MANUAL DE CONSTRUCCIÓN
MONITOR CO2

C. Leonardo Velázquez
Héctor D. Cortés y
J. Antonio del Río

IER
Instituto de Energías Renovables

Manual de construcción Monitor CO_2

Universidad Nacional Autónoma de México
Instituto de Energías Renovables

Diseño de Portada

Jesús Antonio del Río Portilla usando Canva http://canva.com

ISBN 978-607-30-9110-7

DOI pendiente

Publicado en formato digital en Temixco, Morelos, México, 2024.

Manual de construcción Monitor CO_2

2022

Índice general

Capítulo 1

Introducción

Este manual responde a la necesidad de aportar datos cotidianos para atender dos problemas globales que nos están afectando y que requieren información del entorno cercano instantáneamente para definir nuestras acciones: la enfermedad de COVID-19 y el Cambio climático. El primero de ellos requiere monitorizar la calidad del aire en espacios cerrados con ventilación limitada para que con esta información definamos: permanecer en el lugar o ventilarlo o salir de él. El monitoreo de la calidad del aire en espacios cerrados es una de las tareas que nos ha dejado la pandemia del COVID-19 de inicios de la década del 2020. Para atender el segundo problema asociado a las necesidades energética de tener edificaciones confortables, es necesario definir acciones para contender con el cambio climático y usar energía de manera eficiente. Para ello, es importante conocer las condiciones de temperatura y humedad dentro de nuestras edificaciones y con ellas definir las estrategias encaminadas a hacer un uso eficiente de la energía y así disminuir el uso de combustibles fósiles, para disminuir la emisión de gases de efecto invernadero.

En el contexto del COVID-19, una de las enseñanzas de los estudios realizados para entender la propagación de la enfermedad indica que el contagio es fundamentalmente por vía aérea. Los virus que se alojan en las pequeñas gotas (gotículas) de las exhalaciones de una persona contagiada son la principal vía de transmisión al ser aspirados por una persona sana. Por esa razón, la inmensa mayoría de los contagios se dan en ambientes cerrados donde la concentración de esas gotículas es alta. Dado que puede haber personas contagiadas que todavía no presenten síntomas o personas portadoras del virus asintomáticas es de vital importancia medir la cantidad de aire que ha sido exhalado por alguna persona posiblemente contagiada en los lugares

cerrados. Uno de los productos de nuestra respiración es el CO_2, así que la proporción de gas de CO_2 en el aire en una habitación donde haya personas es una medida indirecta de la cantidad de aire que ha sido exhalado por estas personas. De similar manera, podemos decir que la concentración de CO_2 es un indicador indirecto de la cantidad de gotículas que puede haber en esa habitación producto de la respiración de las personas.

Por otro lado, la alta densidad de energía que presentan los combustibles fósiles hizo pensar a la humanidad que se podría utilizar la energía de manera indiscriminada. Este hecho condujo a construir edificaciones que requirieren un uso irracional de energía y, dado que en la actualidad la energía proviene fundamentalmente de los combustibles fósiles, se ha cambiado la composición de la atmósfera conduciendo a un cambio climático antropogénico. Por esta razón, tenemos la necesidad de modificar las edificaciones para que usen eficientemente la energía disminuyendo su uso y con ello atender la emergencia climática. El confort térmico de las personas que usan y habitan un edificio está determinado, entre otros parámetros, por los rangos de temperatura y de humedad que encontramos en él. Así una de las primeras tareas para reacondicionar edificaciones es medir la temperatura y la humedad en ellas lo largo del día, las semanas y los años. Al comparar esta con información con las cartas bioclimáticas de confort de cada región podemos definir las estrategias de reacondicionamiento de las edificaciones o promover cambio en el uso de a energía y comportamiento de las personas.

En la actualidad existen sensores de CO_2, de temperatura y de humedad disponibles en el mercado para ser conectados a dispositivos electrónicos que a su vez pueden ser programados para convertirse en monitores de CO_2 [1], de temperatura y de humedad. En particular, existen dispositivos electrónicos que pueden ser controlados con software y hardware abiertos que facilitan su incorporación a diferentes esquemas de uso sin restricción por propiedad industrial o derechos de autor.

