

Sensor de Calidad de aire para invernaderos con Arduino

Autor: Nolan Pérez González

Objetivos y motivación:

Conocer los niveles de concentración de partículas suspendidas en el aire de forma general y de Dióxido de Carbono de forma específica es información muy útil al cultivar plantas en invernaderos, se estima que las plantas necesitan una concentración de CO₂ en aire de entre 800-1200ppm y superar los 1200ppm puede causar que el cierre de los estomas en la planta, lo que conlleva a una reducción de la capacidad de realizar la fotosíntesis y problemas de fitotoxicidad. Asimismo, la concentración de partículas suspendidas en el aire pueden brindar información sobre el nivel de polución que este tiene.

Ser capaz de medir estos dos indicadores y posteriormente analizar los datos es de gran utilidad en la toma de decisiones al ventilar un invernadero, por eso se ha decidido realizar un proyecto con Arduino para obtener estos datos, se busca que el proyecto es escalable, simple y relativamente barato de fabricar.

Aunque el objetivo claro es su uso para agricultura, estos módulos también se pueden utilizar para otros ambientes como lugares cerrados en los la cantidad de personas que puede haber dentro varía drásticamente, como un cine o un teatro, con el fin de regular, principalmente, el CO₂ generado.

Técnicas y materiales utilizados:

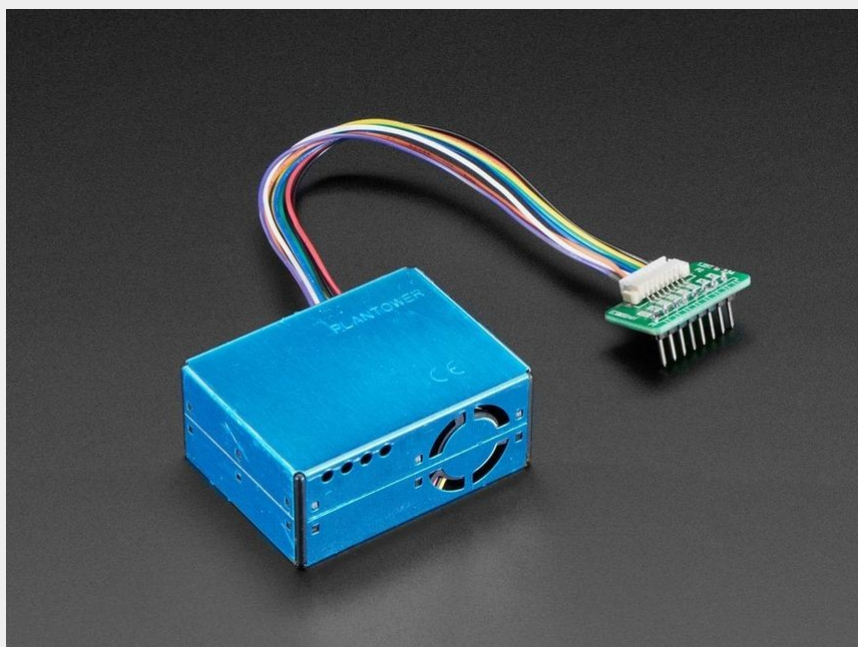
Componentes elegidos para el montaje:

Arduino ESP32-WROOM-32D (Microcontrolador compatible con Arduino IDE para su programación)

