PROTOTIPO DE BAJO COSTO PARA EL MONITOREO Y ALERTA TEMPRANA DE SUSTANCIAS CONTAMINANTES Y CALIDAD DEL AIRE EN EL CAMPUS DE LA UNIVERSIDAD DEL MAGDALENA

JAVIER ELÍAS TOBÓN AYUBB

JOHAN ANDRÉS DE LA HOZ NÚÑEZ

Universidad Magdalena

Facultad de Ingeniería

Programa de Ingeniería Electrónica

Santa Marta, Colombia

Año 2021

PROTOTIPO DE BAJO COSTO PARA EL MONITOREO Y ALERTA TEMPRANA DE SUSTANCIAS CONTAMINANTES Y CALIDAD DEL AIRE EN EL CAMPUS DE LA UNIVERSIDAD DEL MAGDALENA

JAVIER ELÍAS TOBÓN AYUBB
JOHAN ANDRÉS DE LA HOZ NÚÑEZ

Director :
Ph.D CARLOS ARTURO ROBLES ALGARÍN
Codirectora:
Ph.D © AURA POLO LLANOS

Línea de Investigación:
Diseño de sistemas electrónicos

Grupo de Investigación:
Magma ingeniería - Grupo de la Universidad del Magdalena en Matemática aplicada a la Ingeniería

Universidad del Magdalena
Facultad de Ingeniería
Programa de Ingeniería Electrónica
Santa Marta, Colombia
Año 2021

Nota de aceptación:

Aprobado por el Consejo de Programa en cumplimiento de los requisitos exigidos por la Universidad del Magdalena para optar al título de Ingeniero Electrónico

_____

Jurado

_____

Jurado

Santa Marta, ____ de ____ del _____

Dedicatoria

Al creador de las cosas, que nos ha llenado de fortaleza y perseverancia para continuar a pesar de los obstáculos presentados en el camino.

De igual forma dedicamos esta tesis a nuestras madres Yusmarys Núñez y Geneth Ayubb por haber sabido formarnos con buenos sentimientos, hábitos y valores lo que nos ha ayudado a salir adelante.

Y por último a nuestras familias en general por brindarnos su apoyo incondicional en todo este camino lleno de buenos y difíciles momentos.

AGRADECIMIENTOS

En primer lugar queremos expresar nuestro agradecimiento a nuestro director de proyecto, Dr. Carlo Robles, por la dedicación y el apoyo brindado en todo este proceso, por tener la capacidad de escuchar nuestras ideas y propuestas, por guiarnos y sugerirnos ideas nuevas que fueron de gran valor para el desarrollo de toda esta investigación y desarrollo.

Así mismo agradecemos a la vicerrectoría de investigación que a través de su convocatoria para apoyar el desarrollo de trabajos de grado en programas de pregrado nos brindó la oportunidad de desarrollar este prototipo cubriendo todos los materiales utilizados.

A todos nuestros compañeros del programa de Ingeniería Electrónica por su apoyo personal y humano en el transcurso de nuestra vida universitaria.

Resumen

Este proyecto de investigación se realizó dentro del grupo Magma ingeniería en la línea de investigación de diseño de sistemas electrónicos y consiste en el diseño e implementación de un prototipo de monitoreo con dispositivos de bajo costo y herramientas libres, el cual se encarga de medir el estado de sustancias contaminantes y calidad del aire dentro del campus de la Universidad Del Magdalena.

El prototipo está equipado con un sistema de alerta temprana, un sistema de almacenamiento de datos, un sistema de transmisión inalámbrica y una interfaz gráfica. Para esto, se realizó una revisión bibliográfica para determinar las variables a medir y los sensores idóneos para esta tarea, los cuales tienen una buena relación costo-beneficio. Para lo anterior, se contó como insumo inicial el reporte del sistema de vigilancia de la calidad del aire de Corpamag. Posteriormente, se diseñó un sistema embebido capaz de procesar toda la información de los sensores, almacenarla y transmitirla. De esta forma, se obtiene como resultado un prototipo de bajo costo instalado en el árbol solar desarrollado al interior del grupo de investigación y que se encuentra funcionando en las instalaciones de la Universidad del Magdalena.

Palabras clave: Sustancias contaminantes, calidad del aire, monitoreo, sensor.

Abstract

This research project was carried out within the Magma engineering group in the research line of electronic systems design and consists of the design and implementation of a monitoring prototype with low-cost devices and free tools, which is responsible for measuring the state of polluting substances and air quality within the campus of the University of Magdalena.

The prototype is equipped with an early warning system, a data storage system, a wireless transmission system and a graphical interface. For this, a bibliographic review was carried out to determine the variables to be measured and the ideal sensors for this task, which have a good cost-benefit ratio. For the above, the report of the Corpamag air quality surveillance system was counted as an initial input.

Subsequently, an embedded system was designed capable of processing all the information from the sensors, storing it and transmitting it. In this way, the result is a low-cost prototype installed in the solar tree developed within the research group and which is operating in the facilities of the University of Magdalena.

Keywords: Pollutants, air quality, monitoring, sensor.

Contenido

Lista de figuras

Pág.

Lista de tablas

Pág.

Lista de símbolos

Símbolo	Significado
Ghz	Gigahercios
Mhz	Megahercios
PPM	Partes por millón
um	micrómetro
µg/m3	microgramos sobre metro cúbico
ARM	Advanced RISC Machine o máquina RISC avanzada
SVCA	sistemas de vigilancia de calidad del aire
SoC	System on chip o Sistema de chips en español
ICA	Índice de calidad del aire
dBm	decibelios-milivatio
JS	JavaScripts
Ω	Ohm

Objetivo General

Desarrollar un prototipo para el monitoreo de sustancias contaminantes y de la calidad del aire en el campus de la Universidad del Magdalena con el fin de generar alertas tempranas.

Objetivos Específicos

- Establecer las variables o sustancias que serán medidas con el fin de determinar los sensores necesarios para la medición de las variables teniendo en cuenta la relación costo-beneficio.

- Desarrollar un circuito para la adquisición, almacenamiento y transmisión inalámbrica de los datos.

- Diseñar una interfaz de usuario para visualizar de forma sencilla y ordenada los datos transmitidos.

Introducción

En Colombia, el monitoreo, control de la calidad del aire y de los contaminantes atmosféricos ha tomado gran relevancia en los últimos años, puesto que según la OMS (Organización Mundial de la Salud), una de cada ocho muertes ocurridas a nivel mundial es ocasionada por la contaminación del aire. Además, el Departamento Nacional de Planeación estimó que, durante el año 2015, los efectos de este fenómeno se asocian a 10.527 muertes y 67,8 millones de síntomas y enfermedades. Esta situación ha ocasionado un incremento en los costos ambientales asociados con esta contaminación, pasando del 1.1% a 1.59% del Producto Interno Bruto del 2014, (IDEAM s, f.).

En el artículo Huella de carbono en Santa Marta (Colombia) realizado en el 2014, se encontró que el nivel de huella de carbono de la ciudad es de 29.95 en promedio (medido en toneladas por CO2), caso que determina la necesidad de realizar un monitoreo y control de este tipo de agentes contaminantes. Por su parte Villamil J. y Parra J. en su trabajo de investigación evaluaron la emisión de metano y óxido nitroso de los sedimentos de manglar de la Ciénaga Grande de Santa Marta, obteniendo como resultados contenidos de metano entre nd-31569.2 µg/m-2.h-1.

Teniendo en cuenta el panorama anterior, la Corporación Autónoma Regional del Magdalena, Corpamag, puso en funcionamiento desde el 8 de marzo de 1999 el Sistema de Vigilancia de la Calidad del Aire - SVCA, ubicado en la ciudad de Santa Marta, en el cual se miden contaminantes como el PM10, PM2.5, SO2, NOx, CO, y O3. Este sistema cuenta con 16 estaciones de monitoreo, personal profesional y técnico con alta experiencia y un sistema de almacenamiento de información en el cual se han recopilado más de 23.000 registros según el informe publicado el 14 de septiembre de 2012.

La Universidad del Magdalena, consciente de las necesidades del entorno y de la importancia de tener un campus sostenible, seguro y amigable con el ambiente, como lo demuestra la política de sostenibilidad ambiental del actual plan de gobierno, realizó recientemente la medición de la huella de carbono institucional del año 2018 a través de la empresa **CO2cero**, con el fin de tomar acciones institucionales que permitan disminuir los niveles de contaminación de la comunidad académica.

En este contexto, toman relevancia los trabajos de investigación y desarrollo tecnológico como el que se aborda en el presente proyecto de grado. El monitoreo de la calidad del aire dentro del alma mater es de vital importancia debido a la presencia de estudiantes, docentes y personal administrativo, que están expuestos a diferentes agentes contaminantes. Esta situación refleja la necesidad de conocer el estado en que se encuentra el aire en el campus, con el fin de emitir alertas a partir de los límites establecidos por la resolución 2254 de 2017 del Ministerio de Ambiente y Desarrollo Sostenible, por la cual se adopta la norma de calidad del aire ambiente.

Otro factor que afecta la calidad del aire tiene que ver con la presencia de gases de efecto invernadero, como el metano (CH_4) y dióxido de carbono (CO_2) (Benavides & León 2007). La presencia de estos gases con el paso del tiempo se ha convertido en un grave problema para el planeta provocando el deterioro de la capa de ozono mientras empeora las condiciones ambientales.

Según la revista digital universitaria, estos gases son el causante del aumento del calentamiento global, por el cual se han visto efectos como el derretimiento de los glaciares, cambios en el clima, el aumento del nivel de los mares entre otros efectos que provocan el deterioro del planeta y de las condiciones de sostenibilidad para los seres humanos.

Por tanto, es de vital importancia conocer el estado de emisión de gases de efecto invernadero en la Universidad Del Magdalena, para establecer los niveles actuales y su posible incidencia en la salud y el medio ambiente.

1. Factores que afectan la calidad del aire

En este capítulo se estudió de forma general la calidad del aire y los distintos factores que afectan su estado dentro del campus universitario, comprendidos principalmente por material particulado y gases de efecto invernadero.

1.1. Principales agentes contaminantes en el aire

IDEAM (2017) afirma que “La contaminación en el aire es aquel estado en el cual, ciertas partículas o sustancias que se encuentran en el medio ambiente comienzan a sobrepasar los límites permitidos según lo cual llegaría a ser nocivo para las personas al momento de respirar”, por esto es necesario realizar SVCA para poder tener un control de aquellas sustancias o partículas mencionadas en la resolución.

Teniendo en cuenta lo anterior se definieron las sustancias a medir para el pertinente monitoreo de la calidad del aire y sustancias que pueden poner el riesgo la salud de las personas que frecuentan el campus y son descritas en la **Tabla 1**.

Tabla 1: Niveles máximos permisibles de contaminantes en el aire

Contaminante	Nivel máximo permisible (ug/m^3)	Tiempo de exposición
PM_{10}	50	Anual
	100	24 horas
$PM_{2.5}$	25	Anual
	50	24 horas
SO_2	50	24 horas
	100	1 hora
NO_2	60	Anual
	200	1 hora
O_3	100	8 horas
CO	5.000	8 horas
	35.000	1 hora

1.1.1. Partículas en suspensión

Las partículas en suspensión son todas las partículas sólidas y líquidas que se encuentran suspendidas en el aire, que según A. Fernández (1998) en su libro menciona que son importantes dado a sus efectos sobre la salud, tanto desde un punto de vista físico como químico.

1.1.1.1. Partículas PM10

Según Muñoz & Sá (2009), corresponde a un rango de volumen de las partículas, lo cual implica que la partícula tiene diámetro de 10 μm, las cuales se encuentran en suspensión dispersas en la atmósfera. Están formadas principalmente por compuestos inorgánicos como silicatos y aluminatos, metales pesados entre otros, y material orgánico asociado a partículas de carbono. Estas partículas pueden causar grandes daños para la salud provocando graves problemas de bronquitis aguda.

1.1.1.2. Partículas PM2.5

Según los autores Gaviria, Benavides y Arroyave (2008-2009), las partículas PM2.5 se caracterizan por tener un tamaño de 2.5 micrómetros o menos. Típicamente estas partículas son tan pequeñas que sólo pueden ser observadas con un microscopio electrónico. De todas las medidas de contaminación atmosférica, se cree que la PM2.5 representa la mayor amenaza para la salud. Debido a su pequeño tamaño, pueden permanecer en el ambiente largos periodos de tiempo y son absorbidas con facilidad hacia el torrente sanguíneo al ser inhaladas.

1.1.2. Dióxido de azufre (SO_2)

El dióxido de azufre es un gas que carece de color y tiene un característico olor fuerte (Sánchez, Romieu, Ruiz, Pino, & Gutiérrez, 1999). Esta sustancia tiene un gran efecto en los menores provocando que lleguen a sufrir graves daños en los pulmones.

1.1.3. Dióxido de nitrógeno (NO_2)

El dióxido de nitrógeno es un compuesto químico formado por los elementos nitrógeno y oxígeno con un color marrón-amarillento. Se ocasiona por los procesos de combustión a altas temperaturas, como en los vehículos motorizados y las plantas eléctricas (Motores de combustión). Por ello, es un contaminante frecuente en zonas urbanas. En el 2013 Aguilera citado por Gascon, M., & Sunyer, J. (2015)., en un estudio que incluía más de 2.000 participantes provenientes de cuatro ciudades españolas, observaron que el riesgo de padecer infecciones de las vías respiratorias inferiores y otitis a la edad de un año era superior en los niños cuáles madres han sido expuestas a mayores concentraciones de contaminantes provenientes del tráfico NO2 durante su embarazo. En el 2014 la OMS

anunció que este gas es uno de los responsables de aproximadamente 3.7 de millones de muertes prematuras alrededor del mundo.

1.1.4. Ozono (O_3)

La molécula de ozono está formada por 3 átomos de oxígeno y se considera un gas pálido que hace parte de la atmósfera (Sergio Cabrera, Eduardo Lissi, Juan Honeyman. 2005). Según la revista de la facultad de ciencias médicas en la Universidad de Cuenca, las principales fuentes de este contaminante de emisiones son los automóviles, mediante reacciones fotoquímicas de óxidos de nitrógeno provocando como efectos la inflamación de las vías nasales, disminución en la función respiratoria, bronquitis, crisis de asma, irritación ocular y sequedad en la garganta

1.1.5. Monóxido de carbono (CO)

El monóxido de carbono se considera como un gas incoloro e inodoro que se presenta por la combustión incompleta de material orgánico, ante poca presencia de oxígeno (Téllez, Rodríguez, Fajardo 2006). Este gas es considerado uno de los mayores contaminantes de la atmósfera terrestre y uno de los mayores problemas ambientales de América Latina. Las principales fuentes productoras de este contaminante son los vehículos automotores que funcionan con gasolina o diésel, los procesos industriales, los incendios forestales y urbanos y la incineración de materia orgánica. Los vehículos automotores y los procesos industriales son responsables de aproximadamente el 80% de las emisiones de monóxido de carbono a la atmósfera. Esta sustancia además es nociva para la salud, puesto que para 1999 la Asociación Americana de Centros para el Control de Intoxicaciones y el Sistema de Vigilancia de Exposición a Tóxicos, informaron sobre 17.006 casos de intoxicación y 35 muertes relacionadas con exposición a monóxido de carbono.

1.2. Otras sustancias contaminantes

En el aire hay presencia de otras sustancias contaminantes que no son tenidas en cuenta a la hora de hablar de calidad del aire, pero que pueden tener efectos adversos sobre la salud de los seres humanos como las mostradas a continuación.

1.2.1. Metano (CH_4)

Juan Carlos Luque (2016) afirma que el metano es un fuerte GEI y juega un papel importante en la determinación de la capacidad de oxidación de la troposfera. La fuente

más importante de metano es la descomposición de materia orgánica en sistemas biológicos.

1.2.2. Dióxido de carbono (CO_2)

Benavides y León (2007) afirman que el dióxido de carbono es uno de los gases más comunes e importantes en el sistema atmósfera-océano-Tierra, que dentro de los gases GEI es el más importante asociado a actividades humanas y el segundo gas más importante en el calentamiento global después del vapor de agua. Este gas tiene fuentes antropogénicas y naturales.

1.3. Variables atmosféricas

También se tiene en cuenta algunas variables atmosféricas que si bien no son factores contaminantes son magnitudes de consideración, debido a que su variación influye en la medición de los gases que se encuentran en el ambiente.

1.3.1. Temperatura

En 1989, Kane, J., & Sternheim, M. exponen en su libro que: “La temperatura de un objeto está relacionada con la energía cinética media de los átomos y las moléculas que componen dicho objeto”.

1.3.2. Humedad relativa (HR)

Stephen R. Gliessman (2002) define en su libro a la humedad relativa como la relación entre la presión parcial del vapor de agua y la presión de vapor de equilibrio del agua a una temperatura dada. La humedad relativa depende de la temperatura y la presión del sistema de interés.

1.4. Índice de calidad del aire (ICA)

Por su parte Hernández A. M. (2013) refiere que el Índice de calidad del aire (ICA) permite comparar los niveles de contaminación del aire de las estaciones de monitoreo que conforman un Sistema de Vigilancia de Calidad del Aire, en un tiempo t, que corresponde al período de exposición previsto en la norma para cada uno de los contaminantes que se está midiendo.

Así mismo, los niveles de contaminación del aire reflejados por cada uno de los rangos del ICA determinan un conjunto de acciones preventivas que se recomienda sean tenidas en cuenta por la población.

2. Herramientas de hardware

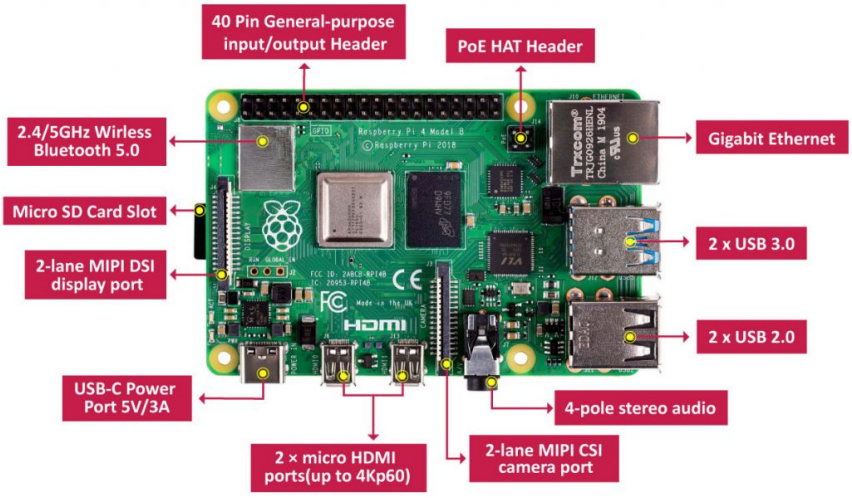
Para el desarrollo del prototipo se usaron algunas herramientas de hardware útiles y teniendo como prioridad su bajo costo, que ayudaron al desarrollo de este, donde destacan tarjetas de desarrollo, sensores, conversores de voltaje y tarjetas de comunicación inalámbrica.

2.1. Tarjeta de desarrollo

Una tarjeta de desarrollo es una placa o circuito que contiene un microcontrolador principal que corre o ejecuta una serie de instrucciones de un programa suministrado. Alrededor de este procesador o unidad principal se ha creado un diseño electrónico que permite: la programación del procesador, suministra el voltaje adecuado para el correcto funcionamiento del controlador y proporciona acceso a las entradas y salidas del microcontrolador para la conexión de sensores y actuadores.

2.1.1. Raspberry Pi 4 B

Figura 1: Raspberry Pi 4 B



Doole, G (2020). Raspberry Pi 4 Pinout, Features and Peripherals [Figura]. Recuperado de: https://microcontrollerslab.com

La Raspberry Pi es una tarjeta de desarrollo aproximadamente del tamaño de una tarjeta de crédito que cuenta con una GPU, una ranura memoria microSD para instalar el sistema operativo, conector USB para la alimentación y por último posee puertos GPIO para conectar sensores y actuadores.

Su modelo Pi 4 B destaca porque hace uso de un nuevo procesador con núcleos ARM Cortex-A72 a 1,5 GHz. El SoC Broadcom BCM2711B0 utilizado es un salto radical por estar fabricado con litografía de 28 nm en lugar de los 40 nm de los Broadcom anteriores.

Se hace uso de esta tarjeta debido a que cuenta con mayor robustez, cantidad de entradas I^2C y digitales acordes a lo requerido y gran potencia de procesamiento. Adicionalmente a esto, su costo no es elevado y está diseñada para trabajos en exteriores, con una única desventaja para este caso como lo es la ausencia de entradas analógicas, solucionado con la inclusión de un conversor analógico digital.

