.9 | 3.9 | 1 |
| 11 | 5.3 | 3.9 | 1.4 |
| 12 | 5.5 | 4 | 1.5 |

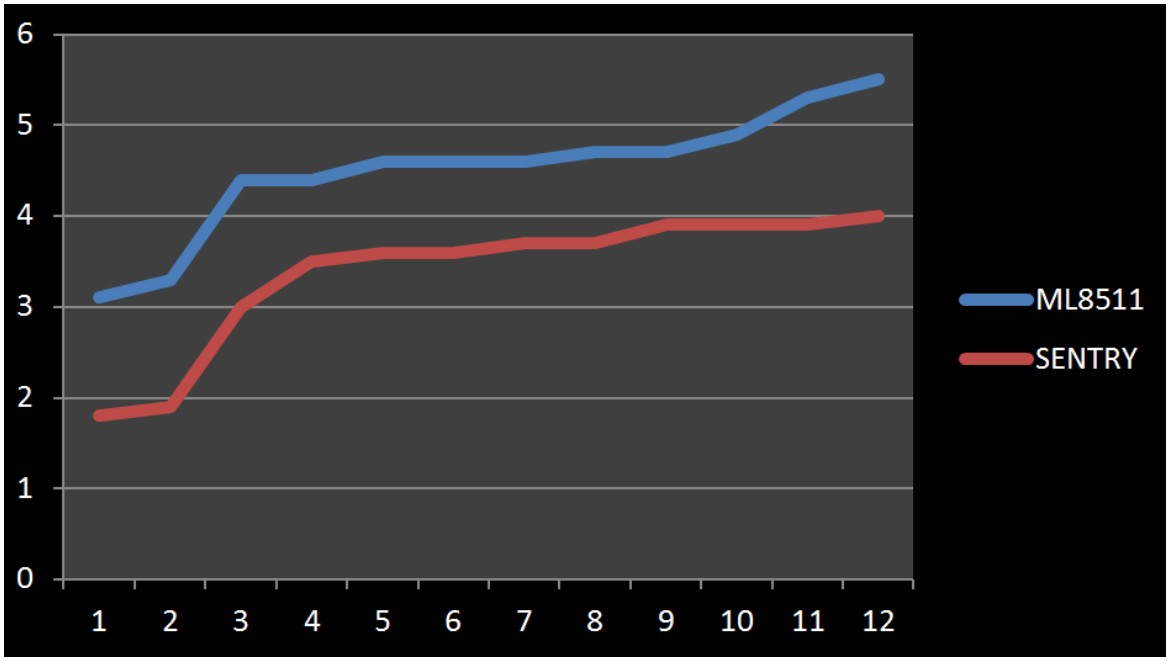



*Ilustración 4.10 Gráfica de las mediciones mostradas en la tabla 4.1.*

Se puede observar claramente una diferencia entre las mediciones, que para los datos mostrados es de 1.13 [IUV]. Al considerar el intervalo de la variable de estudio con un valor mínimo de cero y un máximo de quince, el porcentaje promedio de diferencia es de 7.5%.

La tabla 4.2 y la ilustración 4.11 son una muestra de mediciones realizadas de manera simultánea, con las rutinas de programación para mejora de precisión.