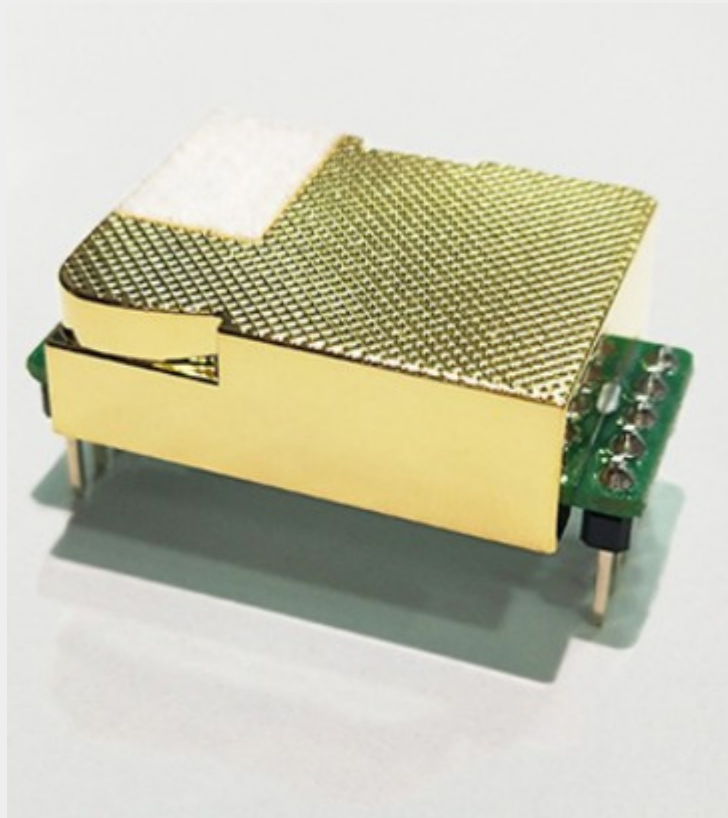
Pinout



Plantower PMS5003 (sensor de alta precisión de concentración de partículas suspendidas en aire)



MH-Z19C (sensor NDIR para contabilizar la concentración de Dióxido de Carbono en aire)



DS1302 (reloj a tiempo real alimentado por una batería de botón)



Software Utilizado:

Arduino IDE v2 (Ambiente de desarrollo y programación de microcontroladores Arduino)

Librerías para Arduino utilizadas:

MH-Z19 (Jonathan Dempsey)

PMS Library (Mariusz Kacki)

RTCLib (NeironX)

Detalle del proyecto:

El dispositivo toma datos de las partículas en suspensión que hay en el aire, separándolas en tres categorías PM1, PM2.5 y PM10 las cuales son clasificadas de esta forma en correspondencia a su tamaño PM1 (1 micrón), PM2.5 (2.5 micrones), PM10 (10 micrones). Resultan un gran aporte de información ya que se puede determinar hasta cierto punto, que tipo específico de partícula hay en el aire, y basado en esto utilizar sensores más avanzados si se perciben datos preocupantes, como ejemplo de partículas que pertenecen a cada categoría:

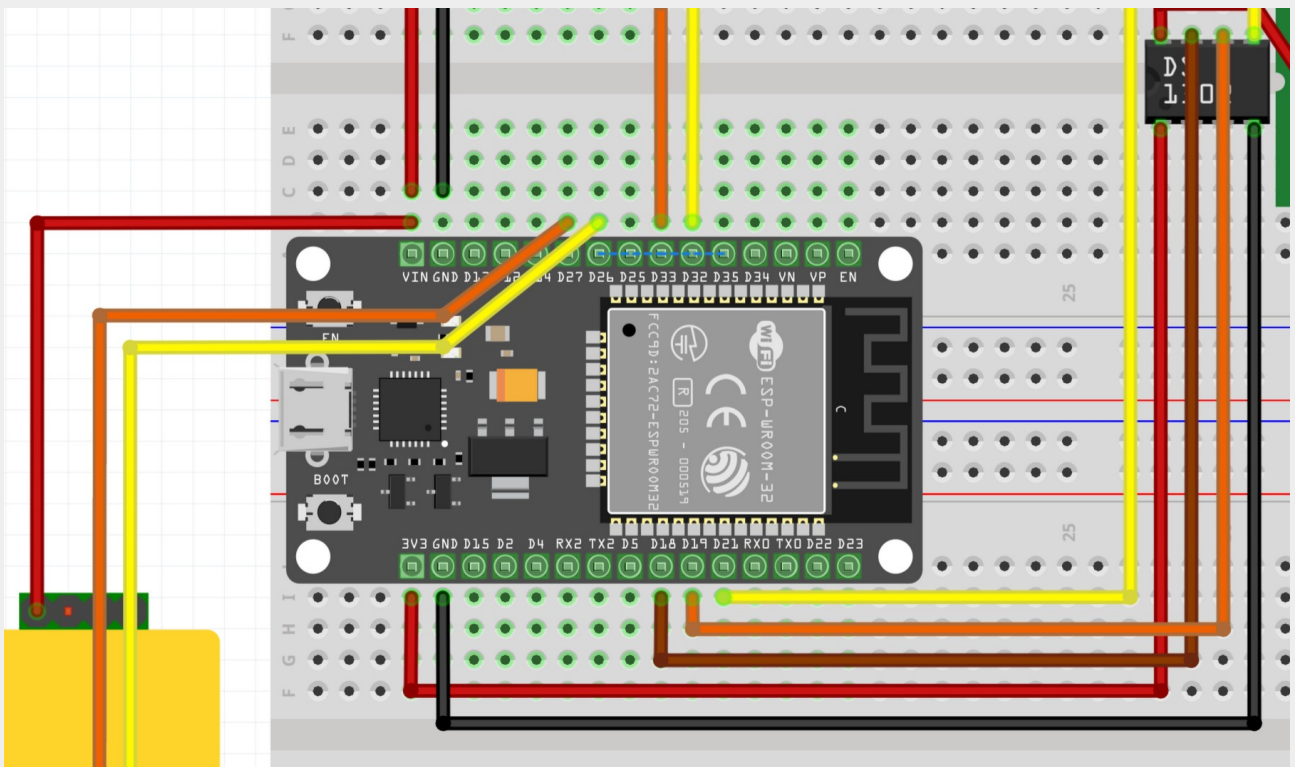
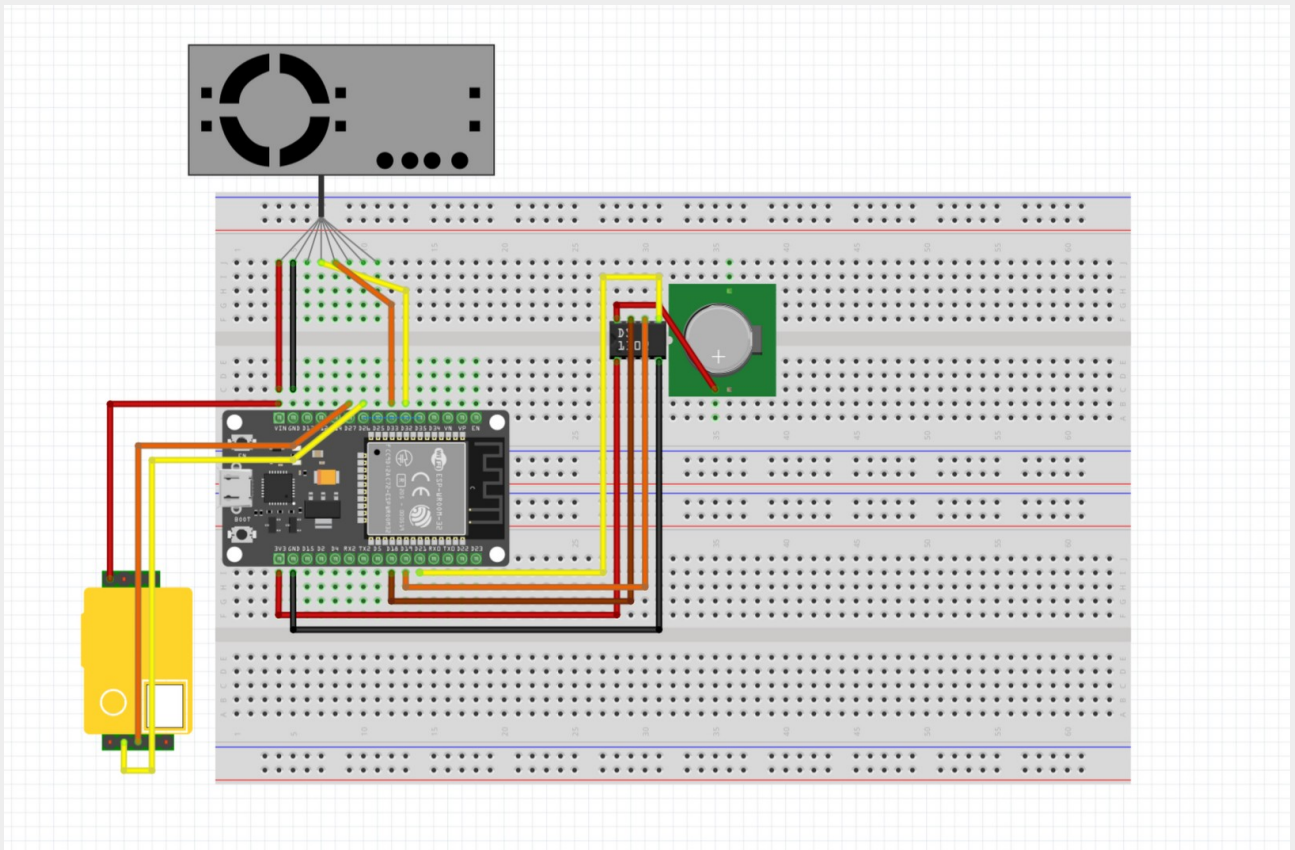
PM1: Virus, humo.