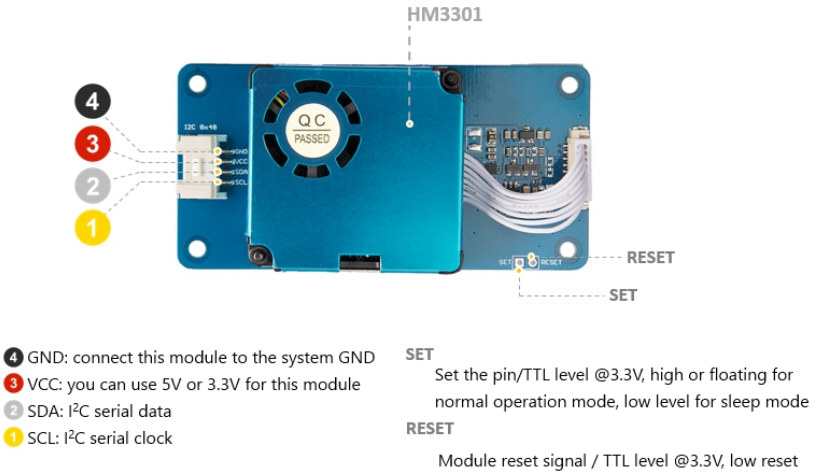
En la constitución del prototipo se encargará de recibir todos los datos procedentes de sensores, proporcionar el índice de calidad del aire, el sistema de alertas, cargar la información en una base de datos local y posteriormente enviarla a el servidor remoto, a través de comunicación inalámbrica.

2.2. Sensores digitales

Entre los dispositivos seleccionados se encuentran sensores con salidas digitales para la adquisición de los datos de diferentes variables físicas del ambiente ya sea mediante bus de serie de datos o Single-Bus.

2.2.1. Sensor láser HM3301

Figura 2: Sensor HM3301

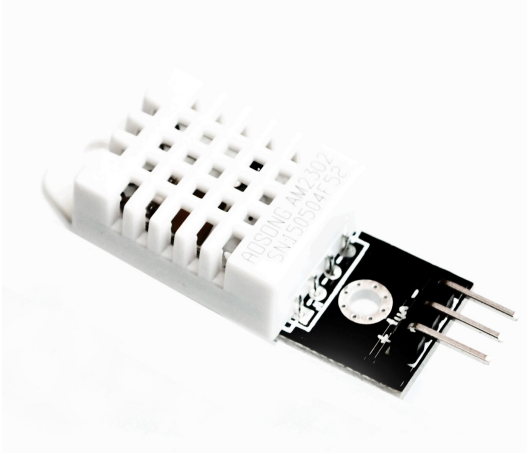


Seeed wiki (Sin fecha). Grove - Laser PM2.5 Sensor (HM3301) [Figura]. Recuperado de: https://wiki.seeedstudio.com

El sensor HM3301 es un sensor de polvo que puede detectar partículas en tiempo real desde $0.3um$. Gracias a su tecnología de dispersión láser, cuenta con 4 conexiones, 2 de las cuales son VCC y GND para su alimentación de 3.3V o 5V y 2 para su comunicación a través de un bus de datos I^2C que son SDA y SCL para capturar los datos mediante un microcontrolador como se observa en la **Figura 2**, con 3 posibles canales de detección de partículas a $1um$, $2.5um$ y $10um$.

2.2.2. Sensor DHT22

Figura 3: Sensor DHT22



Didácticas Electrónicas (Sin fecha). Sensor temp. y hum. AM2302/DHT22 con cable [Figura]. Recuperado de: https://www.didacticaselectronicas.com

Este sensor permite medir la temperatura y humedad relativa del ambiente al que se exponga, de una forma simple y precisa a un muy bajo costo, destaca por tener una salida digital en forma de bus con el que obtienen los datos de una manera fácil y rápida y su estructura física está compuesta por 3 pines (VCC, DATA Y GND), como se observa la **Figura 3** , los cuales son para alimentación de 3.3V - 5V. el cable por donde obtener los datos y la respectiva señal a tierra.

2.2.3. Sensor Air Quality 5 Click

Figura 4: Sensor Air Quality 5 Click

Didácticas Electrónicas (Sin fecha). Air Quality 5 Click [Figura]. Recuperado de: https://www.didacticaselectronicas.com

El Air Quality 5 Click es una configuración electrónica que cuenta con el sensor MICS-6814 el cual es un sensor MOS triple que es sensible a varios gases como lo son el NO_2, CO y CH_4, conectado a un ADC de 12 bits a través de un interfaz I^2C, este dispositivo cuenta con 6 pines de conexión (3.3V, GND, RDY, SCL, SDA y 5V) para su alimentación y recepción de datos que son necesarios para su correcto funcionamiento y RDY como una salida que avisa cuando el dispositivo está listo para entregar datos correctamente.

2.3. Sensores Analógicos

Para la lectura de ciertas sustancias se optó por algunos sensores analógicos debido a que algunos son de bajo costo y más fáciles de usar que sus equivalentes en digital, como desventaja se tiene que la tarjeta de desarrollo seleccionada no cuenta con este tipo de entradas por lo que se hace uso de un conversor análogo/digital para solventar este inconveniente.

Con relación a las sustancias a medir se encontró que la familia de sensores MQ cubre una amplia gama de estas, por lo que son vitales para esta implementación.

Estos sensores son electroquímicos y varían su resistencia cuando se exponen a determinados gases, internamente poseen un calentador encargado de aumentar la temperatura interna y con esto el sensor pueda reaccionar con los gases provocando un cambio en el valor de la resistencia. El calentador dependiendo del modelo puede necesitar un voltaje entre 5 y 2 voltios, el sensor se comporta como una resistencia y

necesita una resistencia de carga (RL) para cerrar el circuito y con este hacer un divisor de tensión y poder leerlo desde un microcontrolador.

2.3.1. Sensor MG-811

Figura 5: Sensor MG-811

DFROBOT (Sin fecha). Gravity: Analog CO2 Gas Sensor For Arduino (MG-811 Sensor) [Figura]. Recuperado de: https://www.dfrobot.com

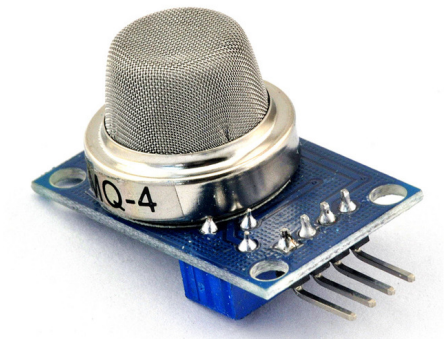
Este sensor análogo fabricado por Wisen es seleccionado debido a su gran sensibilidad a las emisiones de CO_2 este sensor cuenta con la ventaja de poseer un conector tipo gravity de alta calidad que facilita su uso, tiene un voltaje de funcionamiento de 5V, salida a tierra y su salida análoga, todo integrado en este conector, dentro de sus características principales tenemos:

- Una salida digital
- Conector de alta calidad
- Superficie de oro de inmersión
- Circuito de calefacción a bordo
- Circuito de acondicionamiento integrado para amplificar la señal de salida

Adicionalmente cuenta con una salida digital ajustable para encender un led al detectar cierta cantidad de CO_2 en el ambiente, para su correcto funcionamiento el fabricante recomienda una resistencia de carga menor de 35Ω y mayor de 3Ω.

2.3.2. Sensor MQ-4

Figura 6: Sensor MQ-4



SERVOTRONIK (Sin fecha). Sensor MQ-4 metano y gas natural [Figura]. Recuperado de: https://www.servotronik.com.co

Este sensor de la familia MQ es el seleccionado para la detección de metano el cual al igual que la mayoría de sensores de esta familia cuenta con 4 pines, alimentación a 5V, señal a tierra, salida de datos analógica y una salida especial digital la cual enciende un led al detectar cierta cantidad de metano ajustable con una resistencia variable. Dentro de sus características destacan:

- Amplio alcance
- Alta sensibilidad
- Respuesta eficiente
- Vida larga y útil
- Fácil de instalar

Adicionalmente, el fabricante recomienda una resistencia de carga (RL) de 20kΩ para una concentración de 1000 ppm de $\boldsymbol{CH_4}$ en aire limpio.

2.3.3. Sensor MQ-131

Figura 7: Sensor MQ-131

Didácticas Electrónicas (Sin fecha). Sensor de Ozono MQ131 - V2 [Figura]. Recuperado de: https://www.didacticaselectronicas.com

Este sensor fue seleccionado gracias a su alta sensibilidad al ozono, cuenta con 4 pines de conexión, alimentación a 5V, señal a tierra, salida de datos analógica y una salida especial digital la cual enciende un led al detectar cierta cantidad de O_3, ajustable con una resistencia variable. Dentro de sus características destacan:

- Circuito de accionamiento simple Aplicación
- Larga vida útil
- Respuesta eficiente
- Fácil de instalar

El fabricante recomienda una resistencia de carga (RL) de 50kΩ para una correcta detección del gas.

2.3.4. Sensor MQ-136

Figura 8: Sensor MQ-136

Didácticas Electrónicas (Sin fecha).Sensor Gas Hidrógeno MQ136 [Figura]. Recuperado de: https://www.didacticaselectronicas.com

Este sensor es el encargado de la medición de SO_2 dentro de la familia MQ, es similar a los demás integrantes y su forma de manejo es similar, físicamente también está constituido de 4 pines que sirven para alimentación, señal a tierra, salida analógica y la salida digital de detección de sustancia que sirve como alerta a través de su led.

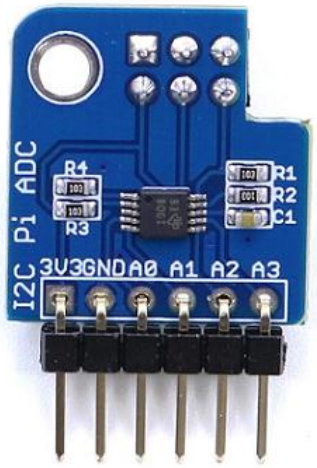
Sus características principales son las siguientes:

- Amplio alcance
- Alta sensibilidad
- Respuesta eficiente
- Vida larga y útil
- Fácil de instalar

El fabricante recomienda una resistencia de carga (RL) de entre 30KΩ-200KΩ para su correcto funcionamiento.

2.4. Conversor Análogo/Digital ADS 1115

Figura 9: Analogo/Digital ADS1115

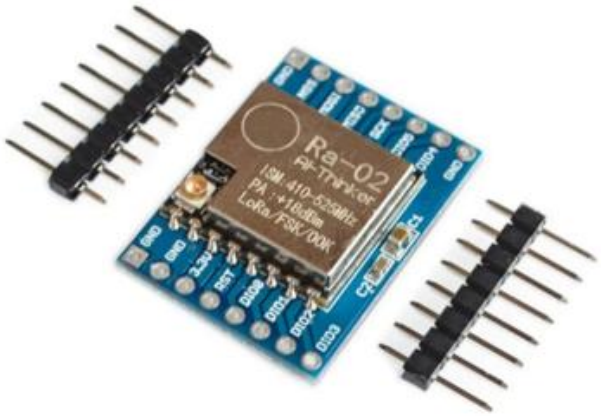


Didácticas Electrónicas (Sin fecha). ADC Raspberry, ADS1115 [Figura]. Recuperado de: https://www.didacticaselectronicas.com

Este dispositivo es un módulo para convertir voltajes analógicos a digitales con una precisión de 16 bits a través de una comunicación I^2C contando con un amplificador de ganancia programable de hasta 16x, cuenta con 5 pines de conexión (5V, GND, 3.3V, SDA y SCL) para su alimentación y obtener los datos a través de un microcontrolador, datos provenientes de las entradas del dispositivo que se pueden configurar como 4 canales de entradas de un solo extremo o dos canales diferenciales.

2.5. Tarjeta Wireless LORA-02

Figura 10: Tarjeta Wireless LORA-02

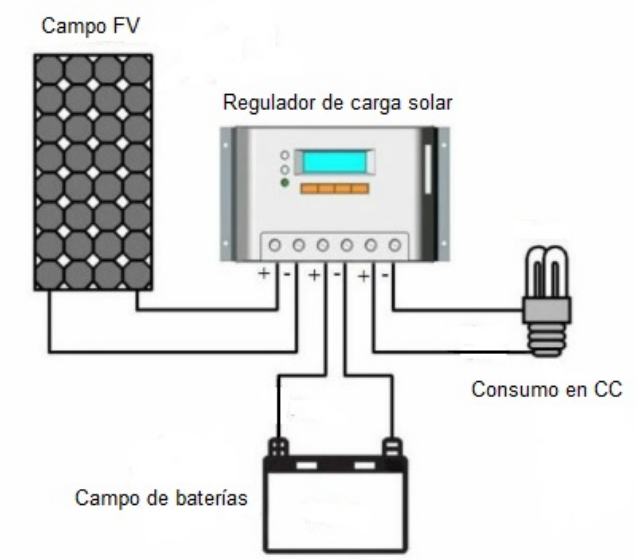


Didácticas Electrónicas (Sin fecha). Tarjeta Wireless LORA-02 433MHz [Figura]. Recuperado de: https://www.didacticaselectronicas.com

La tarjeta wireless LORA-02 es un módulo de inalámbrico LoRA de 433 MHz caracterizado por su bajo consumo eléctrico y bajo costo de adquisición, alimentado de 1.8V a 3.7V (normalmente 3.3V) con corrientes menores o igual a 10.8mA en su recepción y 120mA en la transmisión con más de 20 dBm, mientras que en modo reposo tan solo 0.2uA, logrando comunicaciones punto a punto de hasta 10 Km con una conexión para antenas externas, todo esto a través de una conexión SPI GPIO Half-duplex a un microcontrolador.

2.6. Sistema de alimentación fotovoltaica

Figura 11: Sistema de alimentación fotovoltaico



Monsolar (2021). ¿Qué es y qué hace un regulador de carga solar? [Figura]. Recuperado de: https://www.monsolar.com

Para la realización de este proyecto se pretende diseñar una instalación fotovoltaica debido a que es amigable con el medio ambiente y a su vez, no genera gastos en electricidad a la universidad. Esta estación se encargará de suministrar la energía necesaria para el correcto funcionamiento de todos los dispositivos y consta de paneles fotovoltaicos, batería y controlador de carga.

2.6.1. Panel fotovoltaico

En 2019 Villalta E, señala que “Los paneles o módulos fotovoltaicos, están formados por conjuntos de células fotovoltaicas que producen electricidad a partir de la luz que incide sobre ellos mediante el efecto fotoeléctrico” (p.10)

De acuerdo con lo anterior, en esta implementación serán los encargados de recolectar la energía solar en el lugar de instalación para ser almacenada en la batería la cual

finalmente será la encargada de alimentar todos los dispositivos utilizados para el monitoreo de calidad del aire.

2.6.2. Batería

El portal Energía solar expresa que “Las baterías solares tienen el objetivo de acumular la energía eléctrica generada por las placas solares fotovoltaicas para poder utilizarla durante la noche o en días nublados”(Planas O, 2015).

Una batería consta de pequeños acumuladores eléctricos de 2V integrados en el mismo elemento. Las baterías suministran corriente continua a 6, 12, 24 o 48V. El acumulador es la celda que almacena energía a través de un proceso electroquímico.

Esta será la encargada de suministrar la energía necesaria por las noches y en los días nublados en los cuales la energía solar sea escasa e insuficiente para que los paneles solares entreguen la corriente necesaria para todo el sistema.

2.6.2. Controlador de carga

Bordon M (2010), expresa en su artículo “El Regulador de carga es el dispositivo encargado de proteger a la batería frente a sobrecargas y sobredescargas”.

Este dispositivo se encarga de maximizar el rendimiento del sistema de alimentación fotovoltaico, así mismo proteger y aumentar la vida útil de cada uno de los componentes para que esta implementación sea sustentable en el máximo tiempo posible.

En pocas palabras, este dispositivo será el encargado de transferir la corriente proveniente de los paneles solares para cargar la batería y a su vez regular la salida hacia el sistema para garantizar estabilidad en todo el proceso.

3. Herramientas de software

En este capítulo se estudiarán las herramientas de software seleccionadas para la realización del prototipo, ya que son indispensables para cualquier desarrollo. Dentro del software seleccionado destacan los sistemas operativos con los que trabajan las tarjetas de desarrollo, los lenguajes de programación usados, los lenguajes para el manejo de la base de datos y un software especializado para la creación de PCB.

Cabe destacar que la mayoría de estas herramientas cuentan con versiones de uso libre y gratuito lo cual favorece al costo de desarrollo del prototipo.

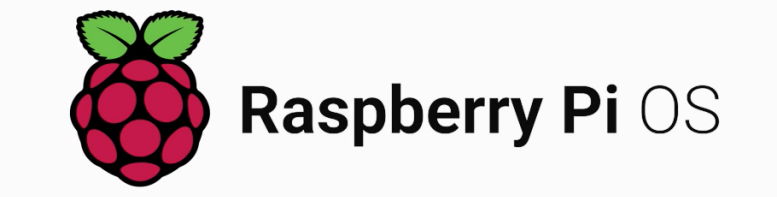
3.1. Sistemas operativos

Para las tarjetas de desarrollo es necesario un software que administre los recursos de las tarjetas de desarrollo, este es el papel que cumple el sistema operativo, el software sobre el cual se van a correr los programas necesarios en diferentes lenguajes de programación para esta aplicación.

En esta ocasión en cada una de las tarjetas de desarrollo se ha optado por una distribución de software basadas en el núcleo de Linux de código abierto.

3.1.1. Raspberry Pi OS with desktop

Figura 12: Raspberry Pi OS



capterra (2021). ¿Qué es Raspberry Pi OS? [Figura]. Recuperado de: https://www.capterra.co

Raspberry Pi OS es un sistema operativo GNU/Linux anteriormente llamado Raspbian basado en Debian el cual es un software de código libre, es el recomendado por el fabricante ya que viene optimizado para el hardware de las Raspberry Pi y viene con más de 35.000 paquetes de software precompilado en un formato fácil para su instalación.

3.2. Lenguajes de programación

Los lenguajes seleccionados son los encargados de indicar las instrucciones paso a paso al hardware para un correcto funcionamiento del sistema totalmente integrado, cada uno de ellos son aplicados para una función específica, ya sea para el interfaz máquina a usuario o la recolección o almacenamiento de datos.

En este desarrollo los lenguajes de programación son de vital importancia tanto para la lectura e interpretación de datos como para su posterior visualización e interacción.

3.2.1 Python

Python es un lenguaje de programación de alto nivel interpretado y de código abierto orientado a objetos, el cual por su sencilla sintaxis se hace muy fácil de leer y entender. Dentro de sus características destaca una amplia biblioteca de herramientas (librerías) y su versatilidad al ser multiplataforma, lo que lo hace uno de los lenguajes más utilizados en todo el mundo.

En este caso se usa como lenguaje base para el manejo de sensores y datos con la Raspberry Pi.

3.2.2 HTML

HTML es un lenguaje de marcado que por sus siglas HyperText Markup Language o sea lenguaje de marcas de texto, creado para el diseño de páginas web ejecutado en la parte FrontEnd, caracterizado por su fácil estructura de programación y amplia aceptación de contenidos como texto, imágenes o videos.

Particularmente este lenguaje se usa dentro del desarrollo del proyecto como la base para la interfaz de usuario donde se visualiza e interactúa con los datos obtenidos con el prototipo.

3.2.3 JavaScript

JavaScript es un lenguaje de programación interpretado diseñado para aplicaciones FrontEnd, permitiendo implementar funciones complejas en páginas web, como contenido dinámico, animaciones, mantener comunicación secuencial con el servidor, etc.

JS como es llamado este lenguaje se encarga de darle el dinamismo a la interfaz web para que el usuario final pueda interactuar con los datos obtenidos de una manera fácil y dinámica.

3.2.4 PHP

PHP es un lenguaje de código abierto muy popular, adecuado para desarrollo web y que puede ser incrustado en HTML, se utiliza para generar páginas web dinámicas donde su contenido es variable. Este lenguaje es ejecutado del lado del servidor y se accede por medio del HTML.

Con la implementación de PHP en el interfaz web, se busca dinamismo en su contenido, debido a la gran cantidad de datos almacenados en el servidor por medio del código fuente de la página.