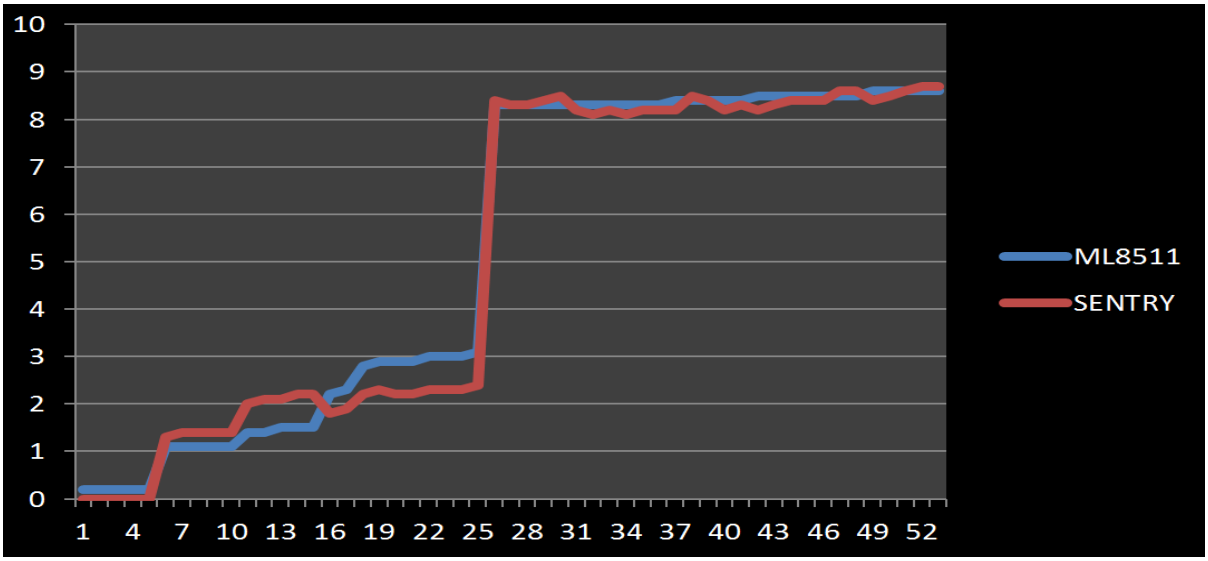



*Ilustración 4.11Gráfica de las mediciones mostradas en la tabla 4.2.*

**Tabla 4.2 Comparación de mediciones con ajuste.**

| MUESTRA | ML8511 | SENTRY | DIFERENCIA | MUESTRA | ML8511 | SENTRY | DIFERENCIA |
|---|---|---|---|---|---|---|---|
| 1 | 0.2 | 0 | 0.2 | 27 | 8.3 | 8.3 | 0 |
| 2 | 0.2 | 0 | 0.2 | 28 | 8.3 | 8.3 | 0 |
| 3 | 0.2 | 0 | 0.2 | 29 | 8.3 | 8.4 | 0.1 |
| 4 | 0.2 | 0 | 0.2 | 30 | 8.3 | 8.5 | 0.2 |
| 5 | 0.2 | 0 | 0.2 | 31 | 8.3 | 8.2 | 0.1 |
| 6 | 1.1 | 1.3 | 0.2 | 32 | 8.3 | 8.1 | 0.2 |
| 7 | 1.1 | 1.4 | 0.3 | 33 | 8.3 | 8.2 | 0.1 |
| 8 | 1.1 | 1.4 | 0.3 | 34 | 8.3 | 8.1 | 0.2 |
| 9 | 1.1 | 1.4 | 0.3 | 35 | 8.3 | 8.2 | 0.1 |
| 10 | 1.1 | 1.4 | 0.3 | 36 | 8.3 | 8.2 | 0.1 |
| 11 | 1.4 | 2 | 0.6 | 37 | 8.4 | 8.2 | 0.2 |
| 12 | 1.4 | 2.1 | 0.7 | 38 | 8.4 | 8.5 | 0.1 |
| 13 | 1.5 | 2.1 | 0.6 | 39 | 8.4 | 8.4 | 0 |
| 14 | 1.5 | 2.2 | 0.7 | 40 | 8.4 | 8.2 | 0.2 |
| 15 | 1.5 | 2.2 | 0.7 | 41 | 8.4 | 8.3 | 0.1 |
| 16 | 2.2 | 1.8 | 0.4 | 42 | 8.5 | 8.2 | 0.3 |
| 17 | 2.3 | 1.9 | 0.4 | 43 | 8.5 | 8.3 | 0.2 |
| 18 | 2.8 | 2.2 | 0.6 | 44 | 8.5 | 8.4 | 0.1 |
| 19 | 2.9 | 2.3 | 0.6 | 45 | 8.5 | 8.4 | 0.1 |
| 20 | 2.9 | 2.2 | 0.7 | 46 | 8.5 | 8.4 | 0.1 |
| 21 | 2.9 | 2.2 | 0.7 | 47 | 8.5 | 8.6 | 0.1 |
| 22 | 3 | 2.3 | 0.7 | 48 | 8.5 | 8.6 | 0.1 |
| 23 | 3 | 2.3 | 0.7 | 49 | 8.6 | 8.4 | 0.2 |
| 24 | 3 | 2.3 | 0.7 | 50 | 8.6 | 8.5 | 0.1 |
| 25 | 3.1 | 2.4 | 0.7 | 51 | 8.6 | 8.6 | 0 |
| 26 | 8.3 | 8.4 | 0.1 | 52 | 8.6 | 8.7 | 0.1 |

Como se puede apreciar las diferencias entre ambos sistemas se redujo, observándose un promedio de diferencia de tan solo 0.28 [IUV], que en términos de porcentaje es de 1.91 %.

También se obtuvieron las facilidades para hacer pruebas comparativas contra un radiómetro de una estación de monitoreo de la Secretaria del Medio Ambiente del Sistema de Monitoreo Atmosférico de la Ciudad de México. En la tabla 4.3 se presenta una muestra de las lecturas realizadas de manera simultánea, entre el sistema de este proyecto y el radiómetro de la estación. La tabla 4.3 y la ilustración 4.12 muestran los resultados.