El propósito de este manual es proporcionar las instrucciones necesarias para armar un dispositivo con capacidad de registrar la concentración de CO_2, la temperatura y la humedad de un recinto.

Este armado será de fácil composición, con elementos electrónicos y computacionales abiertos y disponibles a la mayoría de las personas y cuya preparación consista de una instrucción media en electrónica y computación. El dispositivo final se muestra en la Fig. 1.1. En la carátula del monitor se muestran el valor de la concentración de CO_2, hora del día, voltaje de la pila, temperatura y humedad relativa. Así como el indicador de la intensidad de la señal WiFI de red a la que está conectado. El registro de la información puede hacerse con intervalos de 3 minutos mostrados en pantalla, cabe resal-

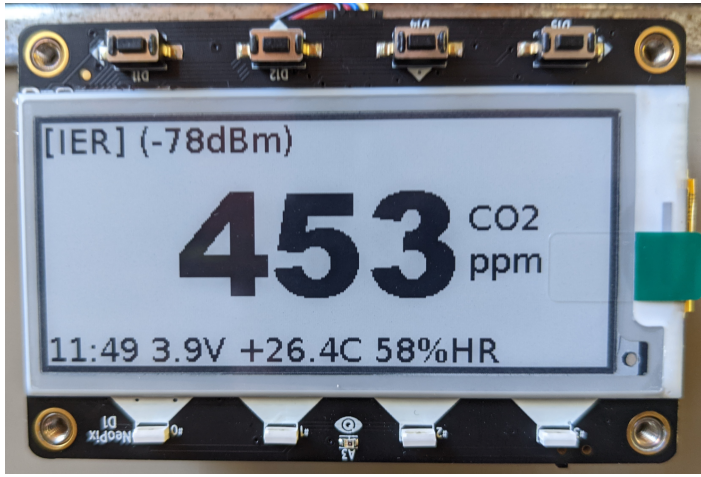


Figura 1.1: Fotografía de un monitor tipo. Se muestran el valor de la concentración de CO_2, hora del día, voltaje de la pila, temperatura, humedad relativa. Así como el indicador del nombre y la intensidad de la señal WiFI de red a la que está conectado.

tar que el sensor por omisión realiza una medición cada 2 s la que envía a la nube y puede ser almacenada en la nube misma o en alguna computadora.

La organización del presente manual es la siguiente: en el capítulo 2 se listan los materiales y se describe brevemente su función. En el capítulo 3, se describe la forma de armar este dispositivo. Es importante mencionar que se han seleccionado las partes para evitar la necesidad de soldar los componentes electrónicos. En el capítulo 4 se describe la forma de configurar el dispositivo y de programarlo. Finalmente en el apéndice se presentan los códigos de los programas. Aunque se anexa la liga para su acceso libre en *GitHub*.

Capítulo 2

Materiales

En este capítulo se presentan los componentes que se utilizarán y se dará una breve explicación sobre ellos y sus características.

- Dos tipos de cinta adhesiva doble cara, una transparente 2.1(a).
- Adafruit *MagTag* Starter Kit - ADABOX017 Essentials
- Adafruit SCD-30 - NDIR CO2 Temperature and Humidity Sensor - STEMMA QT / Qwiic
- STEMMA QT / Qwiic JST SH 4-pin Cable - 100mm Long

Se recomienda ocupar 2 tipos de cinta doble cara, como la que se muestra en la fotografía. Una de estas cintas servirá para unir el monitor con uno de los acrílicos; mientras la cinta trasparente se utilizará para unir la batería a la *MagTag*.

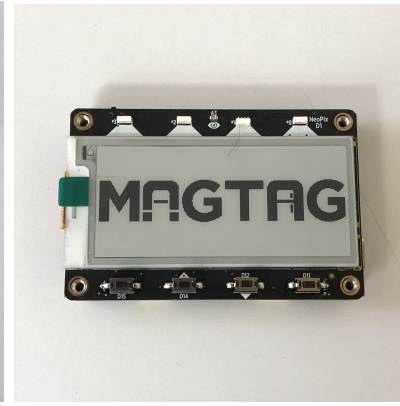
***MagTag*:** *Mag* es la abreviatura para *magnético*, en el paquete de esenciales se incluye: tres acrílicos, una batería de 3.7 V/420mAh, 4 mini pies magnéticos y la *MagTag* la cual consta de una pantalla de tinta y un módulo inalámbrico ESP32-S2, constituyendo una pantalla WiFI de muy bajo consumo.