3.2.5 CSS

CSS es un lenguaje de diseño gráfico que por la traducción de sus ciclas quiere decir Hojas de estilo en cascada, utilizado para darle forma y estilos amigables para los usuarios a las páginas web escritas en HTML o XHTML el cual es escrito en un lenguaje de marcado. Con ***CSS*** se pueden otorgar caracteres como colores, tamaños o formas de acuerdo con el tamaño de la pantalla en la cual se presenta la página web para diferentes contenidos o grupos de ellos mismos y así ser más agradable visualmente para el usuario.

Dentro de este desarrollo su uso está destinado hacia los estilos de la interfaz web, haciendo que sean más amigable visualmente para el usuario que acceda en busca de los datos obtenidos.

3.3. Servidor LAMP

LAMP no es más que un acrónimo usado para describir un servidor con diferentes tecnologías, las cuales son Linux (el sistema operativo sobre el cual corre el servidor), Apache (el servidor en sí encargado de dar respuesta al usuario a sus peticiones a través del navegador), ***MySQL*** (el gestor de la base de datos integrada al servidor) y en este caso PHP el lenguaje de programación que se encarga de integrar todas estas partes y lograr darle solución a un servidor totalmente integrado y completo.

Este servidor se encarga de dar respuestas a los usuarios ante sus peticiones de datos de forma rápida y dinámica con apoyo de otros servicios.

3.3.1. Apache2

Apache es un servidor web HTTP gratuito de código abierto muy usado y disponible para múltiples sistemas operativos, contiene grandes ventajas ya que puede ser modular extensible y fácil de conseguir, puede ser usado para enviar páginas web estáticas y dinámicas.

En este caso ***Apache2*** se encargará de interpretar todas las peticiones dados por el usuario de forma automática, respondiendo de forma óptima.

3.3.2. MySQL

MySQL es un sistema de gestión de bases de datos relacionales de código abierto, cuenta con un modelo cliente-servidor utilizado para crear y administrar bases de datos basadas en un modelo relacional.

Su uso está basado en que uno o más dispositivos (clientes) se conectan a un servidor a través de la red. Cada cliente puede realizar una solicitud desde la interfaz gráfica de usuario en sus pantallas para acceder a la base de datos, y el servidor responderá a esa solicitud de manera acertada entregando el dato requerido.

MySQL es la herramienta seleccionada dentro del proyecto para la creación y administración de la base de datos, donde se almacenan todos los datos de las lecturas obtenidas por el sensor.

3.3.3.PHPMyAdmin

PHPMyAdmin es una herramienta utilizada para administrar bases de datos ***MySQL*** con una interfaz gráfica por medio de una página web utilizando un navegador web. Esta herramienta tiene como base un lenguaje de programación el cual es PHP y así permite de una manera más intuitiva crear, eliminar, editar bases de datos o ejecutar cualquier comando SQL a través de ***PHPMyAdmin***.

Con la ayuda de ***PHPMyAdmin*** en este desarrollo, se puede tener acceso y control a la estructura e información de la base de datos del servidor web.

3.4 Eagle

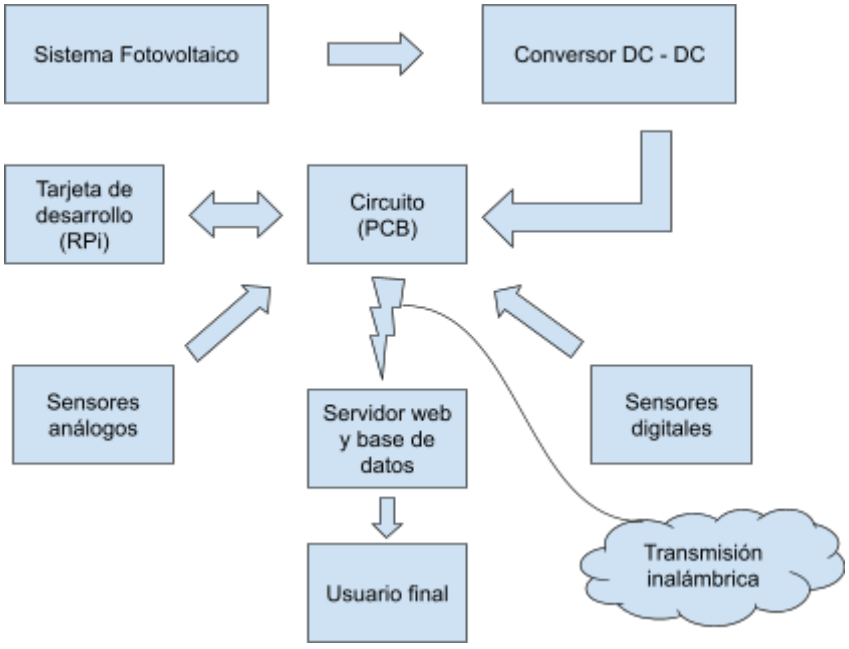
Eagle es un programa de diseño de diagramas y PCBs con auto enrutador creado por Autodesk, famoso alrededor del mundo de los proyectos electrónicos debido a la gran variedad de herramientas y librerías de componentes, también destaca por su función para crear tus propios componentes a la medida y poder hacer tus placas personalizadas.

Con este software se diseñó la PCB con todas las conexiones para todos los dispositivos involucrados en el prototipo.

4. Diseño del prototipo

Es necesario que como resultado final un prototipo funcional para evidenciar de manera física los resultados de este proyecto y por esto se explica detalladamente cómo se usaron y se diseñaron todos sus componentes, cabe resaltar que en la **Figura 13**, se observa un esquema general del proyecto.

Figura 13: Diagrama general



4.1. Estructura física

La estructura física de este proyecto tendrá como base la estructura desarrollada para el árbol solar ubicado en el campus de la Universidad del Magdalena, en la cual se aprovechará su diseño para adherir la caja con protección IP seleccionada para resguardar todos los dispositivos de los daños causados por el ambiente.

En otras palabras, esta implementación usará la estructura del árbol solar (Paneles Y batería) como fuente de alimentación añadiendo a esta una caja ip la cual contendrá los sensores y dispositivos usados para el correcto funcionamiento de la estación de medición de calidad del aire.

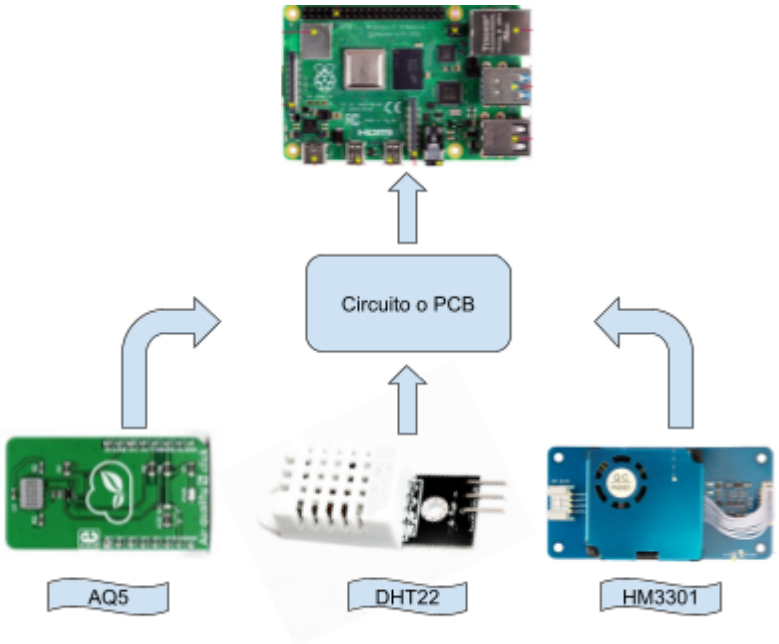
4.2. Uso de sensores

En este apartado se pone en evidencia todo lo relacionado con el uso de los sensores utilizados para la medición de sustancias contaminantes.

4.2.1. Uso de Sensores digitales

Al contar con diferentes tipos de sensores digitales se tienen diferentes tipos de uso y de programación para los sensores análogos, en este caso destacan dos tipos que usan tecnologías como lo son el protocolo de comunicación I^2C y Single-Bus, los cuales se conectar directamente a la tarjeta de desarrollo Raspberry Pi a través del circuito PCB, como se observa en la **Figura 14.**

Figura 14: Sensores digitales



4.2.1.1. Uso del sensor láser HM3301

El sensor láser HM3301 fue el utilizado para tomar lecturas de $PM_{2.5}$ y PM_{10} del medio ambiente. Para recoger los datos medidos por este sensor se utilizó la librería en Python de Grove ***grove_PM2_5_HM3301***, con esta librería obtenemos tanto la concentración estándar como la atmosférica de $PM_{1.0}$, $PM_{2.5}$ y PM_{10} en ug/m^3 a través del protocolo de comunicación I^2C, que en este caso se necesita la concentración atmosférica de $PM_{2.5}$ y PM_{10} en ug/m^3.

Se están utilizando otros componentes electrónicos en el bus I^2C que funcionan a la frecuencia de comunicación habitual la cual es 100 kHz y este sensor exige configurar

bus de comunicación a 20 kHz para la correcta comunicación con el sensor. Teniendo en cuenta que el número de dispositivos a 100 kHz es mayor, no es posible disminuir la frecuencia de funcionamiento para evitar errores de comunicación, por este motivo se opta por implementar la librería ***pigpen*** para hacer lo que se llama bit-banging en los puertos GPIO de la Raspberry Pi para simular una interfaz I^2C de 20 KHz.

Predko, M (2001) expone en su libro que; “El bit-banging es una herramienta de software utilizada para reemplazar los hardware dedicados a generar señales de transmisión o procesar señales recibidas con un golpe de bit” esto con el fin de lograr simular protocolos como es el I^2C el cual es implementado en esta ocasión.

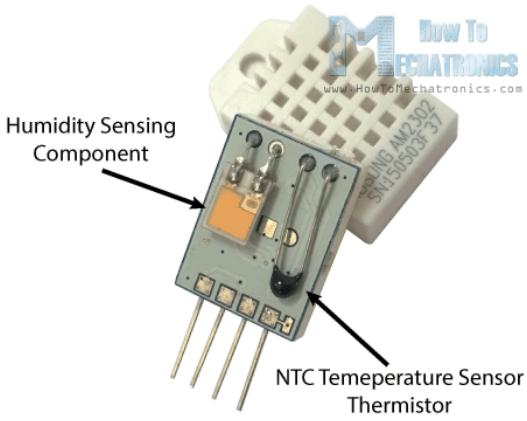
Una vez implementado el anterior sistema se logran obtener las lecturas del sensor mediante el protocolo de comunicación I^2C; cabe destacar que este sensor no necesita calibración ya que el método de lectura de las partículas que implementa es láser, teniendo en cuenta el flujo de aire que el mismo extrae del ambiente.

Para la programación de este sensor se inicia creando el bus I^2C con la función ***pigpio.pi()***, posteriormente se hace uso de la función ***SDL_Pi_HM3301.SDL_Pi_HM3301()*** para la asignación de los pines físicos del bus I^2C, por último para obtener los valores de $PM_{2.5}$ y PM_{10} se ejecuta la función ***hm3301.get_data().***

4.2.1.2. Uso del sensor DHT22

El sensor DHT22 usado para tomar lecturas de temperatura y humedad relativa del medio ambiente, cuenta con un termistor y un componente sensor de humedad.

Figura 15: Componentes DHT22



howtomechatronics (2016). DHT11 & DHT22 Sensors Temperature and Humidity Tutorial using Arduino [Figura]. Recuperado de: https://howtomechatronics.com

Para el uso de este sensor, se hace uso de una librería en python lista para usar de Adafruit la cual es ***adafruit_dht*** para obtener correctamente las lecturas de ambos constantes físicas y no tener problemas en los GPIOs.

Con estas librerías solo bastaría seleccionar el sensor que en este caso es el DHT22, conectar el sensor a los pines correspondientes de acuerdo con lo asignado en la programación con la función ***adafruit_dht.DHT22()*** y tomar las lecturas tanto de temperatura en grados Celsius y humedad relativa en %.

Cabe resaltar que este sensor no requiere una calibración debido a que viene listo para usar de fábrica.

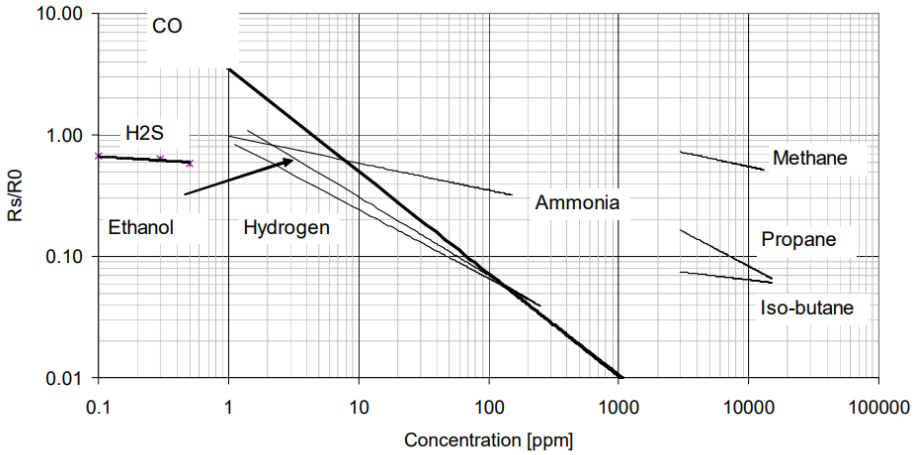
4.2.1.3. Uso del sensor Air Quality 5 Click

El sensor Air Quality 5 Click viene pre-configurado con las resistencias necesarias para la calibración del triple sensor MOS MICs-6814 el cual para su lectura tiene salidas análogas por eso este viene con un conversor análogo/digital integrado.

Este sensor será usado para tomar las lecturas de las variables físicas de ***CO*** y ***NO_2*** el cual serán lecturas provenientes ya del conversor análogo/digital que cuenta el sensor, el cual es el ADS1015, para eso se utiliza las librerías en python ***adafruit_ads1x15*** que ayuda a tomar las lecturas fácilmente a través del protocolo de comunicación I^2C.

Una vez recogida la información a través del protocolo de comunicación I^2C se interpreta los datos a la variable física requerida, que para el caso de ***CO*** se tiene en cuenta la gráfica entregada por el fabricante del MICs-6814 que se observa en la **Figura 16**.

Figura 16: Curva de sensibilidad CO MICs-6814



RED sensor, continuous power ON, 25°C, 50% RH

SENSORTECH (En línea). Datasheet MICS-6814 [Figura]. Recuperado de: https://www.sgxsensortech.com

Teniendo en cuenta la información anterior se obtiene la siguiente ecuación para hallar la concentración de ***CO*** en ppm:

$$CO(ppm) = 4,385 * (\frac{Rco}{R0co})^{-1,179}$$

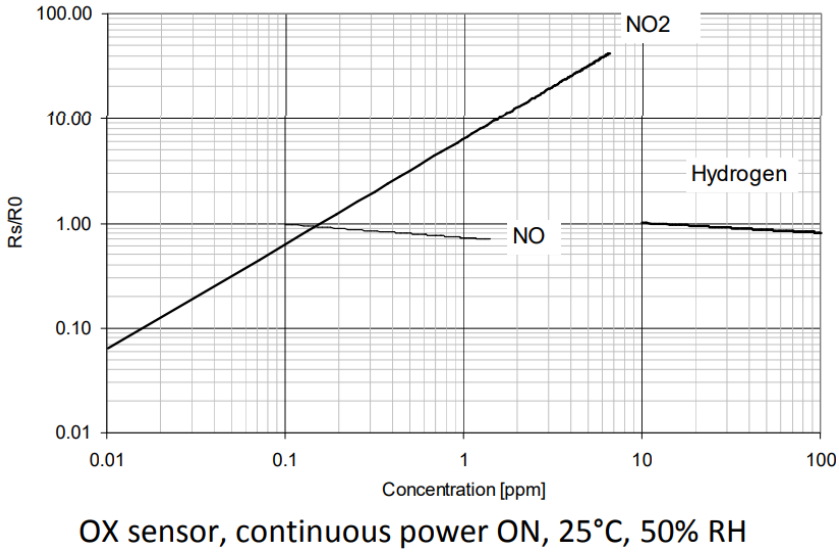
Donde R0co es la resistencia de calibración la cual es establecida teniendo en cuenta el sensor usado para calibración ***RAC 3/5***, Rco es la resistencia del sensor dedicada a esta sustancia que se halla mediante la siguiente ecuación:

$$Rco = \frac{1000000*Vin}{3,3 - Vin}$$

Y Vin es el voltaje medido por el ADS1015 integrado en el sensor para la medición de ***CO***.

Para obtener el valor en ppm, primero se inicia la librería del ADS1015 con la función ***ADS1.ADS1015()***. Posteriormente se utiliza la función ***AnalogIn().voltage*** pasándole los valores de entrada de ***CO***, para obtener los valores de voltaje del sensor y finalmente se hace uso de las ecuaciones mencionadas anteriormente para calcular el valor de la sustancia.

Figura 17: Curva de sensibilidad NO_2 MICs-6814



SENSORTECH (En línea). Datasheet MICS-6814 [Figura]. Recuperado de: https://www.sgxsensortech.com

Con base a la gráfica anterior se obtiene la siguiente ecuación para hallar la concentración de ***NO_2*** en ppm:

$$NO_2(ppm) = 6,855 * \left(\frac{Rno2}{18045}\right)^{1,007}$$

Donde R0no2 es la resistencia de calibración, establecida usando el equipo ***RAC 3/5*** como referencia, Rno2 es la resistencia del sensor dedicada a esta sustancia que se halla mediante la siguiente ecuación:

$$Rno2 = \frac{15000*Vin}{3,3 - Vin}$$

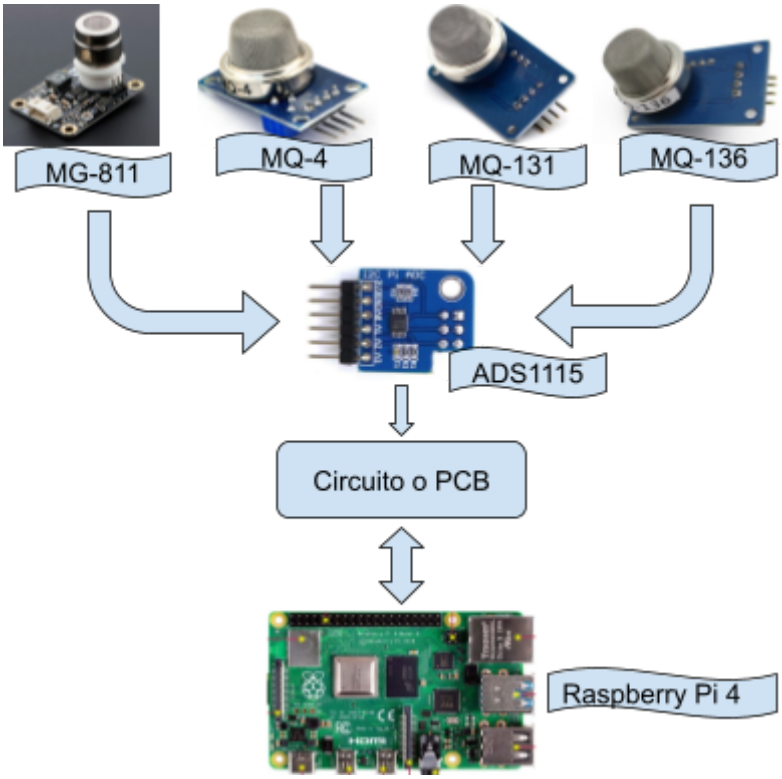
Y Vin es el voltaje medido por el ADS1015 integrado en el sensor para la medición de ***NO_2***.

Para obtener el valor en ppm primero se inicia la librería del ADS1015 con la función ***ADS1.ADS1015()***. Posteriormente se utiliza la función ***AnalogIn().voltage*** pasándole los valores de entrada de ***NO_2*** para obtener los valores de voltaje del sensor y finalmente se hace uso de las ecuaciones mencionadas anteriormente para calcular el valor de la sustancia.