**Tabla 4.3 Comparación de mediciones del sistema propuesta vs estación de monitoreo.**

| MUESTRA | ML8511 | ESTACIÓN | DIFERENCIA | MUESTRA | ML8511 | ESTACIÓN | DIFERENCIA |
|---|---|---|---|---|---|---|---|
| 1 | 2.6 | 2.5 | 0.1 | 26 | 8.4 | 8.6 | 0.2 |
| 2 | 2.7 | 2.6 | 0.1 | 27 | 8.4 | 8.6 | 0.2 |
| 3 | 3.5 | 3.6 | 0.1 | 28 | 9.4 | 9.5 | 0.1 |
| 4 | 3.7 | 3.8 | 0.1 | 29 | 9.8 | 9.7 | 0.1 |
| 5 | 3.7 | 3.8 | 0.1 | 30 | 9.8 | 9.5 | 0.3 |
| 6 | 3.9 | 4 | 0.1 | 31 | 9.9 | 9.7 | 0.2 |
| 7 | 3.9 | 4 | 0.1 | 32 | 10 | 9.9 | 0.1 |
| 8 | 3.9 | 4 | 0.1 | 33 | 10 | 9.9 | 0.1 |
| 9 | 4 | 4.1 | 0.1 | 34 | 10 | 9.8 | 0.2 |
| 10 | 4 | 4.2 | 0.2 | 35 | 10 | 9.7 | 0.3 |
| 11 | 4.1 | 4.2 | 0.1 | 36 | 10 | 9.7 | 0.3 |
| 12 | 4.1 | 4.2 | 0.1 | 37 | 10.1 | 9.9 | 0.2 |
| 13 | 4.1 | 4.2 | 0.1 | 38 | 10.1 | 9.9 | 0.2 |
| 14 | 4.1 | 4.3 | 0.2 | 39 | 10.1 | 9.8 | 0.3 |
| 15 | 4.2 | 4.4 | 0.2 | 40 | 10.2 | 10.5 | 0.3 |
| 16 | 4.3 | 4.6 | 0.3 | 41 | 10.3 | 10.2 | 0.1 |
| 17 | 4.3 | 4.5 | 0.2 | 42 | 10.3 | 10.1 | 0.2 |
| 18 | 4.4 | 4.7 | 0.3 | 43 | 11.3 | 11.5 | 0.2 |
| 19 | 4.5 | 4.8 | 0.3 | 44 | 11.4 | 11.1 | 0.3 |
| 20 | 4.6 | 4.9 | 0.3 | 45 | 11.8 | 12.1 | 0.3 |
| 21 | 5.5 | 5.8 | 0.3 | 46 | 11.9 | 11.8 | 0.1 |
| 22 | 5.6 | 5.7 | 0.1 | 47 | 11.9 | 12.1 | 0.2 |
| 23 | 5.7 | 5.9 | 0.2 | 48 | 12 | 12.2 | 0.2 |
| 24 | 8.2 | 8.5 | 0.3 | 49 | 12.1 | 11.9 | 0.2 |
| 25 | 8.4 | 8.6 | 0.2 | 50 | 12.3 | 12 | 0.3 |

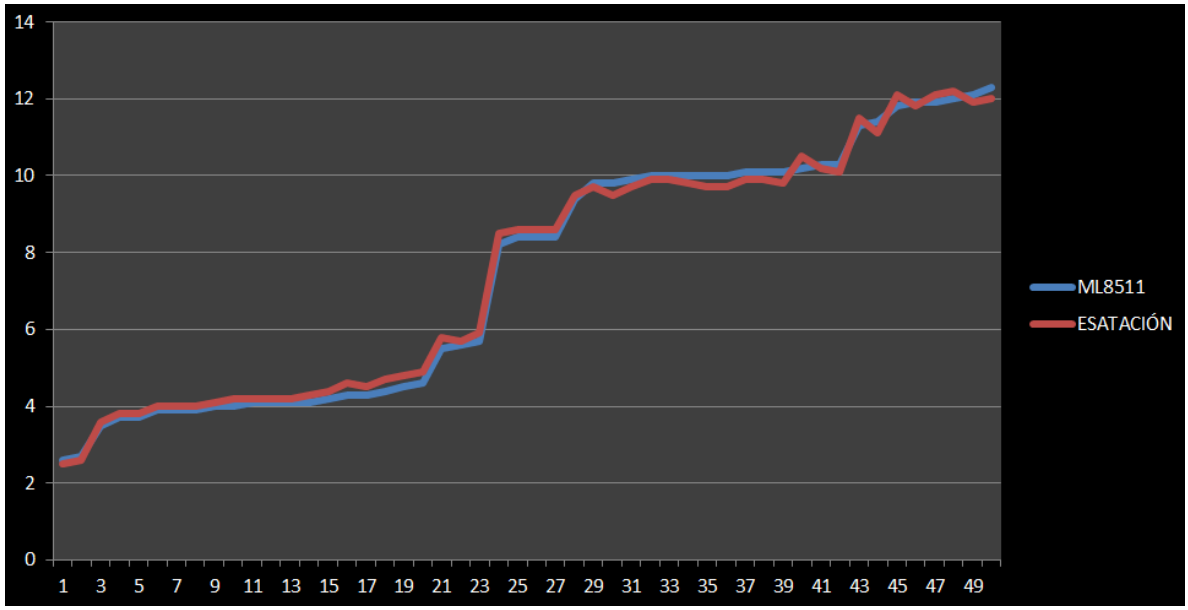



*Ilustración 4.12 Gráfica de las mediciones mostradas en la tabla 4.3.*

Como se puede apreciar las diferencias entre ambos sistemas es reducida, presentando un promedio de diferencia de tan solo 0.2 [IUV], que en términos de porcentaje es de 1.26 %.

Teniendo esa baja diferencia entre las mediciones comparadas se pueden considerar confiables los valores arrojados por el radiómetro de esta propuesta.

Finalizando la programación y cargándola al microcontrolador fue posible obtener bitácoras correctas, donde se logran observar los datos requeridos por el personal médico, mismos que fueron mencionados anteriormente, el archivo guardado en la memoria microSD se puede visualizar en cualquier editor de hojas de cálculo, el más popular es Excel, con el cual se observan registros como el de la tabla 3.6, con la información completa en cada medición, incluyendo la función de olvido( ), en la última línea de dicha tabla se aprecia un “1”en la fila correspondiente, titulada `Olvido´, lo cual indica que el dispositivo no fue apagado antes de las 7:59 p.m. ni fue guardada manualmente la medición en curso (presionando el botón fin), por lo cual se realiza un autoguardado.