Cable JST: Un cable simétrico de poco más de 100mm, equipado con puertos JST-SH hembra en ambos extremos, por lo que no hay que preocuparse de qué lado irá al puerto de monitor y cuál al puerto *MagTag*.

Sensor SCD-30-NDIR de CO_2: La tecnología que compone este es de tipo NDIR (Infrarrojo no dispersivo), lo que permite que mida las concentración de CO_2. Además incluye un sensor de temperatura y humedad. El

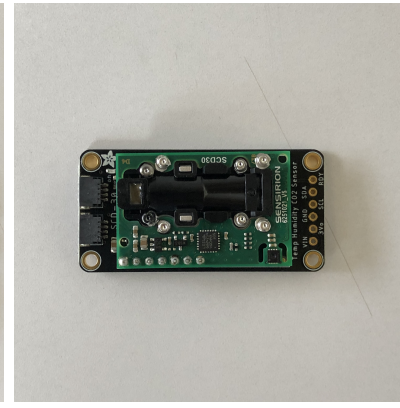
intervalo medible de CO_2 es de 400 ppm - 10,000 ppm con una precisión de $\pm$ 30 ppm.

(a) cintas



(b) MagTag

(c) cable



(d) Sensor

Figura 2.1: *Materiales*

Capítulo 3

Armado

Una vez reunidos los componentes y materiales podemos proceder con el armado. Las herramientas requeridas serán tijeras para cortar la cinta y un desarmador para colocar un par de tornillos. Ya que para ensamblar este monitor, **¡no es necesario soldar!** Seguro te parecerá genial.

1. Comenzamos por cortar un trozo de cinta doble cara transparente para unir la batería con la *MagTag*. En la parte trasera de esta, podrás observar que hay un espacio aproximadamente de 2.5 $\times$ 2 cm donde cabe perfectamente. No te olvides de conectar la pila al puerto de la alimentación de la *MagTag*.
2. Continuamos uniendo el sensor SCD-30 al acrílico trasparente de manera similar al paso anterior. En este caso, usaremos la cinta doble cara blanca procurando que quede lo más centrado posible (pierde cuidado si no queda perfectamente centrado, en lo absoluto afectará a tus mediciones).
3. El siguiente paso es unir la *MagTag* con el acrílico trasparente que porta el sensor. Esto será posible por el enroscado de los mini pies magnéticos. Así que alinea las piezas en el siguiente orden: a)*MagTag* b)Acrílico Transparente c)Sensor SCD-30, como podrás observar en la figura 3.3 Ahora, fácilmente podrás colocar los mini pies magnéticos. El resultado será similar al mostrado en la fig. 3.4.
4. Finalmente, conectaremos el sensor a la *MagTag* a través del Cable JST. Habrás notado que, a diferencia de la *MagTag* que solo cuenta con un puerto, el sensor cuenta con dos puertos; puedes escoger el que gustes.

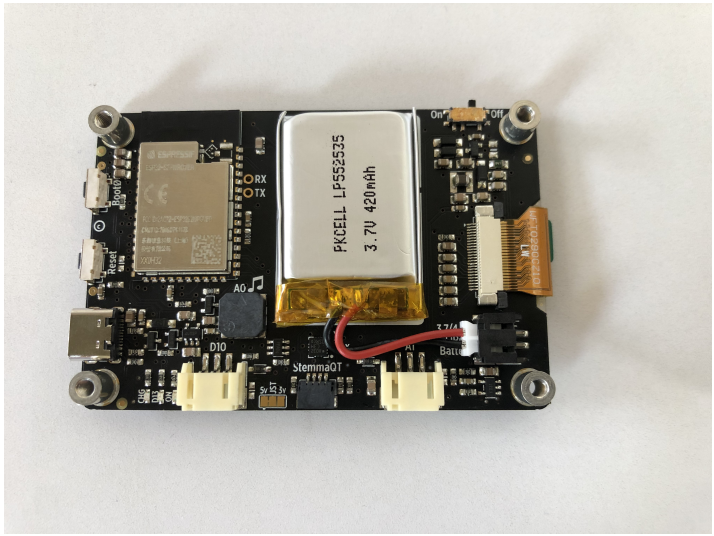


Figura 3.1: Paso 1: *unión de pila a MagTag*

Adicionalmente, por estética, se puede colocar una de las tapas acrílicas del kit, en este caso escogimos la nube (fig. 3.6) pues los datos se cargan ahí.