4.2.2. Uso de sensores analógicos

En el caso de los sensores analógicos, todos los dispositivos usados hacen parte de una misma familia. Por lo que su uso y programación es de manera similar. Contando además con la conversión de su señal de salida a digital (I^2C) por medio de un conversor analógico-digital para su correcta lectura por la tarjeta usada tal como se observa en la **Figura 18.**

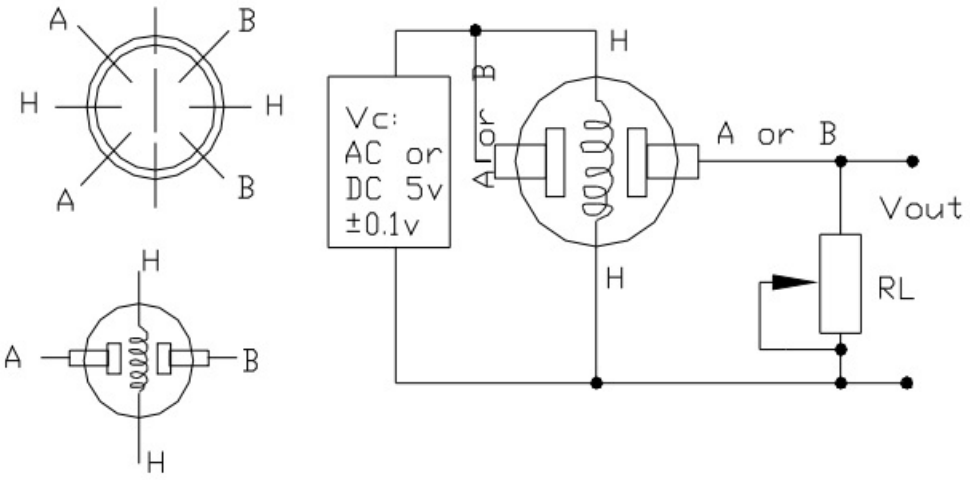
Figura 18: Sensores análogos



Estos sensores electroquímicos basan su funcionamiento en la variación de su resistencia cuando son expuestos a diferentes sustancias, por lo cual su lectura depende de ese valor y de la curva de sensibilidad de cada gas. Por último para su uso y almacenamiento se realiza la conversión del dato a partes por millón (ppm).

Para la lectura de los valores analógicos se hace uso de la librería para python ***adafruit_ads1x15*** la cual permite convertir los valores del sensor al protocolo de comunicación I^2C, para ser interpretados por la tarjeta Raspberry.

Figura 19: Esquema general de un sensor MQ

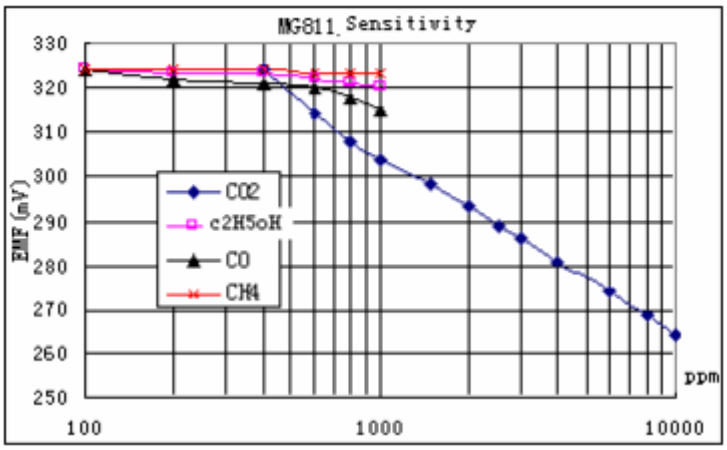


naylamp mechatronics (Sin fecha). TUTORIAL SENSORES DE GAS MQ2, MQ3, MQ7 Y MQ135 [Figura]. Recuperado de: https://naylampmechatronics.com/

4.2.2.1. Uso del sensor MG-811

Este sensor anteriormente mencionado se usa para la medición de CO_2 y cuenta con un voltaje de funcionamiento de 5V y su curva de sensibilidad es mostrada en la **Figura 20.**

Figura 20: Curva de sensibilidad MG-811



Hanwei Electronics (En línea). Datasheet MG-811 sensor [Figura]. Recuperado de: https://html.alldatasheet.es

Para su programación, se toman 3 puntos de la curva (P1,P2 y P3) mostrados en la **Tabla 2**. Para su posterior uso en la ecuación de conversión a ppm.

Así mismo se lee el valor del pin analógico del sensor y es dividido entre la ganancia del sensor equivalente a 8.5V, para tener el valor promedio del voltaje en la resistencia y por último se obtiene el valor de la concentración en ppm de la siguiente ecuación:

$$C02\ ppm\ =\ 10^{\frac{ratio-P2}{P3}-P1}$$

Donde $ratio$ es la división del valor analógico leído por el sensor y la ganancia del sensor, $P1,\ P2\ y\ P3$ puntos de la gráfica de sensibilidad de ***CO_2***.

Cabe resaltar que la ecuación es una potencia de base 10 estimada por regresión lineal, ya que la curva de sensibilidad está dada en el mismo formato.

Para tener una medición adecuada y veraz de los datos obtenidos es necesario tener un sensor anteriormente calibrado como referencia. Para este caso, se usó el sensor análogo infrarrojo de gravity SEN0219 que viene listo para usar de fabrica ya que es infrarrojo.

Tabla 2: Valores de la curva de CO_2

$P1$	2.602
$P2$	0.0658
$P3$	0.030 / (2.602-3)

P1 es el punto más bajo de la gráfica de sensibilidad a 10.000 ppm de ***CO_2***, P2 es el punto de calibración obtenido luego de la exposición del sensor a 400 ppm de ***CO_2*** y P3 es un cálculo de la pendiente de la recta donde 0.030 es la caída de voltaje del sensor cuando se mueve del aire limpio a 1000 ppm de ***CO_2***.

Luego se procede a comparar el comportamiento de ambos sensores frente a una misma situación (espiración humana) con la cual se determinó que el sensor **MG-811** arroja lecturas erróneas, para solucionar este inconveniente se hizo la calibración en un ambiente controlado, tomando nuevos puntos en la recta con recta a lo medido en un ambiente no controlado.

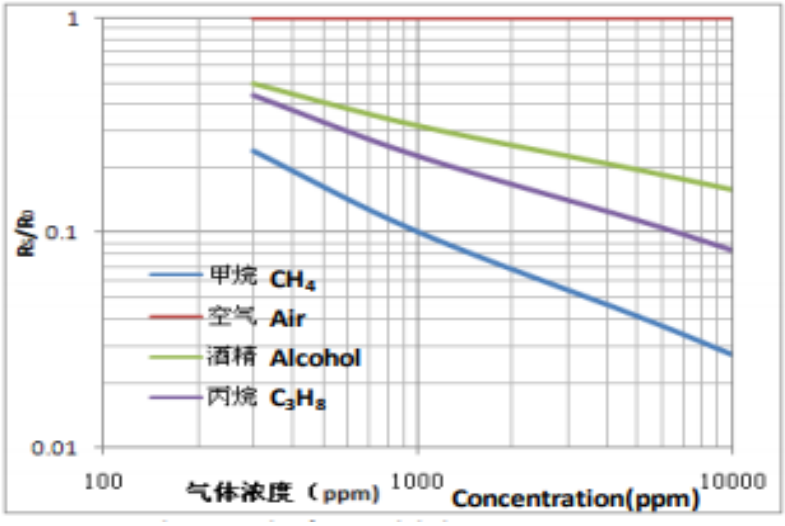
Para obtener el valor en ppm primero se inicia la librería del ADS1115 con la función ***ADS.ADS1115()***, Posteriormente se utiliza la función ***AnalogIn().voltage*** pasándole los valores de entrada de ***CO_2*** para obtener los valores de voltaje del sensor y finalmente se

hace uso de las ecuaciones mencionadas anteriormente para calcular el valor de la sustancia.

4.2.2.2. Uso del sensor MQ-4

Este sensor seleccionado para medir metano, funciona de manera similar a los otros sensores MQ, este cuenta con una curva de sensibilidad en escala logarítmica y el fabricante recomienda para su óptimo funcionamiento una resistencia de carga de 20kΩ.

Figura 21: Curva de sensibilidad MQ4



Hanwei Electronics (En línea). Datasheet MQ-4 sensor [Figura]. Recuperado de: https://html.alldatasheet.es

Para el uso de este sensor se realizó la reconstrucción de la curva de sensibilidad al ***CH_4*** teniendo en cuenta las mediciones obtenidas en paralelo con un sensor de referencia, en este caso el sensor ***ALTAIR 5x*** obteniendo la siguiente ecuación:

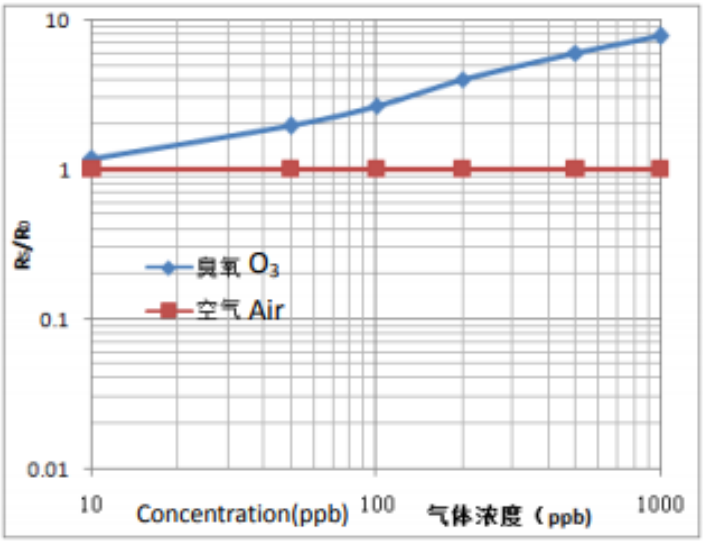
$$CH4\ ppm\ = (Voltage\ +\ 0.0148148)\ /\ (0.002074074)$$

Para obtener los valores de voltaje primero se inicia la librería del ADS1115 con la función ***ADS.ADS1115()***, Posteriormente se utiliza la función ***AnalogIn().voltage*** pasándole los valores de entrada de ***CH_4*** y finalmente se hace uso de las ecuaciones mencionadas anteriormente para calcular el valor de la sustancia en ppm.

4.2.2.3. Uso del sensor MQ-131

Con este sensor se realiza la medición de Ozono y su funcionamiento es igual al del MQ4 mostrado anteriormente, a diferencia que los valores de su curva son distintos, su resistencia de carga es de 1MΩ y su curva característica es mostrada a continuación:

Figura 22: Curva de sensibilidad MQ131



Zhengzhou Winsen Electronics Technology Co (En línea). Datasheet MQ-131 sensor [Figura]. Recuperado de: https://aqicn.org

Los valores de Ozono en ppm son obtenidos leyendo los valores de la resistencia Rs y haciendo la relación con la resistencia medida en aire limpio Ro ($\frac{Rs}{Ro}$), su ecuación es la siguiente:

$$O3\ ppm = 10^{log_{10}(\frac{ratio-P2}{P3} - P1)}$$

Por último sus valores de la curva son mostrados en la **Tabla 4**

Tabla 4: Valores de la curva de O_3

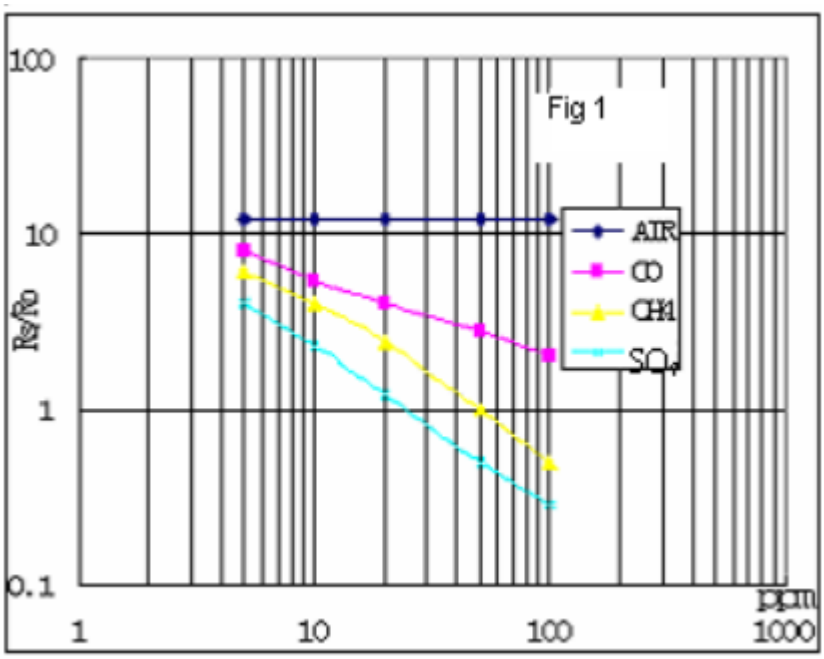
$P1$	2.3
$P2$	0.53
$P3$	-0.44

4.2.2.4. Uso del sensor MQ-136

Al igual que los sensores anteriores este sensor analógico electroquímico funciona debido a la variación de su resistencia, cuando es expuesto a la sustancia determinada, en este caso SO_2 lo cual hace necesario la inclusión del divisor de tensión para poder medirla y luego efectuar la típica relación con la medición previamente hecha en ambiente libre de la sustancia. Su resistencia de carga varía entre 30KΩ-200KΩ para su buen funcionamiento.

Su curva de sensibilidad es mostrada en la **Figura 23** y la ecuación que rige su funcionamiento en pmm es la misma que en los anteriores sensores MQ.

Figura 23: Curva de sensibilidad MQ136



Hanwei Electronics (En línea). Datasheet MQ-136 sensor [Figura]. Recuperado de: http://www.sensorica.ru

4.3.1 Instalación del gestor de paquete PIP

Según su web oficial, PIP es un sistema de gestión de paquetes utilizado para instalar y administrar paquetes de software escritos en Python, para su instalación es necesario seguir los siguientes pasos:

- ***sudo apt update***: Con este comando se actualiza el índice de paquetes de la Raspberry Pi 4.
- ***sudo apt-get install python3-pip***: Con este comando se instala el gestor de paquetes PIP y todas sus dependencias.

4.3.2 Desarrollo del software para LORA-02

Para poder utilizar este dispositivo fue necesario instalar la librería ***pyLoRa*** mediante el gestor de paquetes ***pip***, esta librería nos proporciona las conexiones físicas, el modo de uso, una prueba unitaria para evaluar el funcionamiento de cada dispositivo LORA así como una versión de ejemplo de servidor y cliente para poder enviar y recibir información.

Esta librería viene preconfigurada a 868 Mhz y debido a que este dispositivo trabaja a una frecuencia de operación estándar es necesario configurar a 433 MHz con la ayuda de la función ***set_freq()*** incluida dentro de esta librería. Debido a la hoja de datos dada por el fabricante, la potencia de salida se configura a 18dBm y la tasa de codificación a Cr = 4/5 mediante la función ***set_pa_config()*** y ***set_coding_rate()*** respectivamente.

4.3.2.1 Desarrollo del sofware para el servidor LORA-02

La parte del software del servidor LORA-02 cuenta con tres sencillos pasos los cuales son: Solicitar información, recibir información y decodificar información.

- ***Solicitar información***: En este paso el servidor pregunta por la información transmitiendo una palabra clave, que será escuchada por el cliente, para esto la palabra clave es cargada al payload (carga útil) y luego configurar el dispositivo en modo transmisión con la función ***set_mode(MODE.TX***) hasta que el cliente está disponible para enviar la información.

- ***Recibir información***: Una vez el cliente está disponible para enviar la información, el servidor entra en modo recepción, para esto se utiliza la función ***set_mode(MODE.RXCONT)*** y luego que el cliente envía la información al servidor, dicha información es cargada a una variable mediante la función ***read_payload(nocheck=True).***

- ***Decodificar información***: La información recibida se encuentra codificada en código ASCII, por lo tanto es necesario decodificarla a utf-8 usando la función ***decode("UTF-8", 'ignore')***, para esto es necesario que la información esté en bytes, por ende se usa la función ***bytes()*** y así almacenarla en una variable para su uso requerido, en este caso cargarla a la base de datos.

4.3.2.2 Desarrollo del sofware para cliente LORA-02

Una vez la información ha sido recolectada y está lista para ser transmitida mediante el dispositivo LORA-02, se deben realizar tres sencillos pasos los cuales son: Codificar la información, esperar al servidor y transmitir información.

- ***Codificar la información****:* El protocolo de transmisión del módulo LORA-02 exige enviar la información en formato ASCII, por consiguiente es necesario codificar la información obtenida de los sensores a este formato, para esto se hace uso de una función nativa de python llamada ***ord(),*** para que la librería ***pyLoRa*** permita su transmisión.

- ***Esperar al servidor***: Para ser transmitida la información se debe asegurar que el servidor esté listo para recibirla, para esto inicialmente el dispositivo se configura en modo recepción con la función ***set_mode(MODE.RXCONT***) hasta que el servidor envíe la palabra clave, indicando que está listo para recibir.

- ***Transmitir información***: Una vez el servidor esté listo para recibir se pasa la información a la carga útil con la función ***write_payload()*** y finalmente es transmitida al servidor usando la función ***set_mode(MODE.TX***).

4.3.3 Conexiones del hardware para LORA-02

El dispositivo de transmisión inalámbrica LORA-02 cuenta con un chip **ESP8266** que funciona a una frecuencia estándar de 433 MHz, por tanto, es necesario el uso de una antena para garantizar la distancia a la que será transmitida la información. En la **Tabla 5** podemos observar las conexiones hacia la Raspberry Pi según la librería ***PyLoRa.***

Tabla 5: Conexiones físicas del LORA- 02 a RPi

Ra - 02 LoRa	***GPIOs Raspberry Pi***
MOSI	*GPIO 10*
MISO	*GPIO 9*
SCK (SCLK)	*GPIO 11*

NSS	*GPIO 8 (CEO)*
DIO0 (IRQ)	*GPIO 4*
DIO1	*GPIO 17*
DIO2	*GPIO 18*
DIO3	*GPIO 27*
RST (Reset)	*GPIO 22*

4.4. Cálculo Índice de calidad del aire (ICA)

El objetivo en este apartado es mostrar en el interfaz para el usuario el color de la alerta en la que se encuentra el índice de calidad del aire. Para esto se toma como base la tabla otorgada por IDEAM mostrada en la **Tabla 6**.

Tabla 6: Puntos de cortes para el cálculo del ICA

Rangos *ICA*	Clasificación	O_3 8h (ppm)	O_3 1h (ppm)[1]	PM_{10} 24h ($\mu g/m^3$)	$PM_{2,5}$ 24h ($\mu g/m^3$)	CO 8h (ppm)	SO_2 24h (ppm)	NO_2 1h (ppm)
$0 \leq ICA \leq 50$	Verde	0,000 0,059	-	0 54	0,0 15,4	0,0 4,4	0,000 0,034	(2)
$51 \leq ICA \leq 100$	Amarillo	0,060 0,075	-	55 154	15,5 40,4	4,5 9,4	0,035 0,144	(2)
$101 \leq ICA \leq 150$	Anaranjado	0,076 0,095	0,125 0,164	155 254	40,5 65,4	9,5 12,4	0,145 0,224	(2)
$151 \leq ICA \leq 200$	Rojo	0,096 0,115	0,165 0,204	255 354	65,5 150,4	12,5 15,4	0,225 0,304	(2)
$201 \leq ICA \leq 300$	Morado	0,116 0,374 (0,155 0,404) (4)	0,205 0,404	355 424	150,5 250,4	15,5 30,4	0,305 0,604	0,65 1,24
$301 \leq ICA \leq 400$	Marrón	(3)	0,405 0,504	425 504	250,5 350,4	30,5 40,4	0,605 0,804	1,25 1,64
$401 \leq ICA \leq 500$	Marrón	(3)	0,505 0,604	505 604	350,5 500,4	40,5 50,4	0,805 1,004	1,65 2,04

Hernández A (2013).Hoja metodológica del indicador índice de calidad del aire - ICA (versión 1,00) [Figura]. Recuperado de: http://www.ideam.gov.co

Teniendo en cuenta lo anterior, se realiza el promedio de los valores medidos en el tiempo estipulado por IDEAM de acuerdo a la variable física requerida y así, determinar la clasificación en la que se encuentra el índice de calidad del aire en el lugar de la implementación.

4.5 Instalación del servidor WEB

La instalación del servidor web se realiza desde el terminal de la Raspberry Pi, ingresando las líneas de código ***sudo apt update && sudo apt upgrade*** y ***sudo apt install apache2:***

- ***sudo apt update && sudo apt upgrade***: Mediante esta línea de código se verifica que el dispositivo y sus librerías se encuentren en su última versión y así la instalación se realice de manera satisfactoria.