**Tabla 4.4 Bitácora de pruebas.**

| Olvido | IUV | Año | Mes | Día | Hora inicial | Minuto inicial | Hora final | Minuto final | Horas expo | Minutos expo |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2020 | 7 | 21 | 13 | 42 | 13 | 49 | 0 | 7 |
| 0 | 7 | 2020 | 7 | 21 | 13 | 50 | 14 | 18 | 0 | 28 |
| 0 | 6 | 2020 | 7 | 21 | 14 | 19 | 15 | 29 | 1 | 10 |
| 0 | 7 | 2020 | 7 | 21 | 15 | 30 | 15 | 55 | 0 | 25 |
| 0 | 8 | 2020 | 7 | 24 | 14 | 4 | 14 | 4 | 0 | 0 |
| 0 | 7 | 2020 | 7 | 24 | 14 | 4 | 14 | 4 | 0 | 0 |
| 0 | 1 | 2020 | 7 | 24 | 14 | 4 | 14 | 4 | 0 | 0 |
| 0 | 7 | 2020 | 7 | 24 | 14 | 5 | 15 | 13 | 1 | 8 |
| 0 | 1 | 2020 | 7 | 25 | 10 | 14 | 10 | 14 | 0 | 0 |
| 0 | 1 | 2020 | 7 | 25 | 10 | 14 | 18 | 31 | 8 | 17 |
| 1 | 0 | 2020 | 7 | 25 | 18 | 31 | 19 | 59 | 1 | 28 |
| 0 | 1 | 2020 | 7 | 29 | 14 | 23 | 14 | 39 | 0 | 16 |

# Conclusiones

Se diseñaron modelados 3D para la fabricación de la carcasa protectora que necesita cada dispositivo, logrando un armazón totalmente adecuado para el uso que se dará al radiómetro, el cual permite perfectamente la extracción y conexión de los elementos que así lo requieren, como el cargador y la memoria MicroSD, además, la tapa protege la circuitería para estar expuesto a la intemperie, permitiendo al mismo tiempo la interacción directa del usuario con los botones, switch y pantalla.

El radiómetro fabricado incluye una batería recargable con su respectivo módulo controlador de carga, de tal manera que se puede cargar cada fin de semana y la batería rendirá perfectamente para realizar el trabajo requerido, sin la necesidad de ningún reemplazo de batería.

La pantalla que se incorporó al dispositivo permite una correcta comunicación con el usuario, todos los mensajes son mostrados correctamente, es posible manejar los tiempos de cada mensaje, para evitar la pérdida de información.

La programación restringe el uso del dispositivo en horarios inadecuados, como lo es en la noche (de 8 pm a 7 am), cuando no es posible tener una exposición a la radiación UV, si el usuario intenta iniciar una sesión dentro de ese horario, en la pantalla se le indicará que es imposible realizarlo, pues se encuentra fuera de horario, hasta que se encuentre en un horario adecuado el software permitirá dar inicio a una lectura, además, mediante otra función del programa, se guardará automáticamente la medición si el usuario olvida realizarlo manualmente.

Se logró obtener mediciones confiables del índice de radiación ultravioleta, recibiendo valores en la escala estandarizada que recomienda emplear la Organización Mundial de la Salud de 1 a 11, mismas que al ser comparadas con lecturas arrojadas por radiómetros comerciales certificados, no encontramos variaciones, confirmando así la veracidad de los resultados.

El radiómetro incluye elementos (un buzzer y un vibrador) que alertarán al usuario en cada error al emplearlo, además se activan cada hora para que no olvide que está activado y debe apagarlo al finalizar una sesión, además, como una utilidad extra se presenta en todo momento la hora actualizada.

# Referencias


American Academy of Family Physicians. (12 de julio de 2017). *familydoctor.org*. Recuperado el 10 de febrero de 2020, de https://es.familydoctor.org/vitamina-d/

Anónimo. (14 de febrero de 2017). *Robótica fácil*. Recuperado el 14 de febrero de 2020, de https://roboticafacil.es/prod/sensor-uv-guva-s12sd/

Anónimo. (2019). Recuperado el 13 de febrero de 2020, de GOBIERNO DE LA CIUDAD DE MÉXICO: http://www.aire.cdmx.gob.mx/default.php?opc=%27ZaBhnmI=&dc=%27aA==

Anónimo. (10 de julio de 2019). *American Cancer Society*. Recuperado el 13 de febrero de 2020, de https://www.cancer.org/es/cancer/cancer-de-piel/prevencion-y-deteccion-temprana/que-es-la-radiacion-de-luz-ultravioleta.html

Anónimo. (14 de febrero de 2020). *MecatrónicaLATAM*. Recuperado el 14 de febrero de 2020, de https://www.mecatronicalatam.com/es/tutoriales/sensores/

Anónimo. (14 de febrero de 2020). *Naylamp mechatronics*. Recuperado el 14 de febrero de 2020, de https://naylampmechatronics.com/sensores-luz-y-sonido/169-modulo-ml8511-detector-uv.html