Una vez que el monitor está armado, ahora se debe proceder a instalar el software y establecer los parámetros para su buen funcionamiento.

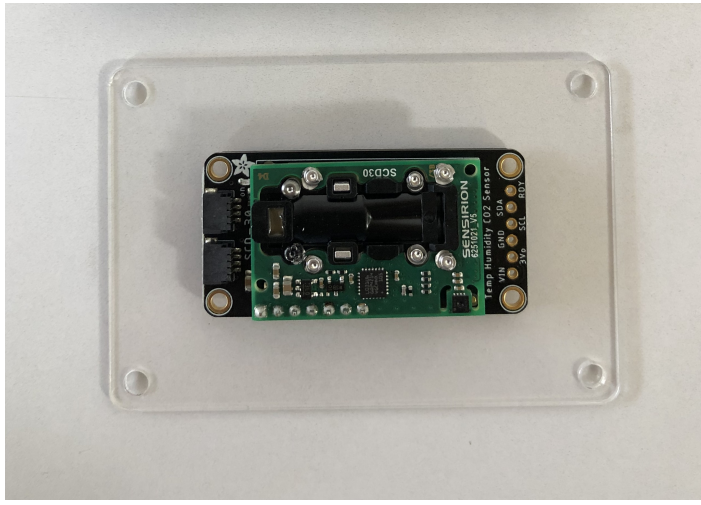


Figura 3.2: Paso 2: *Unión de sensor SCD-30 a una placa de acrílico*

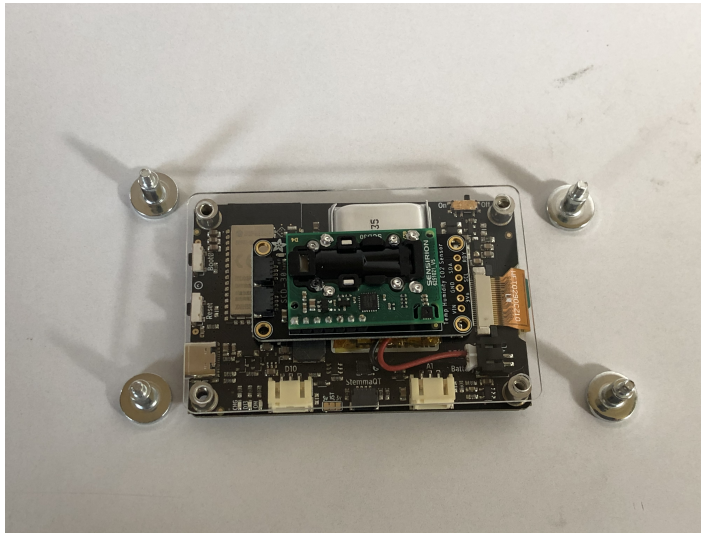


Figura 3.3: Paso 3: *alineación de componentes*

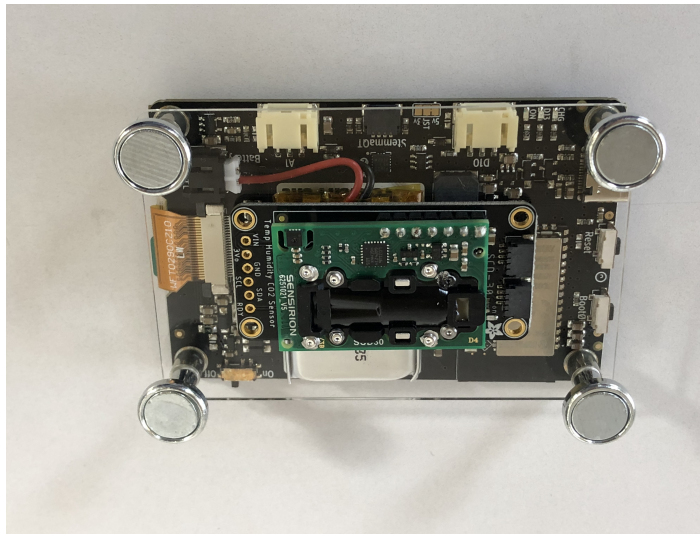


Figura 3.4: Paso 4: *enroscado de mini pies magnéticos*

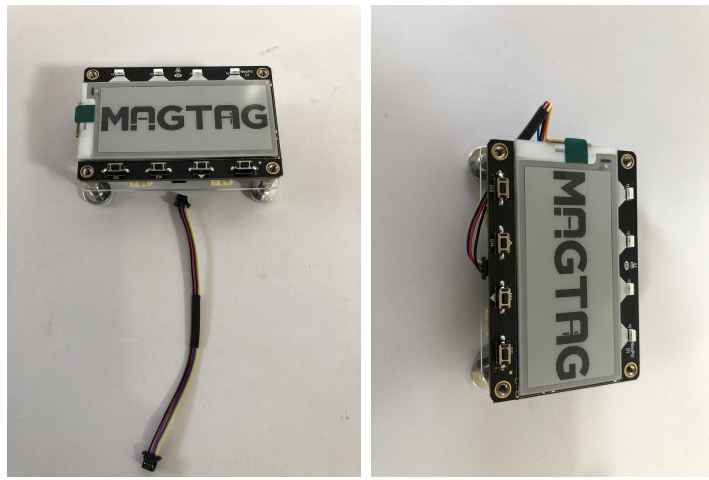


Figura 3.5: *Puesta de Cable JST*

Figura 3.6: Paso 4: *enroscado de mini pies magnéticos*

Capítulo 4

Programación

Para la "programación" se utilizarán las herramientas incluidas en tu Sistema Operativo: un Explorador de Archivos, y un Editor de Textos simples, sin formatos. Sólo necesitarás instalar el navegador *Chrome*. La idea de utilizar el navegador *Chrome* es establecer un entorno compatible con los Sistemas Operativos Linux, MacOS, Windows y hasta *Chrome*book, evitando instalar paquetes externos adicionales, y ampliando la base de usuarios que pueden utilizar este dispositivo.

4.1. Preparación de Adafruit IO

Adafruit IO es una plataforma para albergar datos de dispositivos del Internet de las Cosas (*Internet of the Things*, IoT) en la "Nube". Esta plataforma está diseñada para mostrar, responder e interactuar con dispositivos que miden, calculan y actúan en proyectos donde tanto los dispositivos como los programas computacionales son de código abierto.