- ***sudo apt install apache2***: Mediante esta instrucción se instala el servidor web ***Apache2***. Para confirmar que la instalación sea exitosa se accede desde el navegador de la Raspberry Pi a la dirección local la cuál es ***127.0.0.1*** obteniendo como resultado una página web con la versión de ***Apache2*** instalada.

Ahora la página web debe alojarse en el siguiente directorio de la Raspberry Pi ***/var/www/html***.

4.6 instalación de PHP

La instalación de php se realiza desde la terminal del sistema operativo de la Raspberry Pi, escribiendo el comando ***sudo apt install php***.

Para verificar que la instalación ha sido exitosa se crea un archivo desde la terminal al directorio donde se aloja la página web, con el comando ***sudo nano /var/www/html/p.php*** con la línea de código en php ***<?php echo "Mensaje de prueba"; ?>***. Luego se reinicia el servicio de ***apache2*** con ***sudo service apache2 restart*** y por último desde el navegador del dispositivo se accede a la dirección local **127.0.0.1**, obteniendo como resultado el mensaje de prueba, lo que confirma la instalación.

4.7. Diseño del sistema de almacenamiento local y remoto

Es necesario tener un sistema de almacenamiento para guardar la información proveniente de los sensores, en este caso la base de datos creada con ***MySQL***. De acuerdo a esto el usuario podrá acceder a estos datos mediantes la página web de visualización que se encuentra alojada en el servidor local del sistema, así como en el remoto, ubicado estratégicamente en el campus universitario.

4.7.1. Diseño de la base de datos

La base de datos es creada con ***MySQL*** a través de ***PHPMyAdmin***, se diseña teniendo en cuenta la información que es necesaria almacenar, como lo es temperatura, humedad, fecha, hora y el resto de variables físicas medidas en este proyecto. Dicha base de datos es protegida con contraseña para que solo sea posible tener acceso a través de la página web adaptada para este fin y por los administradores si es necesario agregar o eliminar información de la base de datos y corregir errores de ser necesario.

Esta base de datos tendrá una copia idéntica en la localización remota, dichos datos serán transmitidos inalámbricamente hasta ese punto a través de la tarjeta de LORA-02.

Para el diseño de esta base de datos, es necesario la instalación del servicio ***MySQL***, ingresando los siguientes comandos desde la terminal de la Raspberry Pi:

- ***sudo apt install mariadb-server php-mysql***: Con este comando se instala tanto el servicio ***MySQL*** y las librerías necesarias para la conexión de la base de datos y php.

- ***sudo service apache2 restart***: Es necesario reiniciar el servicio de ***Apache2*** mediante esta línea de comando para la instalación de ***MySQL*** se acople en el servidor.

Una vez realizada la instalación es necesario realizar una configuración inicial para iniciar el servicio ***MySQL***, Por consiguiente se ejecuta el comando: ***sudo mysql_secure_installation***.

4.7.1.1 Iniciar servicio de MySQL

Para iniciar el servicio de ***MySQL*** se ingresa en el terminal de la Raspberry Pi el comando ***sudo mysql -u root*** y se procede a comenzar el servicio de gestión de base de datos ***MySQL***, para gestionar las base de datos creadas es necesario un usuario y contraseña que lo podemos crear usando la instrucción ***CREATE USER 'usuario'@'%' IDENTIFIED BY 'contraseña';*** asignando el nombre de usuario y contraseña correspondiente, para lograr administrar correctamente la base de datos es necesario otorgar todos los privilegios al usuario anteriormente creado mediante los comando ***GRANT ALL PRIVILEGES ON *.* to 'usuario'@'%';*** y ***FLUSH PRIVILEGES;***. Por último se escribe ***exit;*** para salir de ***MySQL***.

Con el fin de permitir conexiones remotas se realizan cambios en el archivo de configuración de ***MySQL,*** ubicado en el directorio /***etc/mysql/mariadb.conf.d/*** se edita ***50-server.cnf***, cambiando el ***bind-address*** a 0.0.0.0.

4.7.2 Instalación de PHPMyAdmin

Para la instalación de este servicio es necesario ejecutar los siguientes comandos:

- ***sudo apt install phpmyadmin*:** Encarga de instalar las librerías necesarias para que ***PHPMyAdmin*** pueda ejecutarse.

- ***sudo phpenmod mysqli*:** Mediante esta instrucción se enlaza ***PHPMyAdmin*** con el servicio ***MySQL***.

- ***sudo service apache2 restart*:** Es necesario reiniciar el servicio de ***Apache2*** mediante esta línea de comando para la instalación de ***PHPMyAdmin*** se acople en el servidor.

4.7.2.1 Uso de PHPMyAdmin

Una vez se encuentra instalada esta herramienta, otorga la posibilidad de crear la base de datos, para esto es necesario acceder desde el navegador web a la direccion local ***127.0.0.1*** o la que el router le asigne a la Raspberry Pi, concatenando ***/phpmyadmin*** que dirige a la página web de ***PHPMyAdmin*** que permite gestionar la base de datos.

Inicialmente se requiere el usuario y contraseña anteriormente creado al iniciar el servicio de ***MySQL*** para iniciar sesión, luego se procede a creer la base de datos seleccionado la opción ***Nueva*** ubicada al costado izquierdo de la página web con los requisitos nombre y el tipo de cotejamiento de la base de datos, para este caso es ***utf8_general_ci.***

Una vez creada la base de datos, se asigna el nombre de la tabla que almacena los datos recolectados, en este caso llamada ***mediciones,*** con el número de columnas necesarias para el uso requerido, contando tanto el ID que es la referencia que indica la fila del dato, la fecha, la hora y la medición de los sensores serán 14 columnas.

Se procede a nombrar cada una de las columnas, asignando el tipo de dato y la longitud del mismo. Para el caso específico del ID tiene el atributo de número sin signo, con índice primario y el extra de ser un valor autoincremental como se observa en la **figura 24**.

Figura 24: Estructura de las columnas en la tabla de la base de datos

	#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra
☐	1	**id** 🔑	bigint(11)		UNSIGNED	No	*Ninguna*		AUTO_INCREMENT
☐	2	**fecha**	date			No	*Ninguna*		
☐	3	**hora**	time			No	*Ninguna*		
☐	4	**pm2_5**	int(11)			No	*Ninguna*		
☐	5	**pm10**	int(11)			No	*Ninguna*		
☐	6	**so2**	decimal(4,3)			No	*Ninguna*		
☐	7	**no2**	decimal(5,4)			No	*Ninguna*		
☐	8	**co2**	decimal(10,3)			No	*Ninguna*		
☐	9	**co**	decimal(9,4)			No	*Ninguna*		
☐	10	**ozono**	decimal(4,3)			No	*Ninguna*		
☐	11	**metano**	decimal(9,4)			No	*Ninguna*		
☐	12	**temperatura**	decimal(3,1)			No	*Ninguna*		
☐	13	**humedad**	decimal(3,1)			No	*Ninguna*		
☐	14	**ica**	decimal(4,1)			No	*Ninguna*		

4.8. Envío de información a la base de datos

Para enviar la información a la base de datos es necesaria la instalación de la librería ***mysql.connector,*** ejecutando el comando ***pip3 install mysql-connector-python*** en el terminal y seguir una serie de pasos para que la información quede almacenada.

4.8.1 Pasos para enviar la información

- Importación de la librería ***mysql-connector.***

- Creación del objeto que realiza la conexión a la base de datos con la función ***mysql.connect()*** pasando los parámetros que la identifican (anfitrión, usuario, contraseña y nombre).

- Creación del cursor en la conexión de la base de datos.

- Creación del query que será enviado a la base de datos ***MySQL*** con la estructura con los nombres de las columnas que se postea la nueva información.

- Creación de una tupla con la información que será enviada a la base de datos, con el orden requerido por la estructura del query.

- Mediante el cursor se ejecuta el query y la tupla a través de la función ***execute()***, procediendo a ejecutar la función ***commit()*** en la conexión a la base de datos.

- Se cierra el cursor y la conexión a la base de datos con la función ***close()***.

4.9. Diseño de interfaz web

Para esta interfaz basada en ***HTML***, ***CSS***, ***JavaScripts*** y ***PHP*** se busca que el usuario pueda tener acceso a los datos almacenados en el servidor de una manera simple y rápida, dando la posibilidad de observar el comportamiento de las sustancias de manera gráfica realizadas en JavaScrip para su posible análisis e interpretación.

Cuenta con varios filtros para realizar búsqueda de datos así como para ordenarlos y la posibilidad de visualizarlos por hora de registro, mayor a menor o viceversa con posibilidad de limitar la cantidad de datos a mostrar. Dichos filtros fueron realizados con ***HTML*** Y ***CSS*** como gran parte de esta plataforma, las cuales generan una petición en ***PHP*** a la base de datos ***MySQL***.

El argumento principal para la búsqueda de los datos a través de la página web, es ingresando la fecha de una manera fácil y gráfica, generando la petición a la base de datos.

- ***Estructura básica de la página web****:* Para la realización de la estructura se hace uso del lenguaje de marcado *HTML* con el fin de definir la posición de menús, texto fijo, gráficas y tablas que mostraran el contenido deseado, teniendo en cuenta la estructura básica de *HTML5* donde todo el contenido será alojado en la etiqueta <*body*>, para esto se utilizan etiquetas específicas de acuerdo a la necesidad ya sean de tipo texto, imágenes, tablas entre otras.

- ***Apariencia de la página web***: Este interfaz web debe adaptarse al tamaño de las diferentes pantallas en las cuales será visualizada la información, para esto se implementa el lenguaje de diseño gráfico ***CSS***, que mediante el uso de ***media queries*** con los métodos ***FLEX*** y ***DRIP*** se logra cambiar el modo de trabajo de pixeles a porcentajes o número de bloques en pantalla, obteniendo como resultado una página que se adapte a cualquier tipo de pantalla; Además de lo anterior, con ***CSS*** es posible modificar los estilos de la página web tales como tamaño de texto, colores, bordes, tipografía, fondos, márgenes, etc.
Mediante los ***media queries*** es posible detectar el tamaño de pantalla del dispositivo en el cual se visualiza el interfaz web, una vez la barra de navegación no sea totalmente accesible por el usuario, el formato de la página cambia haciendo visibles o invisibles tanto la barra de navegación o el botón de menú según sea requerido.

- ***Interactividad página web:*** El usuario interactúa con la página web para obtener información deseada, para ello con ***HTML*** se generan botones o enlaces a páginas secundarias, cuando el usuario interactúa con estos objetos, por medio del lenguaje de programación ***JavaScript*** se escuchan estos eventos accionando la función establecida, por ejemplo, un botón que despliega un menú de navegación. Además de esto, con el uso de este lenguaje se realizan otras funciones, tales como hacer consultas al servidor sin necesidad de recargar la página para actualizar información en tiempo real, verificar si hay datos nuevos y crear diferentes tipos de gráficas dinámicas.

- ***Conexion base de datos página web:*** La página web está diseñada para visualizar la información almacenada en la base de datos, sin importar que exista información nueva o modificada; se hace posible gracias a la comunicación que se establece entre el servidor a través de ***PHP*** y el lado del cliente con ***JavaScript***.
Mediante el método ***AJAX*** basado en ***JS,*** se logra realizar peticiones al servidor mientras ***PHP*** las escucha extrayendo información de la base de datos según sea requerido y enviarla al cliente para su visualización con la ayuda de la inserción de los datos por medio de ***JS***.

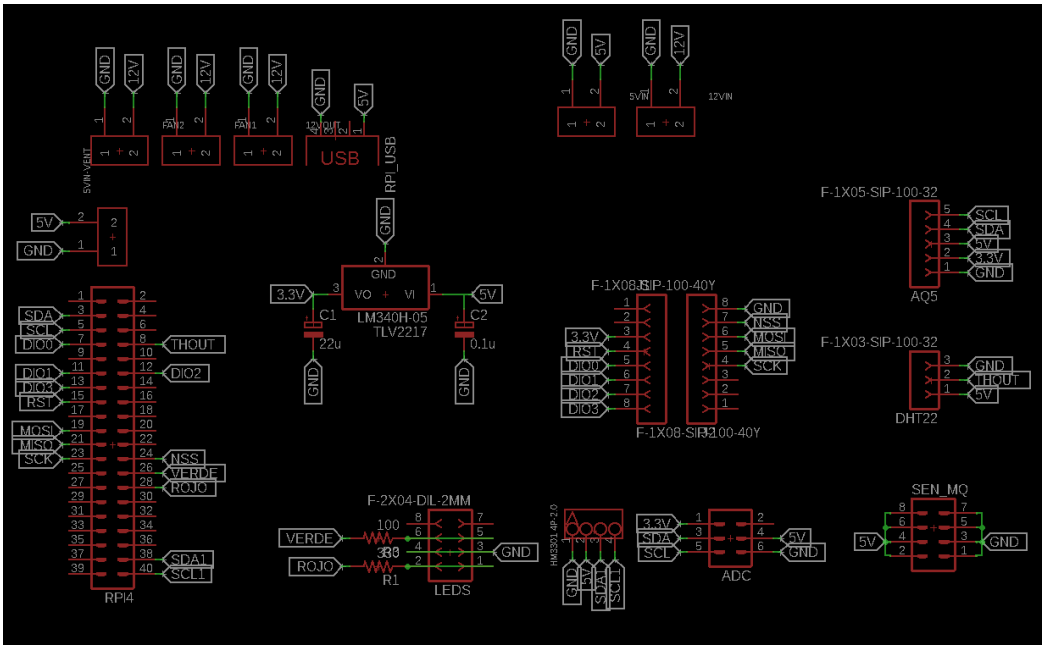
4.10. Diseño de la PCB

La PCB se diseñó utilizando el software llamado ***EAGLE***, ya que de manera gráfica facilita la creación de diseños de PCB totalmente integradas y cumpliendo con normas de diseño electrónico.

Para esto se tiene en cuenta las polaridades de los componentes, como sus medidas para evitar errores de corto circuito y de espacio limitado, cumpliendo con la necesidad de conectar múltiples dispositivos al mismo terminal de la Raspberry Pi, como el interfaz I^2C, a la vez conservar una buena conducción en las pistas y evitar el sobrecalentamiento de estas mismas preservando su vida útil.

En este diseño se conectan todos los sensores usados, la tarjeta de desarrollo, la alimentación para todos los dispositivos, el módulo de transmisión inalámbrica, así como los ventiladores encargados de recolectar todo el aire que se encuentra en el ambiente.

Figura 25: Esquemático diseño PCB



5. Resultados

5.1 Resultados obtenidos de los sensores

Luego de un tiempo de sensado de aproximadamente seis semanas se realizò la comparación de los datos obtenidos con los de referencia, en este frente a los datos obtenidos del reporte del sistema de vigilancia de la calidad del aire de Corpamag, donde se encontró que haciendo una media a de los valores obtenidos por este prototipo se obtienen valores similares teniendo en cuenta que, esta entidad realiza los reportes de manera anuales de las lecturas y el prototipo en cuestión pública arios datos al dia para generar alertas tempranas.

5.1.1 Lecturas sensores digitales

5.1.1.1 Resultado del sensor láser HM3301

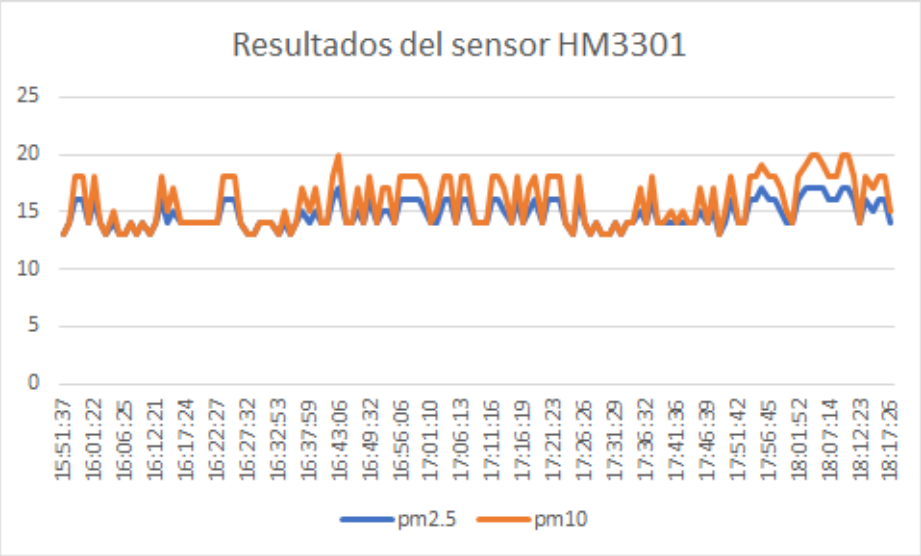
Siendo un sensor láser que entrega los resultados sin una anterior calibración, se puede observar en la **Tabla 7** la impresión de formato de los datos de **$PM_{2.5}$** y **PM_{10}** respectivamente, y en la **Figura 26** se aprecia el comportamiento del material particulado a través de un gran rango de tres horas de lectura aproximadamente.

Tabla 7: Resultados medición de $PM_{2.5}$ y PM_{10}

Fecha	Hora	$PM_{2.5}$	PM_{10}
26/08/2021	11:53:10	14	14
26/08/2021	11:54:10	13	13
26/08/2021	11:55:11	13	13
26/08/2021	11:56:11	14	15
26/08/2021	11:57:12	14	14
26/08/2021	11:58:13	14	14
26/08/2021	11:59:13	13	13
26/08/2021	12:00:14	13	13
26/08/2021	12:01:14	13	13

26/08/2021	12:02:15	14	14

Figura 26: Resultados medición de $PM_{2.5}$ y PM_{10}



5.1.1.2 Resultado del sensor DHT22

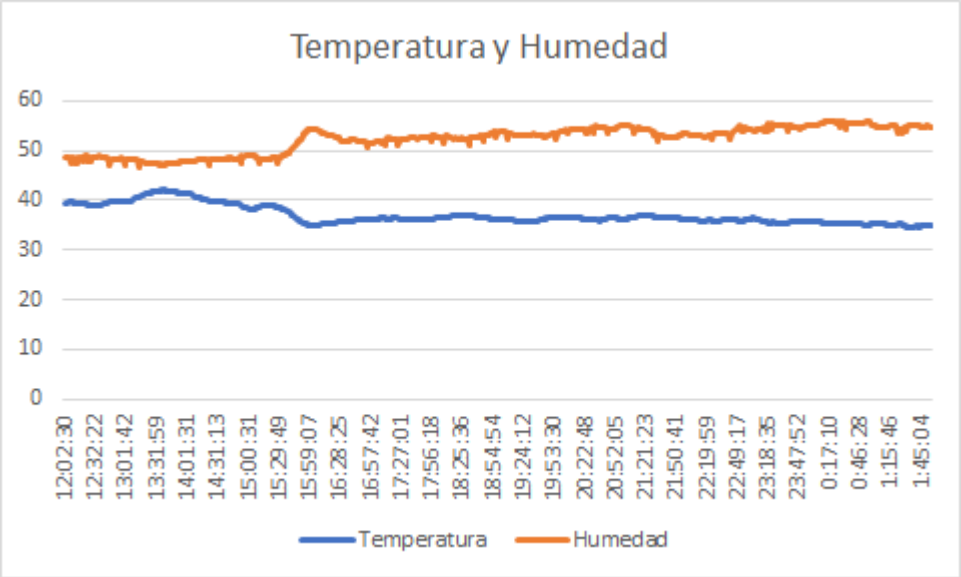
El formato de impresión de las lecturas obtenidas por el sensor DHT22 de temperatura y humedad se pueden observar de forma respectiva, en la **Tabla 8**. Donde queda claro que son lecturas más que aceptables para lo requerido, mientras que en la **Figura 27,** se puede evidenciar la captura de estas sustancias a través de un gran rango de once horas de lectura aproximadamente, observando así de forma rápida su comportamiento.

Tabla 8: Resultados medición de temperatura y humedad

Fecha	Hora	Temperatura	Humedad
26/08/2021	11:53:10	33.7	65
26/08/2021	11:54:10	33.5	64.1
26/08/2021	11:55:11	33.5	63.7
26/08/2021	11:56:11	33.5	63.7
26/08/2021	11:57:12	33.4	62.6

26/08/2021	11:58:13	32.8	64.2
26/08/2021	11:59:13	33.2	64.7
26/08/2021	12:00:14	34.8	68.2
26/08/2021	12:01:14	34.2	63.1
26/08/2021	12:02:15	33.4	66.2

Figura 27: Resultados medición de temperatura y humedad



5.1.1.3 Resultado del sensor Air Quality 5 Click

En la **Tabla 9** se evidencia la captura de los datos con su respectivo formato de impresión de las sustancias **NO_2** y **CO** respectivamente, luego de su calibración, comparando con el equipo de laboratorio **RAC 3/5.** En la **Figura 28 y 29** se puede evidenciar el censado de **NO_2** y **CO a** medida que transcurre un lapso de cuatro horas aproximadamente, observando así de forma rápida su comportamiento.