Anónimo. (16 de febrero de 2020). *Naylamp Mechatronics SAC*. (N. M. SAC., Productor) Recuperado el 16 de febrero de 2020, de https://naylampmechatronics.com/blog/38_Tutorial-arduino-y-memoria-SD-y-micro-SD-.html

Anónimo. (2020). *Paraquesirve.tv*. Recuperado el 22 de febrero de 2020, de https://paraquesirve.tv/interruptor/

Anónimo. (2020). *SHOPTRONICA*. Recuperado el 02 de marzo de 2020, de https://www.shoptronica.com/curiosidades-tutoriales-y-gadgets/4079-que-son-los-interruptoes-pulsadores-conmutadores-0689593950512.html

Anónimo. (17 de febrero de 2020). *tdrobótica.co*. Recuperado el 17 de febrero de 2020, de http://tdrobotica.co/sensor-de-luz-ultravioleta-uv/780.html

Autodesk Inc. (10 de febrero de 2020). *Programas de diseño 3D profesionales*. Obtenido de https://www.autodesk.mx/solutions/3d-design-software

*ECUAROBOT*. (2020). Recuperado el 2020 de 08 de 20, de http://ecuarobot.com/2019/12/23/sensor-uv-ml8511-y-arduino-para-medicion-de-intensidad-de-rayos-uv/

FANDOM. (15 de febrero de 2020). *FANDOM*. Recuperado el 15 de febrero de 2020, de https://perifericos.fandom.com/es/wiki/Memorias_extraibles

García Cobo, J. (24 de febrero de 2020). Recuperado el 24 de febrero de 2020, de HARDWARELIBRE: https://www.hwlibre.com/pantallas-lcd-arduino/

LAPIS Semiconductor. (8 de marzo de 2013). *Sparkfun*. Recuperado el 13 de febrero de 2020, de https://cdn.sparkfun.com/datasheets/Sensors/LightImaging/ML8511_3-8-13.pdf

Llamas, L. (18 de febrero de 2020). *Ingeniería, informática y diseño.* Recuperado el 18 de febrero de 2020, de https://www.luisllamas.es/reloj-y-calendario-en-arduino-con-los-rtc-ds1307-y-ds3231/

*MASTER, INNOVACIÓN QUE SE VIVE*. (18 de Febrero de 2020). Recuperado el 13 de febrero de 2020, de https://shop.master.com.mx/product/detail_ofs?id=7554

*Mecatrónica LATAM*. (2018). Recuperado el 14 de febrero de 2020, de https://www.mecatronicalatam.com/tutorial/es/sensores

Nacional Institutes of Health, O. o. (14 de agosto de 2019). *USA.gov*. Obtenido de https://ods.od.nih.gov/factsheets/VitaminD-DatosEnEspanol/

Planas, O. (13 de abril de 2017). *Energía Solar*. Recuperado el 12 de febrero de 2020, de https://solar-energia.net/que-es-energia-solar/radiacion-solar

U.S. National Library of Medicine. (20 de diciembre de 2019). *MedlinePlus*. Recuperado el 10 de febrero de 2020, de https://medlineplus.gov/spanish/vitamind.html

*U.S.A. National Library of Medicine*. (29 de mayo de 2019). Recuperado el 10 de ferbero de 2020, de https://medlineplus.gov/spanish/vitaminddeficiency.html

# Fuentes de ilustraciones

Ilustración 2.1 Radiación solar. Recopilada el 12 de febrero del 2020 de: https://image.shutterstock.com/image-illustration/natural-greenhouse-effect-human-enhanced-260nw-532639924.jpg

Ilustración 2.2 Radiación UV. Recopilada el 12 de febrero del 2020 de: https://i.ytimg.com/vi/GlK5Un-ctdI/maxresdefault.jpg

Ilustración 2.3 Potencia de los rayos UV. Recopilada el 13 de febrero del 2020 de: https://i.pinimg.com/originals/4f/03/c3/4f03c3aeecd8e12a6ac7266d82ad9ef6.gif

Ilustración 2.4 Sensores. Recopilada el 13 de febrero del 2020 de: https://shop.master.com.mx/product/detail_ofs?id=7554

Ilustración 2.5 ML8511. Recopilada el 14 de febrero del 2020 de: https://naylampmechatronics.com/536-large_default/modulo-ml8511-detector-uv.jpg

Ilustración 2.6 Fotodiodo del ML8511. Recopilada el 14 de febrero del 2020 de: http://ecuarobot.com/wp-content/uploads/2019/12/ml8511-uv-characteristics.png

Ilustración 2.7 Gráfica del ML8511. Recopilada el 14 de febrero del 2020 de: https://cdn.sparkfun.com/datasheets/Sensors/LightImaging/ML8511_3-8-13.pdf

Ilustración 2.8 GUVA-S12SD. Recopilada el 14 de febrero del 2020 de: https://images-na.ssl-images-amazon.com/images/I/41l2kGaRCsL._SX342_.jpg

Ilustración 2.9 SEN_0777. Recopilada el 14 de febrero del 2020 de: http://tdrobotica.co/3008-large_default/sensor-de-luz-ultravioleta-uv.jpg