PM2.5: Polvo fino, niebla tóxica, bacterias, polen.

PM10: Esporas, polvo.

También se toman datos del CO2 en el aire (en partes por millón o ppm).

Las conexiones entre los dispositivos son como se muestran en las siguientes imágenes:



A nivel de lógica, los datos se recogen en una struct para luego transmitirla hacia una cliente de forma inalámbrica (a través de Bluetooth con BLE o “Bluetooth Low Energy”), el tamaño del paquete de [data](https://bleak.readthedocs.io/en/latest/os) <https://bleak.readthedocs.io/en/latest/os> es de 14 Bytes y contiene la hora obtenida del reloj DS1302 RTC, los datos de las tres clases de partículas en suspensión, la concentración de CO2 y la temperatura.

Estos datos se pueden manipular luego en el cliente para realizar otros análisis más complejos, contamos con un solo dispositivo en el momento pero se podría escalar a varios, obtener una gran cantidad de datos y automatizar su depuración en un lenguaje de programación externo con ayuda de bases de datos.

Aunque en el caso que nos compete, solo se conecta a un portátil con un script listener en Python que obtiene la struct enviada por el Arduino, la transforma en un Ordered_Dict para luego mostrarlo por consola, además, guarda todos los datos en un archivo .csv que almacena en el directorio en el que se encuentra el script y crea uno nuevo cada día para mantener una estadística.

Posibles Expansiones:

Como ya se mencionó antes, se pueden fabricar en masa varios módulos de este tipo y conectarlos a un ordenador central que tome los datos definidos estándares de la struct planteada para realizar análisis más complejos.

Se le podría añadir de forma sencilla un bot de Telegram que envíe los .csv al final del día, o haga una media en el tiempo y lo envíe cada períodos de tiempo más grandes, para automatizar la recepción de información y poder acceder desde un móvil.

Aún el microcontrolador tiene pines libres para agregar más dispositivos así que se podría añadir un sensor de humedad, hay ciertos tipos de plantas que no se cultivan bien en ambientes secos, o que no se pueden cultivar en ambientes muy húmedos.

También sería interesante explorar la posibilidad de agregarle un sensor de

radiación UV, ya que hay especies de plantas que no pueden sobrevivir en zonas con mucha de esta radiación y hay que utilizar mallas con filtro UV o películas de invernadero estabilizadas para que se puedan cultivar en dichas zonas.

Fuentes y materiales utilizados:

Archivo fuente .ino para el circuito: <https://github.com/NolanPG/air-quality-arduino-project/blob/main/ESP32-air-quality.ino>

Esquema hecho con [Fritzing](#):

<https://github.com/NolanPG/air-quality-arduino-project/blob/main/Schematics.tar.gz>

Script en Python para ordenador externo: https://github.com/NolanPG/air-quality-arduino-project/blob/main/arduino_helper.py

<https://envira.es/que-son-particulas-contaminantes-pm/>

<https://envira.es/exceso-de-co2-en-las-plantas-efectos/>

<https://vgd-led.com/es/Plantas-con-efectos-de-radiaci%C3%B3n-ultravioleta./>

<https://pypi.org/project/aiofiles/>

<https://docs.python.org/es/3/library/asyncio.html>

<https://bleak.readthedocs.io/en/latest/>

Enlaces a especificaciones de los dispositivos utilizados (se encuentran embebidos en la sección de componentes para el montaje)

Reflexiones y Notas Adicionales:

Conectar los RX y TX de los módulos con pines que estén definidos como TX y RX (RX con TX, TX con RX, Receptor con Transmisor, Transmisor con Receptor), importante que no tenga restricciones el pin a utilizar.