Tabla 9: Resultados medición de NO_2 y CO

Fecha	Hora	NO2	CO
26/08/2021	11:53:10	0.00019	0.36

26/08/2021	11:54:10	0.00019	0.6
26/08/2021	11:55:11	0.0002	0.6
26/08/2021	11:56:11	0.0002	0.6
26/08/2021	11:57:12	0.0002	0.6
26/08/2021	11:58:13	0.0002	0.6
26/08/2021	11:59:13	0.0002	0.6
26/08/2021	12:00:14	0.0002	0.61
26/08/2021	12:01:14	0.0002	0.6
26/08/2021	12:02:15	0.00018	0.61

Figura 28: Resultados medición de NO_2

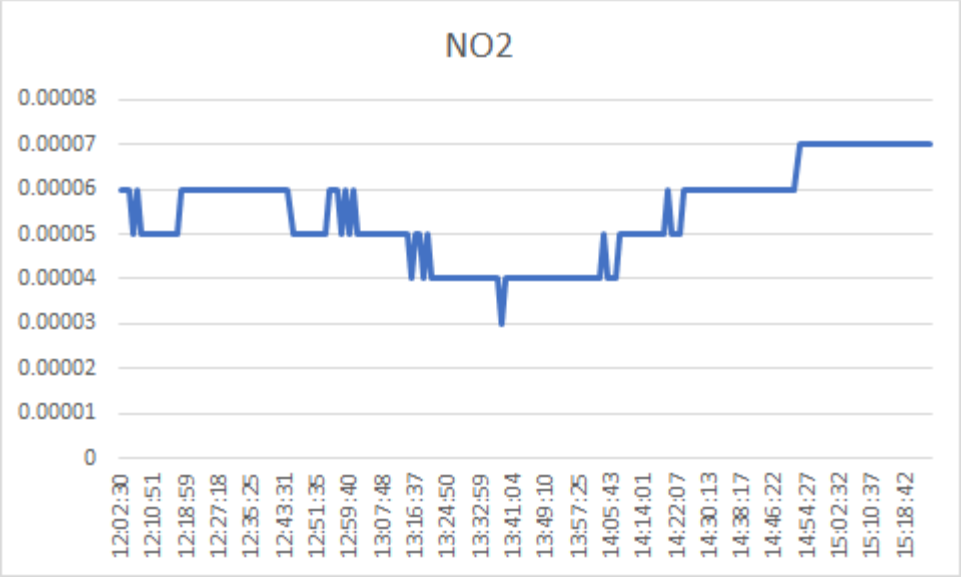
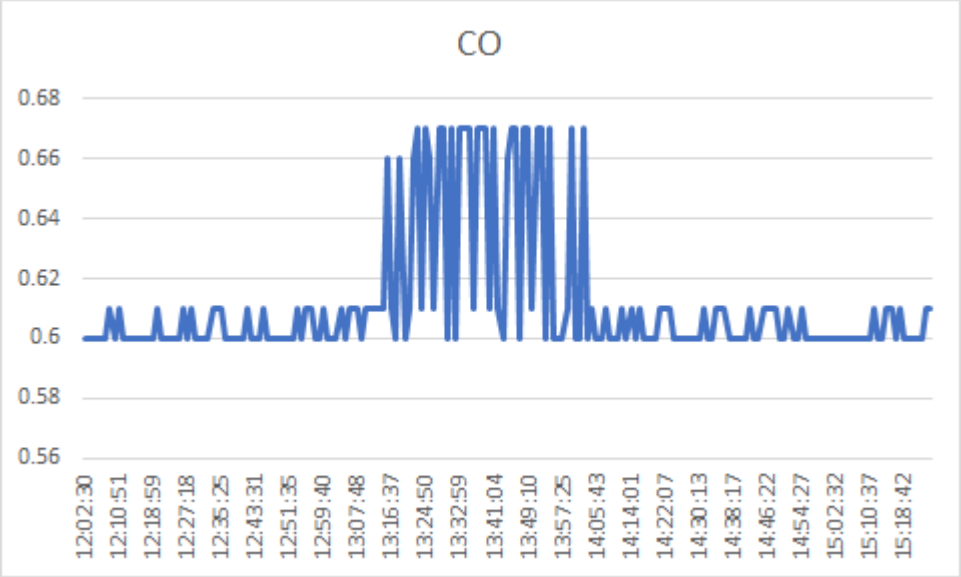


Figura 29: Resultados medición de CO



5.1.2 Lecturas sensores analógicos

5.1.2.1 Resultado del sensor MG-811

Las lecturas de este sensor se pueden observar en la **Tabla 10**, donde se evidencia unas lecturas aceptables de **CO_2**, debido a su proceso de calibración y calentamiento antes de realizar la programación, adicionalmente se puede apreciar el formato de impresión de datos con fecha hora y el valor de la sustancia medible en PPM.

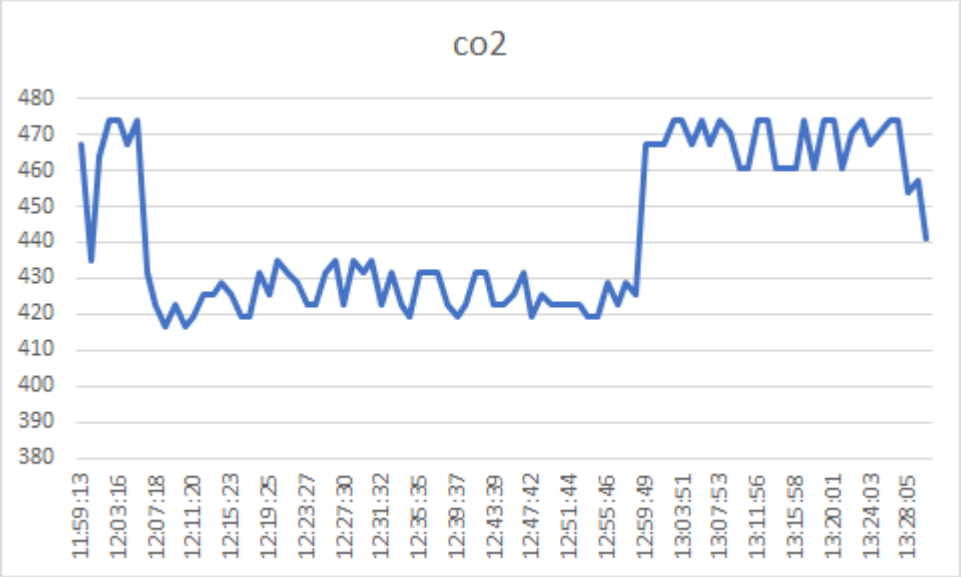
Para analizar el comportamiento de esta sustancia a través del tiempo, es necesario consultar la **Figura 30**, donde se observan de manera gráfica las mediciones obtenidas con este sensor en un lapso de tres horas aproximadamente.

Tabla 10: Resultados medición de CO_2

Fecha	Hora	CO2(ppm)
26/08/2021	11:53:10	422.527
26/08/2021	11:54:10	474.024
26/08/2021	11:55:11	447.535
26/08/2021	11:56:11	419.5

26/08/2021	11:57:12	460.59
26/08/2021	11:58:13	474.024
26/08/2021	11:59:13	467.259
26/08/2021	12:00:14	434.851
26/08/2021	12:01:14	463.912
26/08/2021	12:02:15	474.024

Figura 30: Resultados medición de CO_2



5.1.2.2 Resultado del sensor MQ-4

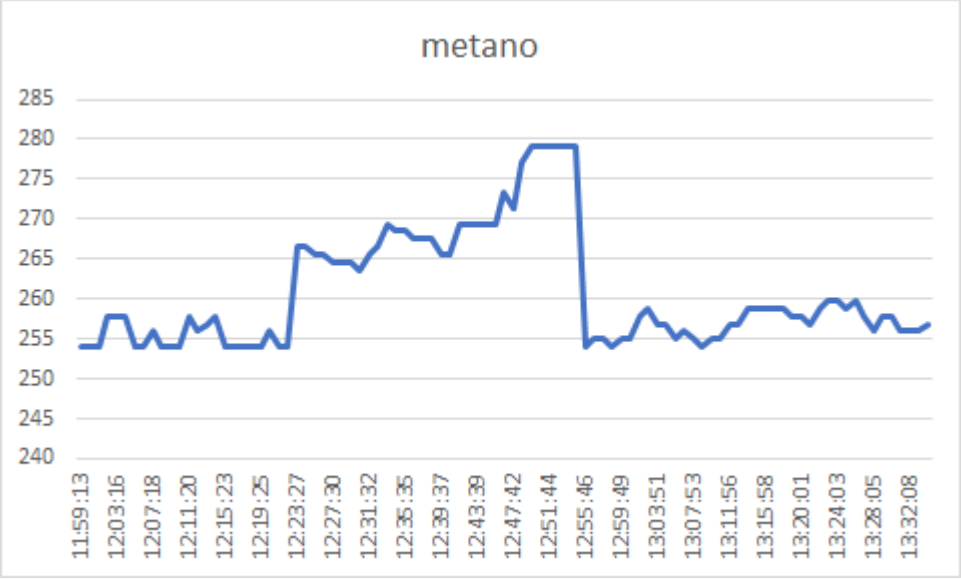
El resultado obtenido de la captura de datos del sensor de **CH_4** con su formato de impresión establecido puede observarse en la figura **Tabla 11**, la cual evidencia el calentamiento y calibración exitosa de este sensor para esta medición del gas, y así posteriormente ser enviada a la base de datos.

En la **Figura 31** se puede observar de forma gráfica los datos obtenidos por el sensor MQ-4 en un rango de dos horas aproximadamente, para observar su comportamiento a medida que transcurre el tiempo.

Tabla 11: Resultados medición de CH_4

Fecha	Hora	CH4
26/08/2021	11:53:10	254.0096
26/08/2021	11:54:10	254.0096
26/08/2021	11:55:11	254.0096
26/08/2021	11:56:11	257.8668
26/08/2021	11:57:12	257.8668
26/08/2021	11:58:13	257.8668
26/08/2021	11:59:13	254.0096
26/08/2021	12:00:14	254.0096
26/08/2021	12:01:14	255.9382
26/08/2021	12:02:15	254.0096

Figura 31: Resultados medición de CH_4



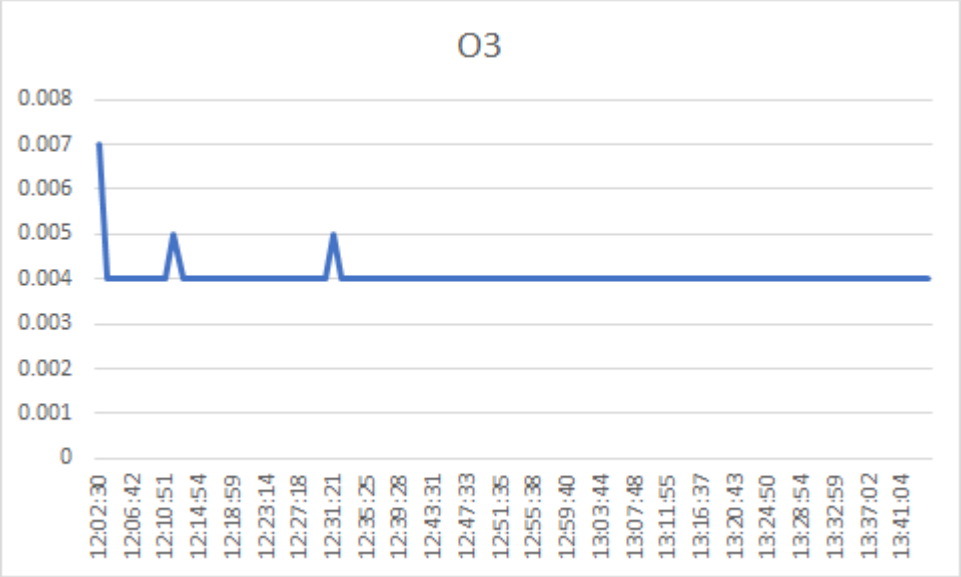
5.1.2.3 Resultado del sensor MQ-131

En la **Tabla 12,** se observa de manera clara el formato de publicación de datos lecturas de **O_3,** que de acuerdo al proceso de calentamiento y adquisición de datos recomendaciones del fabricante. Mientras que su comportamiento a través del tiempo es evidenciable en la **Figura 32**, donde se muestran las mediciones obtenidas por este sensor en un rango de dos horas aproximadamente.

Tabla 12: Resultados medición de O3

Fecha	Hora	O3
26/08/2021	11:53:10	0.007
26/08/2021	11:54:10	0.004
26/08/2021	11:55:11	0.004
26/08/2021	11:56:11	0.004
26/08/2021	11:57:12	0.004
26/08/2021	11:58:13	0.004
26/08/2021	11:59:13	0.004
26/08/2021	12:00:14	0.004
26/08/2021	12:01:14	0.004
26/08/2021	12:02:15	0.004

Figura 32: Resultados medición de O3



5.1.2.4 Resultado del sensor MQ-136

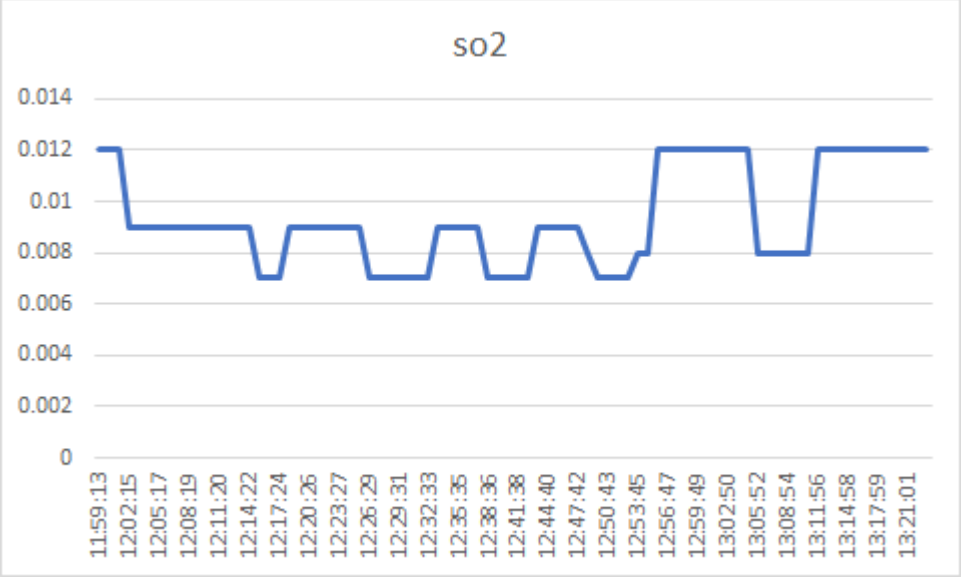
En la **Tabla 13,** se evidencia la captura de los datos de **SO_2** con su formato establecido luego de su calibración fue comparado con el equipo de laboratorio **RAC 3/5**. En la **Figura 33** se muestran de manera gráfica los resultados obtenidos por este sensor a través de dos horas de medición aproximadamente, para su fácil interpretación y análisis.

Tabla 13: Resultados medición de SO_2

Fecha	Hora	SO2
26/08/2021	11:53:10	0.007
26/08/2021	11:54:10	0.007
26/08/2021	11:55:11	0.007
26/08/2021	11:56:11	0.007
26/08/2021	11:57:12	0.007
26/08/2021	11:58:13	0.007
26/08/2021	11:59:13	0.012

26/08/2021	12:00:14	0.012
26/08/2021	12:01:14	0.012
26/08/2021	12:02:15	0.009

Figura 33: Resultados medición de SO_2



5.2. Resultados sistema de almacenamiento remoto

El sistema de almacenamiento está compuesto por la página de visualización y la base de datos, a continuación se observan los resultados obtenidos.

5.2.1 Resultados página de visualización

La página web de visualización resultante, mostrada en las **Figuras 34, 35** y **36** cuenta con un espacio para observar el comportamiento de cada una de las variables censadas en tiempo real de manera numérica y gráfica, así mismo también es posible realizar la búsqueda de una hora específica para corroborar el comportamiento de las sustancias y observar el historial de mediciones por días, ordenarlo por hora, magnitud del valor sensado entre otros.

Adicionalmente, la página web de visualización está diseñada de manera ***responsive***, es decir que se puede visualizar en todos los dispositivos, como se observa en las **Figuras 37, 38** y **39.**

Figura 34: Página de visualización inicio

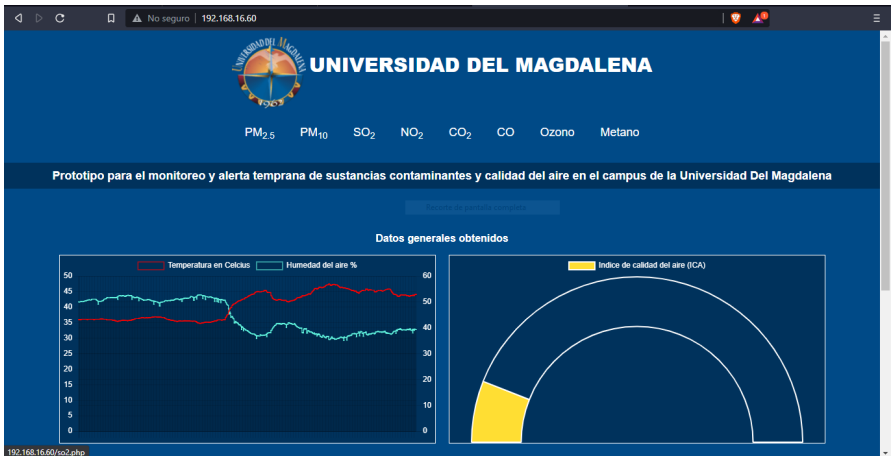


Figura 35: Página de visualización historial de datos

Fecha	Hora	CO_2 (PPM)
Monday, 20 de September de 2021	00:00:11	447.535
Monday, 20 de September de 2021	00:01:12	447.535
Monday, 20 de September de 2021	00:02:13	460.590
Monday, 20 de September de 2021	00:03:13	454.016
Monday, 20 de September de 2021	00:04:14	447.535
Monday, 20 de September de 2021	00:05:15	447.535
Monday, 20 de September de 2021	00:06:15	454.016
Monday, 20 de September de 2021	00:07:16	454.016
Monday, 20 de September de 2021	00:08:16	447.535
Monday, 20 de September de 2021	00:09:17	454.016
Monday, 20 de September de 2021	00:10:18	447.535
Monday, 20 de September de 2021	00:11:18	454.016
Monday, 20 de September de 2021	00:12:19	454.016
Monday, 20 de September de 2021	00:13:19	454.016
Monday, 20 de September de 2021	00:14:20	460.590
Monday, 20 de September de 2021	00:15:21	447.535
Monday, 20 de September de 2021	00:16:21	441.148