Ilustración 2.10 Memoria micro SD. Recopilada el 15 de febrero del 2020 de: http://www.naylampmechatronics.com/img/cms/Blog/Tutorial%20SD/Tama%C3%B1os%20de%20SD.jpg

Ilustración 2.11 Módulo de memoria micro SD. Recopilada el 16 de febrero del 2020 de: http://www.naylampmechatronics.com/img/cms/Blog/Tutorial%20SD/Conexiones%20Arduino%20y%20modulo%20SD.jpg

Ilustración 2.22 Circuito abierto y circuito cerrado. Recopilada el 22 de febrero del 2020 de: https://arquimedes.matem.unam.mx/lite/desarrollo/UnADM/Finales/_Un_150_CambioDeBase/content/escenas/imgs/circ_circ_tache.png

Ilustración 2.23 Interruptor. Recopilada el 26 de febrero del 2020 de: https://amytronics.com/wp-content/uploads/2017/11/el-interruptor-por-dentro-fig1-300x207.jpg

Ilustración 2.24 Interruptor basculante. Recopilada el 26 de febrero del 2020 de: https://www.electronicaembajadores.com/datos/fotos/articulos/grandes/it/it11/it11101.jpg

Ilustración 2.25 Interruptor de pulsador. Recopilada el 26 de febrero del 2020 de: https://www.trivinet.com/img/uploads/2817.jpg

Ilustración 2.26 Interruptor rotativo. Recopilada el 26 de febrero del 2020 de: https://static.acotron.com/image/cache/data/26373_e-780x975.png

Ilustración 2.27 Interruptor deslizante. Recopilada el 26 de febrero del 2020 de: https://image.made-in-china.com/45f3j00cLhESHlFbDoP/50mA-3pin-Mini-PCB-on-off-2-Way-Slide-Switch.jpg

Ilustración 2.28 Interruptor automático o interruptor magneto-térmico. Recopilada el 26 de febrero del 2020 de: https://blogmaterialelectrico.files.wordpress.com/2013/04/material-elc3a9ctrico1.jpg

Ilustración 2.29 Reed switch. Recopilada el 26 de febrero del 2020 de: https://cdn-tienda.bricogeek.com/925-thickbox_default/reed-switch.jpg

Ilustración 2.30 Interruptor centrífugo. Recopilada el 26 de febrero del 2020 de: https://images-na.ssl-images-amazon.com/images/I/71PPJwBx1gL._SX342_.jpg

Ilustración 2.31 Interruptores DIP. Recopilada el 26 de febrero del 2020 de: https://www.neoguias.com/wp-content/uploads/2018/02/dip-switch.jpg

Ilustración 2.32 Símbolo del pulsador. Recopilada el 02 de marzo del 2020 de: https://www.diarioelectronicohoy.com/blog/imagenes/2010/11/xPulsadores.jpg.pagespeed.ic.CItcrk1XqT.webp