Figura 36: Página de visualización sustancias individuales

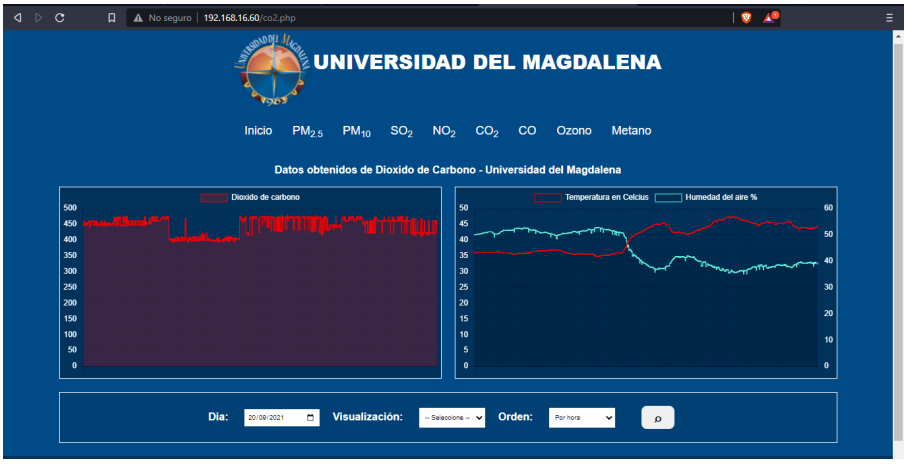


Figura 37: Página de visualización inicio - versión móvil

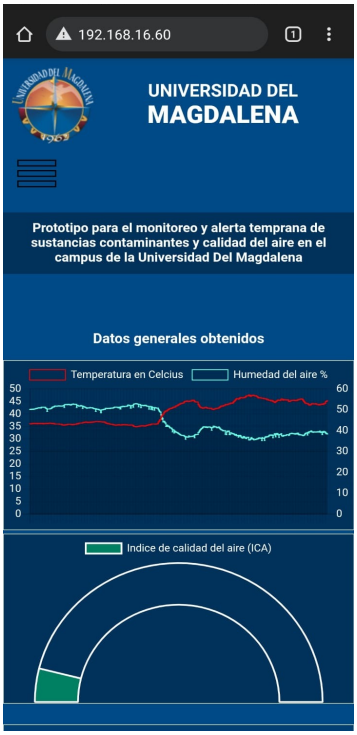
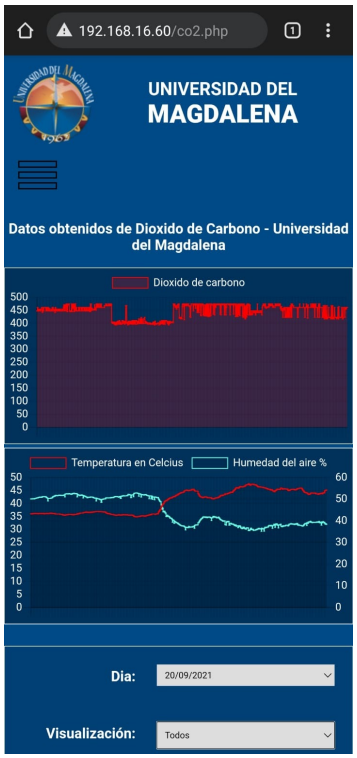


Figura 38: Página de visualización historial de datos - versión móvil

Fecha	Hora	CO_2 (PPM)
Monday, 20 de September de 2021	00:00:11	447.535
Monday, 20 de September de 2021	00:01:12	447.535
Monday, 20 de September de 2021	00:02:13	460.590
Monday, 20 de September de 2021	00:03:13	454.016
Monday, 20 de September de 2021	00:04:14	447.535
Monday, 20 de September de 2021	00:05:15	447.535
Monday, 20 de September de 2021	00:06:15	454.016
Monday, 20 de September de 2021	00:07:16	454.016
Monday, 20 de September de 2021	00:08:16	447.535
Monday, 20 de September de 2021	00:09:17	454.016
Monday, 20 de September de 2021	00:10:18	447.535
Monday, 20 de September de 2021	00:11:18	454.016
Monday, 20 de September de 2021	00:12:19	454.016
Monday, 20 de September de 2021	00:13:19	454.016
Monday, 20 de September de 2021	00:14:20	460.590

Figura 39: Página de visualización sustancias individuales - versión móvil

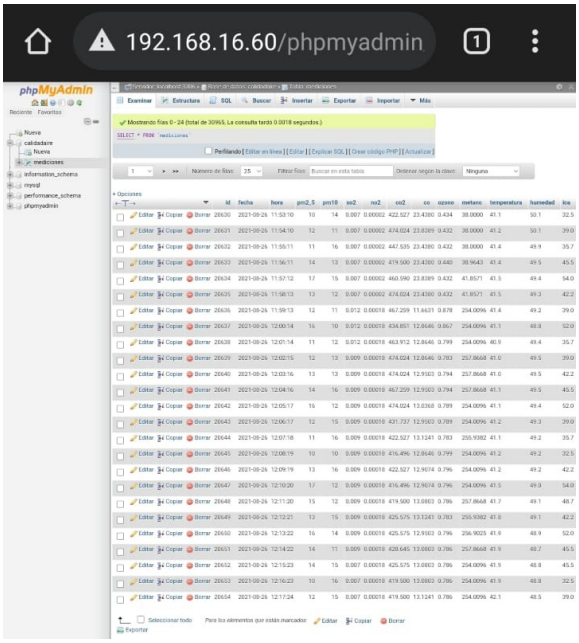


5.2.2 Resultado base de datos

En cuanto a la base de datos su resultado se puede observar en la **Figura 40**, donde se evidencia cómo son almacenados los datos provenientes de los sensores que posteriormente serán visualizados en la página web.

Dicha base de datos es gestionada gracias a los servicios de código abierto de ***PHPMyAdmin*** donde es posible editar el tipo de datos de las lecturas, el tamaño del dato agregar/eliminar columnas y limpiar los datos de la forma como sea requerido.

Figura 40: Base de datos en PHPMyAdmin



5.3. Resultados de la PCB

Luego de realizar las conexiones de las pistas con la función de autoruteo que facilita el programa, se tiene como resultado dos capas que son enviadas a producción, tanto la capa TOP, la cual es la capa del diseño superior, como la capa BOTTOM, la cual es la capa inferior del diseño de esta PCB, tal cual se aprecian en las **Figuras 41** y **42**.

Figura 41: Diseño final PCB parte superior

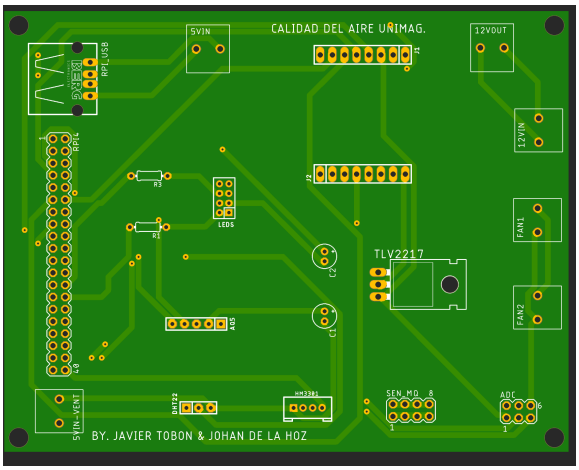
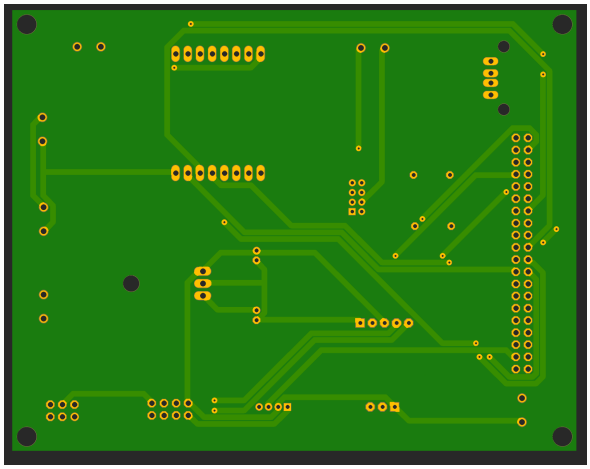


Figura 42:Diseño final PCB parte inferior



5.4 Resultado del montaje

En la **Figura 43,** se observa el resultado final de la PCB con todos los sensores dentro de su caja ip encargada de proteger estos dispositivos del agua, el polvo y del medio ambiente en general.

Así mismo en la **Figura 44,** se puede apreciar la totalidad del proyecto instalado en la estructura que sostiene los paneles, la caja y guarda la batería que alimenta los dispositivos.

Por último en la **Figura 45,** se observa el montaje final del servidor encargado de recibir la información obtenida por los sensores y enviarla al sistema de almacenamiento remoto.

Figura 43: Resultado montaje interior

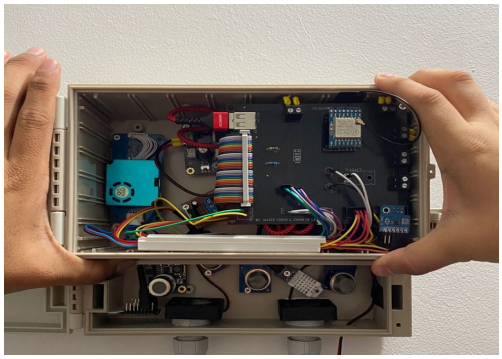


Figura 44: Resultado montaje estructura

Figura 45: Resultado montaje servidor

6. Conclusiones

Esta investigación tuvo como objetivo desarrollar un prototipo para el monitoreo de sustancias contaminantes y de la calidad del aire en el campus de la Universidad del Magdalena, con el fin generar alertas tempranas para enviar las posibles problemáticas que pudieran generarse a raíz de la contaminación del aire. Con base en los resultados obtenidos y evaluados se puede concluir que gracias a las sustancias medidas pertinentemente mediante la implementación de los diversos sensores usados, es posible determinar de manera correcta el estado de la calidad del aire dentro del campus de la Universidad del Magdalena.

Adicionalmente se puede concluir que se realizó de manera óptima el estudio y análisis de las sustancias que afectan la calidad del aire dentro del entorno objeto de estudio de esta investigación, con el fin de escoger las más importantes al ser sensadas y así mismo se realizó con éxito un prototipo capaz de adquirir almacenar y transmitir los datos recolectados por los distintos sensores.

De acuerdo a lo visto en los resultados se puede evidenciar, que luego del proceso de calibración de los sensores que lo requieren, los datos obtenidos son aceptables según los sensores de referencia y los parámetros establecidos por IDEAM.

Finalmente se realizó con éxito la interfaz de usuario encargada de visualizar de forma sencilla y gráfica, los datos recolectados por los sensores en el campus de la Universidad del Magdalena para que todos los usuarios puedan acceder a estos.

A. Anexo: Código desarrollado en Python para el cliente

```
#ejecutar en la ventana de comando "sudo pigpiod" antes de
ejecutar este código

import SDL_Pi_HM3301

import traceback

import pigpio

import math

import cmath

import time

import board

import busio

import adafruit_dht

import adafruit_ads1x15.ads1015 as ADS1

from adafruit_ads1x15.analog_in import AnalogIn as AI1

import adafruit_ads1x15.ads1115 as ADS

from adafruit_ads1x15.analog_in import AnalogIn as AI2

from datetime import datetime

import mysql.connector as mysql

from SX127x.LoRa import *

from SX127x.board_config import BOARD


# Configuración de entrada Temperatura y Humedad

dhtDevice = adafruit_dht.DHT22(board.D14,
use_pulseio=False)


# Activación de bus i2c

i2c = busio.I2C(board.SCL, board.SDA)
```

```
# R0 para NO2
R0_no2 = 2159900

# R0 para CO
R0_co = 100152
#R0_co = 1140152

# Configuración de Rl y R0 para metano
Rl_ch4 = 20
R0_ch4 = 10.07

# Configuración de Rl y R0 para dióxido de azufre
Rl_so2 = 20
R0_so2 = 0.50

# Configuración de Rl, R0 y los puntos de la recta para O3
Rl_o3  = 1000000
R0_o3  = 3150
p0_o3 = 0.69
p1_o3 = 0.69
p2_o3 = -0.76

# Configuración de los puntos de la recta para dióxido de
carbono
p0_co2 = 2.602
p1_co2 = 0.0658
```

```
p2_co2 = 0.030 / (2.602-3)


#Configuracion de pines para la comunicación con Lora
BOARD.setup()


#Función encargada de calcular la concentración en ppm de
NO2
def val_no2():
    # Inicio de librería toma de datos sensor AQ5
    ads1 = ADS1.ADS1015(i2c)
    ads1.gain = 1


    # Asignación entrada de señal del sensor AQ5 para NO2
    adc_no2 = AI1(ads1, ADS1.P0)


    Rno2 = (adc_no2.voltage * 15000) / (3.3 -
adc_no2.voltage)
    NO2 = pow(Rno2 / R0_no2, 1.007) / 6.855
    NO2 = "{0:.5f}".format(NO2)
    return(NO2)


#Función encargada de calcular la concentración en ppm de CO
def val_co():
    # Inicio de librería toma de datos sensor AQ5
    ads1 = ADS1.ADS1015(i2c)
    ads1.gain = 1
```

```
    # Asignación entrada de señal del sensor AQ5 para NO2
    adc_co = AI1(ads1, ADS1.P2)
    cont = 0
    while cont<=20:
        volt_CO = adc_co.voltage
        if (volt_CO > 0):
            Rco = (volt_CO * 1000000) / (3.3 - volt_CO)
            CO = pow((Rco / R0_co) , -1.179) * 4.385
            CO = "{0:.2f}".format(CO)
            cont == 21
            return(CO)


#Función encargada de calcular la concentración en ppm de
Metano
def val_ch4():
    # Inicio de librería de toda de datos sensores análogos
    ads2 = ADS.ADS1115(i2c)


    # Asignación entrada señal de sensor analogo CH4
    adc_ch4 = AI2(ads2, ADS.P0)


    cond = 0
    while cond<20:
        sensorValue_ch4 = round(adc_ch4.voltage, 5)
        if sensorValue_ch4 > 0:
            ppm_ch4 = (sensorValue_ch4 +
0.0148148)/(0.002074074)
```

```
            ppm_ch4 = "{0:.4f}".format(ppm_ch4)
            cond = 21
            return(ppm_ch4)


    #Función encargada de calcular la concentración en ppm de
    SO2
    def val_so2():
        # Inicio de librería de toda de datos sensores análogos
        ads2 = ADS.ADS1115(i2c)


        # Asignación entrada señal de sensor analogo SO2
        adc_so2 = AI2(ads2, ADS.P1)


        cond = 0
        while cond<20:
            sensorValue_so2 = adc_so2.voltage
            if sensorValue_so2 > 0:
                RS_so2 = ((5/sensorValue_so2) - 1)*Rl_so2
                RS_so2 = RS_so2/10
                ratio_so2 = RS_so2/R0_so2
                ppm_so2 = (pow(10,(
    ((math.log10(ratio_so2)-0.53)/-0.44) + 2.3)))
                ppm_so2 = "{0:.3f}".format(ppm_so2)
                ppm_so2 = float(ppm_so2)
                if ppm_so2 > 0:
                    cond = 21
                    ppm_so2 = str(ppm_so2)
```

```
                return(ppm_so2)


    #Función encargada de calcular la concentración en ppm de O3
    def val_o3():
        # Inicio de librería de toda de datos sensores análogos
        ads2 = ADS.ADS1115(i2c)

        # Asignación entrada señal de sensor analogo O3
        adc_o3 = AI2(ads2, ADS.P2)

        cond = 0
        while cond<20:
            sensorValue_o3 = adc_o3.voltage
            if sensorValue_o3 > 0:
                RS_o3 = ((5/sensorValue_o3) - 1)*Rl_o3
                ratio_o3 = RS_o3/R0_o3
                ppm_o3 = pow(10,
((math.log10(ratio_o3))-p1_o3)/p2_o3+p0_o3)
                ppm_o3 = "{0:.3f}".format(ppm_o3)
                cond = 21
                return(ppm_o3)

    #Función encargada de calcular la concentración en ppm de
    CO2
    def val_co2():
        # Inicio de librería de toda de datos sensores análogos
```

```
    ads2 = ADS.ADS1115(i2c)


    # Asignación entrada señal de sensor analogo CO2
    adc_co2 = AI2(ads2, ADS.P3)


    cond = 0
    while cond<20:
        cond += 1
        voltaje = adc_co2.voltage
        if voltaje > 0:
            ratio_co2 = voltaje / 8.5
            ppm_co2 = pow(10,
((ratio_co2)-p1_co2)/p2_co2+p0_co2)
            ppm_co2 = "{0:.3f}".format(ppm_co2)
            cond = 21
            return(ppm_co2)


#Función encargada de calcular la Temperatura y Humedad
ambiente
def val_TH():
    i = 1
    while i:
        try:
            temp = dhtDevice.temperature
            temp = "{}".format(temp)


            hume = dhtDevice.humidity
```

```
                hume = "{}".format(hume)
                if temp != 'None':
                    i = 0
                    return(temp, hume)
            except RuntimeError as error:
                continue
            except Exception as error:
                continue


#Función encargada de calcular el PM2.5 y PM10
def val_PM():
   #Función para la creación de un nuevo bus i2c a 20KHz
   mypi = pigpio.pi()


   #Asignación de pines del bus i2c
   mySDA = 20
   mySCL = 21
   #Se le pasan los valores de los pines por donde se
comunica el sensor HM3301
   hm3301 = SDL_Pi_HM3301.SDL_Pi_HM3301(SDA=mySDA,
SCL=mySCL, pi=mypi)
   time.sleep(0.01)
   try:
       cond = 0
       while cond<20:
           #se obtienen los datos del sensor
           PM = hm3301.get_data()
```

```
                if (hm3301.checksum() != True):
                    print("Checksum Error!")

                if (float(PM[0])==0 and float(PM[1])==0):
                    cond += 1
                else:
                    cond = 21
                    PM2_5 = PM[0]
                    PM10 = PM[1]
                    hm3301.close()
                    return(PM2_5, PM10)
                time.sleep(0.2)

        except:
            print ("closing hm3301")
            print(traceback.format_exc())
            hm3301.close()

    # calculo del indice de calidad del aire
    def val_ICA(NO2, CO, SO2, O3, PM):

        #Índice de calidad del aire para NO2
        NO2 = float(NO2)
        if (0.65 <= NO2 and 1.24 >= NO2):
            ICA_NO2 = 167.79661*NO2 + 91.93220
        elif (1.25 <= NO2 and 1.64 >= NO2):
            ICA_NO2 = 253.84615*NO2 - 16.30769
```

```
elif (1.65 <= NO2 and 2.04 >= NO2):
    ICA_NO2 = 253.84615*NO2 - 17.84615
else:
    ICA_NO2 = 0


#Índice de calidad del aire para CO
CO = float(CO)
if (0 <= CO and 4.4 >= CO):
    ICA_CO = 11.36363*CO
elif (4.5 <= CO and 9.4 >= CO):
    ICA_CO = 10*CO + 6
elif (9.5 <= CO and 12.4 >= CO):
    ICA_CO = 16.89655*CO - 59.51724
elif (12.5 <= CO and 15.4 >= CO):
    ICA_CO = 16.89655*CO - 60.20689
elif (15.5 <= CO and 30.4 >= CO):
    ICA_CO = 6.64429*CO + 98.01342
elif (30.5 <= CO and 40.4 >= CO):
    ICA_CO = 10*CO - 4
elif (40.5 <= CO and 50.4 >= CO):
    ICA_CO = 10*CO - 4


#Índice de calidad del aire para SO2
SO2 = float(SO2)
if (0 <= SO2 and 0.034 >= SO2):
    ICA_SO2 = 1470.58823*SO2
```

```
elif (0.035 <= SO2 and 0.144 >= SO2):
    ICA_SO2 = 449.54128*SO2 + 35.26605
elif (0.145 <= SO2 and 0.224 >= SO2):
    ICA_SO2 = 620.25316*SO2 + 11.06329
elif (0.225 <= SO2 and 0.304 >= SO2):
    ICA_SO2 = 620.25316*SO2 + 11.44303
elif (0.305 <= SO2 and 0.604 >= SO2):
    ICA_SO2 = 331.10367*SO2 + 100.01337
elif (0.605 <= SO2 and 0.804 >= SO2):
    ICA_SO2 = 497.48743*SO2 + 0.02010
elif (0.805 <= SO2 and 1.004 >= SO2):
    ICA_SO2 = 497.48743*SO2 + 0.52261

#Índice de calidad del aire para O3
O3 = float(O3)
if (0 <= O3 and 0.059 >= O3):
    ICA_O3 = 847.45762*O3
elif (0.060 <= O3 and 0.075 >= O3):
    ICA_O3 = 3266.66666*O3 - 145
elif (0.076 <= O3 and 0.095 >= O3):
    ICA_O3 = 2578.94736*O3 - 95
elif (0.096 <= O3 and 0.115 >= O3):
    ICA_O3 = 2578.94736*O3 - 96.57894
elif (0.116 <= O3 and 0.404 >= O3):
    ICA_O3 = 343.75*O3 + 161.125
elif (0.405 <= O3 and 0.504 >= O3):
    ICA_O3 = 1000*O3 - 104
```

```
elif (0.505 <= O3 and 0.604 >= O3):
    ICA_O3 = 1000*O3 - 104


#Índice de calidad del aire para PM2.5
PM2_5 = float(PM[0])
if (0 <= PM2_5 and 15.4 >= PM2_5):
    ICA_PM2_5 = (3.24675)*PM2_5
elif (15.5 <= PM2_5 and 40.4 >= PM2_5):
    ICA_PM2_5 = (1.96787)*PM2_5 + (20.49799)
elif (40.5 <= PM2_5 and 65.4 >= PM2_5):
    ICA_PM2_5 = (1.96787)*PM2_5 + (21.30120)
elif (65.5 <= PM2_5 and 150.4 >= PM2_5):
    ICA_PM2_5 = (0.57714)*PM2_5 + (113.1967)
elif (150.5 <= PM2_5 and 250.4 >= PM2_5):
    ICA_PM2_5 = (0.99099)*PM2_5 - (51.85585)
elif (250.5 <= PM2_5 and 350.4 >= PM2_5):
    ICA_PM2_5 = (0.99099)*PM2_5 - (52.75675)
elif (350.5 <= PM2_5 and 500.4 >= PM2_5):
    ICA_PM2_5 = (0.66044)*PM2_5 - (196.51567)


#Índice de calidad del aire para PM10
PM10 = float(PM[1])
if (0 <= PM10 and 54 >= PM10):
    ICA_PM10 = (25/27)*PM10
elif (55 <= PM10 and 154 >= PM10):
    ICA_PM10 = (49/99)*PM10 + (214/9)
elif (155 <= PM10 and 254 >= PM10):
```

```
        ICA_PM10 = (49/99)*PM10 + (2404/99)
    elif (255 <= PM10 and 354 >= PM10):
        ICA_PM10 = (49/99)*PM10 + (818/33)
    elif (355 <= PM10 and 424 >= PM10):
        ICA_PM10 = (33/23)*PM10 - (7092/23)
    elif (425 <= PM10 and 504 >= PM10):
        ICA_PM10 = (99/79)*PM10 - (18296/79)
    elif (505 <= PM10 and 604 >= PM10):
        ICA_PM10 = PM10 - 104


    #Selección del índice de calidad del aire ICA
    ica = 0
    if (ICA_PM2_5 > ica):
        ica = ICA_PM2_5
    if (ICA_PM10 > ica):
        ica = ICA_PM10
    if (ICA_CO > ica):
        ica = ICA_CO
    if (ICA_NO2 > ica):
        ica = ICA_NO2
    if (ICA_O3 > ica):
        ica = ICA_O3
    if (ICA_SO2 > ica):
        ica = ICA_SO2
    ica = "{:.1f}".format(ica)
    return(ica)
```

```
#Función encargada de arrojar la fecha y la hora actual

def val_FH():

   F_H = datetime.now()

   format = F_H.strftime('%Y-%m-%d')

   fecha = format

   format = F_H.strftime('%H:%M:%S')

   hora = format

   return(fecha, hora)


#Función encargada de enviar la información a la base de
datos

def
enviar_BD(fecha,hora,PM2_5,PM10,SO2,NO2,CO2,CO,O3,CH4,Temp,
Hume,ICA):


   #Iniciamos la conexión a la base de datos

   db = mysql.connect(host="127.0.0.1", user="calidadaire",
passwd="64548606", database="calidadaire")


   #Creamos un cursor para la base de datos

   cursor = db.cursor()


   #Generamos el query que será enviado a la base de datos
MySQL

   query = "INSERT INTO
mediciones(fecha,hora,pm2_5,pm10,so2,no2,co2,co,ozono,metan
o,temperatura,humedad,ica) VALUES
(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"
```

```
    #Asignamos la información a enviar de acuerdo al orden
especifico
    valores =
(fecha,hora,PM2_5,PM10,SO2,NO2,CO2,CO,O3,CH4,Temp,Hume,ICA)


    #Se envía el query mediante el cursor
    cursor.execute(query,valores)
    db.commit()


    #Cerramos el cursor y la conexión a la base de datos
    cursor.close()
    db.close()


class mylora(LoRa):
    def __init__(self, verbose=False):
        super(mylora, self).__init__(verbose)
        self.set_mode(MODE.SLEEP)
        self.set_dio_mapping([0] * 6)


    def on_rx_done(self):
        self.clear_irq_flags(RxDone=1)
        payload = self.read_payload(nocheck=True )
        mens=bytes(payload).decode("utf-8",'ignore')
        mens=mens[2:-1]
        print(mens)
        if mens=="calidadaire":
            time.sleep(2)
```

```
                self.write_payload(valor)
                self.set_mode(MODE.TX)

            time.sleep(8)

    def start(self):
        self.reset_ptr_rx()
        self.set_mode(MODE.RXCONT)
        start_time = time.time()
        while (time.time() - start_time < 5):
            pass;

def
codi(fecha,hora,PM2_5,PM10,SO2,NO2,CO2,CO,O3,CH4,Temp,Hume,
ICA):
    codificado = [255, 255, 0, 0]
    for letra in fecha:
        codificado.append(ord(letra))
    codificado.append(64)
    for letra in hora:
        codificado.append(ord(letra))
    codificado.append(64)
    for letra in str(PM2_5):
        codificado.append(ord(letra))
    codificado.append(64)
    for letra in str(PM10):
        codificado.append(ord(letra))
```

```
        codificado.append(64)
        for letra in SO2:
            codificado.append(ord(letra))
        codificado.append(64)
        for letra in NO2:
            codificado.append(ord(letra))
        codificado.append(64)
        for letra in CO2:
            codificado.append(ord(letra))
        codificado.append(64)
        for letra in CO:
            codificado.append(ord(letra))
        codificado.append(64)
        for letra in O3:
            codificado.append(ord(letra))
        codificado.append(64)
        for letra in CH4:
            codificado.append(ord(letra))
        codificado.append(64)
        for letra in Temp:
            codificado.append(ord(letra))
        codificado.append(64)
        for letra in Hume:
            codificado.append(ord(letra))
        codificado.append(64)
        for letra in ICA:
            codificado.append(ord(letra))
```

```
    codificado.append(0)

    return(codificado)


#Función principal

def main():

    #Pasamos los datos de configuración para el módulo Lora
RA02

    lora = mylora(verbose=False)

    lora.set_freq(433.0)

    lora.set_pa_config(pa_select=1, max_power=21,
output_power=18)

    lora.set_bw(BW.BW125)

    lora.set_coding_rate(CODING_RATE.CR4_5)

    lora.set_spreading_factor(12)

    lora.set_rx_crc(True)

    lora.set_low_data_rate_optim(True)

    assert(lora.get_agc_auto_on() == 1)


    while 1:

        NO2 = val_no2()

        CO = val_co()

        CH4 = val_ch4()

        SO2 = val_so2()

        O3 = val_o3()

        CO2 = val_co2()

        TH = val_TH()

        PM = val_PM()
```

```
        ICA = val_ICA(NO2, CO, SO2, O3, PM)

        FH = val_FH()


print(FH[0],FH[1],PM[0],PM[1],SO2,NO2,CO2,CO,O3,CH4,TH[0],T
H[1],ICA)


enviar_BD(FH[0],FH[1],PM[0],PM[1],SO2,NO2,CO2,CO,O3,CH4,TH[
0],TH[1],ICA)

        global valor

        valor =
codi(FH[0],FH[1],PM[0],PM[1],SO2,NO2,CO2,CO,O3,CH4,TH[0],TH
[1],ICA)

        print("Esperando servidor...")

        lora.start()

        time.sleep(55.0)


if __name__ == '__main__':

   main()
```