# Anexo

Código en Arduino:

```
#include <Wire.h>  // Biblioteca para la comunicacion I2C (SD y la RTClib)
#include "RTClib.h"  //biblioteca RTC de Adafruit
#include <SD.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(2, 3, 6, 7, 9, 10);  // const int rs = 2, en = 3, d4 = 6, d5 = 7, d6 = 9, d7 = 10;
RTC_DS3231 rtc;  // Se define nombre para los metodos del RTC DS3231
File myFile;   //Declaramos archivo
unsigned long  tiempo1 = 0;    //tiempo de referencia inicial de bandera de olvido
unsigned long tiempo2 =0 ;    //tiempo de referencia para alertas cada hora
float valorUV;   //Variable que almacenara valor UV
int valor = 0;          //valor de indice UV
int horaIni = 0;            //hora inicial
int minutoIni = 0;         //minuto inicial
int horaExpo = 0;
int minutoExpo = 0;
int banderaIni = 0;    //bandera auxiliar

////////////////////////////////////////////////////////////////////////////
void setup()
{
        tiempo1 = millis();
        pinMode(5, OUTPUT); // Declaramos el pin de la bandera de olvido como salida
        pinMode(A0, INPUT);  //Se declaran el pin del sensor como entrada
        pinMode(A2, INPUT);  // Declaramos el pin del boton inicio como entrada
        pinMode(8 , INPUT);  // Declaramos el pin del boton fin como entrada
        lcd.begin(8, 2);

        if (! rtc.begin())  // Comprobamos que se reconozca al reloj de tiempo real
```

```
{
        lcd.setCursor(0, 0);
        lcd.print("Error:  ");
        lcd.setCursor(0, 1);
        lcd.print("en RTC  ");
        delay(200);
        while (1)
        {
        }
}
if (!SD.begin(4))    //Se comprueba existencia de memoria micro SD
{
        lcd.setCursor(0, 0);
        lcd.print("Error en");
        lcd.setCursor(0, 1);
        lcd.print("MICRO SD");
        delay(200);
        while (1)
        {
        }
}
if(!SD.exists("datalog.csv"))  //Comprobamos exitencia de archivo
{
        myFile = SD.open("datalog.csv", FILE_WRITE); // Creacion del archivo
        if (myFile)              //Comprobamos creacion del archivo
        {
                myFile.print("Olvido");
                myFile.print(',');
                myFile.print("IUV");
                myFile.print(',');
                myFile.print("A&o");
                myFile.print(',');
                myFile.print("Mes");
```

```
                myFile.print(',');
                myFile.print("Dia");
                myFile.print(',');
                myFile.print("Hora inicial");
                myFile.print(',');
                myFile.print("Minuto inicial");
                myFile.print(',');
                myFile.print("Hora final");
                myFile.print(',');
                myFile.print("Minuto final");
                myFile.print(',');
                myFile.print("Horas expo");
                myFile.print(',');
                myFile.print("Minutos expo");
                myFile.println("");
                myFile.close();
        }
        else
        {
                lcd.setCursor(0, 0);
                lcd.print("Error de");
                delay(2000);
                lcd.setCursor(0, 1);
                lcd.print("archivo ");
                while (1)
                {
                }
        }
}
lcd.setCursor(0, 0);    //Mensaje indicando que los componentes fueron reconocidos
lcd.print("BIEN-");
lcd.setCursor(0, 1);
lcd.print("VENIDO");
```

```
        delay(3000);
}
////////////////////////////////////////////////////////////////////////////
void loop()
{
        DateTime now = rtc.now();    //Revisión de horario y fecha
        if((now.hour()<=6) || (now.hour()>=20)) //Si el horario no es adecuado
        {
                apagar();
        }
        else  //Si el horario es adecuado
        {
                if (banderaIni == 0 )  //Si no hay un trabajo iniciado
                {
                        if (digitalRead(A2) == 1)  //Si el boton inicio fue presionado
                        {
                                lcd.setCursor(0, 0);
                                lcd.print("Tomando ");
                                lcd.setCursor(0, 1);
                                lcd.print("Lecturas");

                                while (digitalRead(A2) == HIGH) //Mientras siga presionado
                                {
                                        medirUV();
                                        if ( valorUV > valor )  //Se conserva el valor más grande
                                        {
                                                valor = valorUV;
                                        }
                                }

                                DateTime now = rtc.now();
                                horaIni = now.hour();
                                minutoIni = now.minute();
```

```
                    tiempo1 = millis();
                    banderaIni = 1;
                }
                else  //Si el boton inicio no es presionado
                {
                    lcd.setCursor(0, 0);
                    lcd.print("En      ");
                    lcd.setCursor(3, 0);
lcd.print(now.hour(), DEC);
lcd.print(":");
int minut = now.minute();
if (minut <= 9)
 {
  lcd.print(0);
  lcd.print(minut);
 }
else
 {
  lcd.print(now.minute(), DEC);
 }
      lcd.setCursor(0, 1);
      lcd.print("espera  ");
delay(170);
                }
      }
      else //Si hay un trabajo iniciado
      {
                if (digitalRead(A2) == 1)   Si el boton inicio es presionado
                {
                trabajo();
                }
      }
      if (banderaIni == 1) //Si hay un trabajo iniciado
```

```
        {
                tiempo2 = millis();
                if ((tiempo2 - tiempo1) > 3600000) // Si ha transcurrido una hora
                {
                        alerta();
                }

if (digitalRead(8) == LOW) //Si el boton fin no es presionado
          {
  lcd.setCursor(0, 0);
  lcd.print("Operando");
          DateTime now = rtc.now();
  lcd.setCursor(0,1);
  lcd.print("  ");
  lcd.setCursor(6,1);
  lcd.print("  ");
  lcd.setCursor(2,1);
  lcd.print(now.hour(), DEC);
  lcd.print(":");
  int minut = now.minute();
   if (minut <= 9)
    {
     lcd.print(0);
     lcd.print(minut);
    }
   else
    {
     lcd.print(now.minute(), DEC);
    }
   delay(50);
 }

                if((now.hour()>=19) && (now.minute() >=59))