B. Anexo: Código de librería en Python para uso de HM3301 a 20Khz

```
import time

import pigpio


SDA = 20

SCL = 21

DATA_CNT = 29


class SDL_Pi_HM3301(object):
```

```
    def __init__(self, SDA=20, SCL=21, I2C_Address =0x40,
pi= None):
        self.pi = pi
        self.SDA = SDA
        self.SCL = SCL
        self.I2C_Address = I2C_Address
        self.last_data = None
        self.PM_2_5_conctrt_atmosph = 0      #PM2.5
Concentración del ambiente atmosférico ,unidad:ug/m3
        self.PM_10_conctrt_atmosph = 0       #PM10
Concentración del ambiente atmosférico ,unidad:ug/m3
        self.pi.set_pull_up_down(self.SDA, pigpio.PUD_UP)
        self.pi.set_pull_up_down(self.SCL, pigpio.PUD_UP)
        h = self.pi.bb_i2c_open(self.SDA, self.SCL, 20000)

        (count, data) = self.pi.bb_i2c_zip(
             self.SDA, [4, self.I2C_Address, 2,7,1,
0x80,2,7,1,0x88,3,0])
        time.sleep(10.0/1000.0)

    def read_HM3301_data(self):
        (count, data) = self.pi.bb_i2c_zip(
             self.SDA, [4,self.I2C_Address,2,7,1,0x81,3,
2,6,DATA_CNT,3,0   ])
        return list(data)

    def close(self):
        self.pi.bb_i2c_close(self.SDA)
        self.pi.stop()
```

```
    def checksum(self):
        sum = 0
        for i in range(DATA_CNT-1):
            sum += self.last_data[i]
        sum = sum & 0xff
        return (sum==self.last_data[28])

    def parse_data(self, data):
        self.PM_2_5_conctrt_atmosph = data[12]<<8 | data[13]
        self.PM_10_conctrt_atmosph  = data[14]<<8 | data[15]

    def get_data(self):
        data = self.read_HM3301_data()
        self.last_data = data
        self.parse_data(data)
        return list( (self.PM_2_5_conctrt_atmosph,
self.PM_10_conctrt_atmosph))
```

C. Anexo: Código desarrollado en Python para el servidor

```
#!/usr/bin/env python3
import time
from SX127x.LoRa import *
from SX127x.board_config import BOARD
import mysql.connector as mysql

BOARD.setup()
class mylora(LoRa):
```

```
    def __init__(self, verbose=False):
        super(mylora, self).__init__(verbose)
        self.set_mode(MODE.SLEEP)
        self.set_dio_mapping([0] * 6)

    def on_rx_done(self):
        self.clear_irq_flags(RxDone=1)
        payload = self.read_payload(nocheck=True)
        dato_c = bytes(payload).decode("utf-8",'ignore')
        dato_c = dato_c[2:-1]
        valor = dato_c.split("@")
        #Iniciamos la conexión a la base de datos
        db = mysql.connect(host="127.0.0.1",
user="calidadaire", passwd="64548606",
database="calidadaire")

        #Creamos un cursor para la base de datos
        cursor = db.cursor()

        #Generamos el query que será enviado a la base de
datos MySQL
        query = "INSERT INTO
mediciones(fecha,hora,pm2_5,pm10,so2,no2,co2,co,ozono,metan
o,temperatura,humedad,ica) VALUES
(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"

        #Asignamos la información a enviar de acuerdo al
orden especifico
        valores =
(valor[0],valor[1],valor[2],valor[3],valor[4],valor[5],valo
```

```
r[6],valor[7],valor[8],valor[9],valor[10],valor[11],valor[12])


        #Se envía el query mediante el cursor
        cursor.execute(query,valores)
        db.commit()


        #Cerramos el cursor y la conexión a la base de datos
        cursor.close()
        db.close()
        time.sleep(2) # Espera a que el cliente esté listo


    def start(self):
        self.write_payload([255, 255, 0, 0, 99, 97, 108, 105, 100, 97, 100, 97, 105, 114, 101, 0]) #pedimos los datos
        self.set_mode(MODE.TX)
        time.sleep(3)
        self.reset_ptr_rx()
        self.set_mode(MODE.RXCONT)
        start_time = time.time()
        while (time.time() - start_time < 12):
            pass;
        self.reset_ptr_rx()
        self.set_mode(MODE.RXCONT)


lora = mylora(verbose=False)
lora.set_freq(433.0)
```

```
lora.set_pa_config(pa_select=1, max_power=21,
output_power=18)

lora.set_bw(BW.BW125)

lora.set_coding_rate(CODING_RATE.CR4_5)

lora.set_spreading_factor(12)

lora.set_rx_crc(True)

lora.set_low_data_rate_optim(True)

assert(lora.get_agc_auto_on() == 1)


while True:

   print("Iniciando servidor...")

   lora.start()
```

Bibliografía

CALIDAD DEL AIRE - IDEAM. (n.d.). Retrieved October 10, 2021, from http://www.ideam.gov.co/web/contaminacion-y-calidad-ambiental/calidad-del-aire

Benavides, H. O., & León, G. E. (2007). Información técnica sobre Gases de Efecto Invernadero y el cambio climático. Ideam, 1–102. https://doi.org/IDEAM–METEO/008-2007

Betancourt-Portela, J. M., Parra, J. P., & Villamil, C. (2013). Emisión de metano y óxido nitroso de los sedimentos de manglar de la ciénaga grande de santa marta, Caribe Colombiano. Boletín de Investigaciones Marinas y Costeras.

Alvarez-Miño, L., Taboada-Montoya, R., Trujillo-Montes, A. C., & Salazar-Ceballos, A. (2016). Huella de carbono en Santa Marta - Colombia y factores asociados. Análisis desde los determinantes sociales de la salud. 2014. Universidad y Salud. https://doi.org/10.22267/rus.161802.42

Aire. (n.d.). Retrieved December 4, 2019, from https://www.corpamag.gov.co/index.php/es/informacion-ambiental/aire

Ministerio de Ambiente y Desarrollo Sostenible. (2017). Resolución 2254 - Por la cual de adopta la norma de calidad del aire ambiente y se dictan otras disposiciones (p. 11). p. 11. Retrieved from http://www.ideam.gov.co/documents/51310/527391/2.+Resolución+2254+de+2017+-+Niveles+Calidad+del+Aire..pdf/c22a285e-058e-42b6-aa88-2745fafad39f%0Ahttp://www.minambiente.gov.co/images/normativa/app/resoluciones/96-res 2254 de 2017.pdf

Ministerio de Ambiente Vivienda y Desarrollo Territorial. (2010). Manual de diseño de sistemas de vigilancia de la calidad del aire. Protocolo Para El Monitoreo y Seguimiento de La Calidad Del Aire, 137.

Rojo, R. R. (2014) Seguridad y medio ambiente en planta química. QUIE0108. IC Editorial.

En funcionamiento segunda estación de monitoreo de calidad del aire. (n.d.). Retrieved December 4, 2019, from http://www.contrastes.com.co/noticiasxxxx/index.php/ar/soledad/actualidad/4342-en-funcionamiento-segunda-estacion-de-monitoreo-de-calidad-del-aire

Muñoz, F., & Carvalho, M. S. (2009). Efecto del tiempo de exposición a PM10 en las urgencias por bronquitis aguda. Cadernos de Saúde Pública. https://doi.org/10.1590/s0102-311x2009000300008

Gaviria, C. F., Benavides, P. C., & Tangarife, C. A. (2008). Contaminación por material particulado (pm 2,5 y pm10) y consultas por enfermedades respiratorias en Medellín. Revista Facultad Nacional de Salud Pública, 29. Retrieved from http://www.scielo.org.co/pdf/rfnsp/v29n3/v29n3a04.pdf

Sánchez, J., Romieu, I., Ruiz, S., Pino, P., & Gutiérrez, M. (1999). Efectos agudos de las partículas respirables y del dióxido de azufre sobre la salud respiratoria en niños del área

industrial de Puchuncaví, Chile. Revista Panamericana de Salud Pública, 6(6), 384–391. https://doi.org/10.1590/s1020-49891999001100003

Gascon, M., & Sunyer, J. (2015). Contaminación del aire y salud respiratoria en niños. Archivos de Bronconeumología, 51(8), 371–372. https://doi.org/10.1016/j.arbres.2015.03.001

Cabrera, S., Lissi, E., Honeyman, J. (2005). Radiación ultravioleta y salud. Editorial Universitaria.

Palacios, E., & Espinoza, C. (2014). Contaminación del aire exterior Cuenca - Ecuador, 2009 - 2013. Posibles efectos en la salud. Revista de La Facultad de Ciencias Médicas, 32(2), 6–17.

Téllez, J., Rodríguez, A., & Fajardo, álvaro. (2006). Contaminación por monóxido de carbono: un problema de salud ambiental. Revista de Salud Pública. https://doi.org/10.1590/s0124-00642006000100010

Luque, J. C. (2016). El gas metano y su relación con las actividades ganaderas de la provincia de Manabi, Ecuador Methane and its relationship to livestock activities in the province of Manabí, Ecuador (Vol. 19).

Kane, J. W., Sternheim, M. M., Vázquez, J. C., & Mirabent, D. J. (1989). Física. Editorial Reverte. https://books.google.com.co/books?id=lj5kLw2uxGIC

Gliessman-Stephen, R. (2002). Agroecología: Procesos ecológicos en agricultura sostenible. Diversidad y Estabilidad Del Agroecosistema, 229.

Hernández A. M. (2013). Hoja metodológica del indicador Índice de Calidad del Aire - ICA (Versión 1,00).Sistema de Indicadores Ambientales de Colombia. Colombia: Instituto de Hidrología, Meteorología y Estudios Ambientales. 13p

Doole, G (2020). Raspberry Pi 4 Pinout, Features and Peripherals [Figura]. Recuperado de: https://microcontrollerslab.com

Seeed wiki (Sin fecha). Grove - Laser PM2.5 Sensor (HM3301) [Figura]. Recuperado de: https://wiki.seeedstudio.com

Didácticas Electrónicas (Sin fecha). Sensor temp. y hum. AM2302/DHT22 con cable [Figura]. Recuperado de: https://www.didacticaselectronicas.com

Didácticas Electrónicas (Sin fecha). Air Quality 5 Click [Figura]. Recuperado de: https://www.didacticaselectronicas.com

DFROBOT (Sin fecha). Gravity: Analog CO2 Gas Sensor For Arduino (MG-811 Sensor) [Figura]. Recuperado de: https://www.dfrobot.com

SERVOTRONIK (Sin fecha). Sensor MQ-4 metano y gas natural [Figura]. Recuperado de: https://www.servotronik.com.co

Didácticas Electrónicas (Sin fecha). Sensor de Ozono MQ131 - V2 [Figura]. Recuperado de: https://www.didacticaselectronicas.com

Didácticas Electrónicas (Sin fecha).Sensor Gas Hidrógeno MQ136 [Figura]. Recuperado de: https://www.didacticaselectronicas.com

Didácticas Electrónicas (Sin fecha). ADC Raspberry, ADS1115 [Figura]. Recuperado de: https://www.didacticaselectronicas.com

Didácticas Electrónicas (Sin fecha). Tarjeta Wireless LORA-02 433MHz [Figura]. Recuperado de: https://www.didacticaselectronicas.com

Monsolar (2021). ¿Qué es y qué hace un regulador de carga solar? [Figura]. Recuperado de: https://www.monsolar.com

Villalta - DISEÑO DEL SISTEMA DE ALIMENTACIÓN BASADO EN ENERGÍA SOLAR DE UNA ESTACIÓN DE BOMBEO C... (n.d.).

¿Qué es una batería solar? Tipos y funcionamiento. (2015). Retrieved October 11, 2021, from https://solar-energia.net/energia-solar-fotovoltaica/elementos/baterias-solares

Simón Bordón, M. (2010). *Estudio y análisis de un regulador de carga de baterías*. https://e-archivo.uc3m.es/handle/10016/10870

capterra (2021). ¿Qué es Raspberry Pi OS? [Figura]. Recuperado de: https://www.capterra.co

Predko, Michael. (2001). *Programming and customizing PICmicro microcontrollers*. 1190.

howtomechatronics (2016). DHT11 & DHT22 Sensors Temperature and Humidity Tutorial using Arduino [Figura]. Recuperado de: https://howtomechatronics.com

SENSORTECH (En línea). Datasheet MICS-6814 [Figura]. Recuperado de: https://www.sgxsensortech.com

SENSORTECH (En línea). Datasheet MICS-6814 [Figura]. Recuperado de: https://www.sgxsensortech.com

naylamp mechatronics (Sin fecha). TUTORIAL SENSORES DE GAS MQ2, MQ3, MQ7 Y MQ135 [Figura]. Recuperado de: https://naylampmechatronics.com/

Hanwei Electronics (En línea). Datasheet MG-811 sensor [Figura]. Recuperado de: https://html.alldatasheet.es

Hanwei Electronics (En línea). Datasheet MQ-4 sensor [Figura]. Recuperado de: https://html.alldatasheet.es

Zhengzhou Winsen Electronics Technology Co (En línea). Datasheet MQ-131 sensor [Figura]. Recuperado de: https://aqicn.org

Hanwei Electronics (En línea). Datasheet MQ-136 sensor [Figura]. Recuperado de: http://www.sensorica.ru

pip documentation v21.3. (n.d.). Retrieved October 17, 2021, from https://pip.pypa.io/en/stable/

Hernández A (2013).Hoja metodológica del indicador índice de calidad del aire - ICA (versión 1,00) [Figura]. Recuperado de: http://www.ideam.gov.co