```

```
                {
                        olvido();
                }
        }
        if (digitalRead(8) == 1) //Si el boton fin es presionado
        {
                if (banderaIni == 1)
                {
                        guardar();
                }
                else
                {
                        noTrabajo();
                }
        }
    }
}
////////////////////////////////////////////////////////////////////////////
void guardar()
{
    lcd.setCursor(0, 0);
    lcd.print("Guardan-");
    lcd.setCursor(0, 1);
    lcd.print("do      ");
    digitalWrite(5, HIGH);
    delay(500);
    digitalWrite(5, LOW);
    lcd.setCursor(0, 0);
    int acarreo = 0;
    DateTime now = rtc.now();
    if ((now.minute() - minutoIni)<0)
    {
        minutoExpo = 60 - minutoIni + now.minute();
```

```
        acarreo = 1;
}
else
{
        minutoExpo = now.minute() - minutoIni;
        acarreo = 0;
}
horaExpo = (now.hour()- horaIni) - acarreo;
myFile = SD.open("datalog.csv", FILE_WRITE);//abrimos  el archivo
if (myFile)
{
        myFile = SD.open("datalog.csv", FILE_WRITE);//abrimos  el archivo
        myFile.print("0");
        myFile.print(',');
        myFile.print(valor);
        myFile.print(',');
        myFile.print(now.year());
        myFile.print(',');
        myFile.print(now.month());
        myFile.print(',');
        myFile.print(now.day());
        myFile.print(',');
        myFile.print(horaIni);
        myFile.print(',');
        myFile.print(minutoIni);
        myFile.print(',');
        myFile.print(now.hour());
        myFile.print(',');
        myFile.print(now.minute());
        myFile.print(',');
        myFile.print(horaExpo);
        myFile.print(',');
        myFile.print(minutoExpo);
```

```
                myFile.println("");
                myFile.close(); //cerramos el archivo
        }
        else
        {
                lcd.setCursor(0, 0);
                lcd.print("Error en");
                lcd.setCursor(0, 1);
                lcd.print("archivo ");
        }
        delay(3000);

        banderaIni = 0;
        valor=0;
}
////////////////////////////////////////////////////////////////////////////
void alerta()
{
        digitalWrite(5, HIGH);
        delay(800);
        digitalWrite(5, LOW);
        delay(300);
        digitalWrite(5, HIGH);
        delay(800);
        digitalWrite(5, LOW);
        delay(300);
        digitalWrite(5, HIGH);
        delay(800);
        digitalWrite(5, LOW);
        tiempo1 = millis();
}

////////////////////////////////////////////////////////////////////////
```

```
void medirUV()
{
        valorUV = 0;
        float valorAcumulado = 0;
        for(int x = 0 ; x < 8 ; x++)
        {
          int lectura = analogRead(A0);
          valorAcumulado += lectura;
        }
        valorAcumulado /= 8;
        float valVoltaje = valorAcumulado * 0.004887586;
        float UV = (valVoltaje-1)/.133333;
        float ajuste = 1.30;
        valorUV = UV * ajuste;
}
//////////////////////////////////////////////////////////////////////////////
void olvido()
{
        int acarreo = 0;
        DateTime now = rtc.now();
        if ((now.minute() - minutoIni)<0)
        {
                minutoExpo = 60 - minutoIni + now.minute();
                acarreo = 1;
        }
        else
        {
                minutoExpo = now.minute() - minutoIni;
                acarreo = 0;
        }

        horaExpo = (now.hour()- horaIni) - acarreo;
        myFile = SD.open("datalog.csv", FILE_WRITE);//abrimos  el archivo
```

```
myFile = SD.open("datalog.csv", FILE_WRITE);//abrimos  el archivo
myFile.print("1");
myFile.print(',');
myFile.print(valor);
myFile.print(',');
myFile.print(now.year());
myFile.print(',');
myFile.print(now.month());
myFile.print(',');
myFile.print(now.day());
myFile.print(',');
myFile.print(horaIni);
myFile.print(',');
myFile.print(minutoIni);
myFile.print(',');
myFile.print(now.hour());
myFile.print(',');
myFile.print(now.minute());
myFile.print(',');
myFile.print(horaExpo);
myFile.print(',');
myFile.print(minutoExpo);
myFile.println("");
myFile.close(); //cerramos el archivo
digitalWrite(5, HIGH);
delay(1000);
digitalWrite(5, LOW);
lcd.setCursor(0, 0);
lcd.print("Bandera ");
lcd.setCursor(0, 1);
lcd.print("olvido  ");
delay(3000);
banderaIni = 0;
```

```
        valor=0;
        for (int i=1; i<11; i++)
        {
                lcd.setCursor(0, 0);
                lcd.print("Fuera de");
                lcd.setCursor(0, 1);
                lcd.print("horario ");
                delay(3000);
                lcd.setCursor(0, 0);
                lcd.print("Apagar  ");
                lcd.setCursor(0, 1);
                lcd.print("equipo  ");
                delay(3000);
        }
}
////////////////////////////////////////////////////////////////////////////
void noTrabajo()
{
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  lcd.setCursor(0, 0);
  lcd.print("Error:  ");
  lcd.setCursor(0, 1);
  lcd.print("        ");
  delay(2000);
  lcd.setCursor(0, 0);
  lcd.print("No hay  ");
  lcd.setCursor(0, 1);
  lcd.print("Trabajo ");
  delay(2500);
}
////////////////////////////////////////////////////////////////////////////
```

```
void trabajo()
{
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  lcd.setCursor(0, 0);
  lcd.print("Error:  ");
  lcd.setCursor(0, 1);
  lcd.print("        ");
  lcd.setCursor(0, 0);
  delay(2000);
  lcd.print("Trabajo ");
  lcd.setCursor(0, 1);
  lcd.print("en curso");
  delay(2000);
}
////////////////////////////////////////////////////////////////////////////
void apagar()
{
          lcd.setCursor(0, 0);
          lcd.print("Fuera de");
          lcd.setCursor(0, 1);
          lcd.print("horario ");
          delay(3000);
          lcd.setCursor(0, 0);
          lcd.print("Apagar  ");
          lcd.setCursor(0, 1);
          lcd.print("equipo  ");
          delay(3000);
  lcd.setCursor(0, 0);
  lcd.print("        ");
  lcd.setCursor(0, 1);
  lcd.print("        ");
```

```
  DateTime now = rtc.now();
  lcd.setCursor(1,0);
  lcd.print(now.hour(), DEC);
  lcd.print(":");
        int min= now.minute();
        if (min <= 9)
        {
              lcd.print(0);
              lcd.print(min);
        }
        else
   {
              lcd.print(now.minute(), DEC);
        }
        delay(3000);
}
```