Es necesario crear una cuenta de Adafruit IO en `io.adafruit.com` para grabar y mostrar los datos del monitor de CO_2. No es necesario tener una cuenta de paga, se puede utilizar una cuenta gratuita, al menos para un monitor. Si vas a colocar varios monitores, tal vez sea necesario obtener una cuenta IO Plus. Una guía para empezar se encuentra en `https://learn.adafruit.com/welcome-to-adafruit-io`

Cuando llegue el momento de configurar el archivo de secretos necesitaremos tus credenciales de identificación: `Username` y `Active Key`. Se encuentran presionando la llave amarilla a la derecha, en el página de inicio de

Adafruit IO.

Una vez que ya tienes tus credenciales, empecemos con las instrucciones para instalar el software.

4.2. Instalando CircuitPython

Cómo instalar CircuitPython (sin instalar nada en tu computadora).

Ir a

`https://circuitpython.org/board/adafruit_MagTag_2.9_grayscale/`.

Ya estando en esa página hagamos los siguientes pasos:

En el momento de ésta edición la versión de CircuitPython es la 8.0.3. Puede ser que la versión sea diferente, puedes utilizar la más reciente.

Para descargar el archivo `.BIN` selecciona el lenguaje `SPANISH` o el lenguaje de tu preferencia, y presiona el botón `DOWNLOAD.BIN`

Luego, para grabar el archivo .BIN en la *MagTag* utilizaremos la aplicación Web en

`https://nabucasa.github.io/esp-web-flasher/`

La página de esta herramienta en línea debe verse como en la imagen siguiente

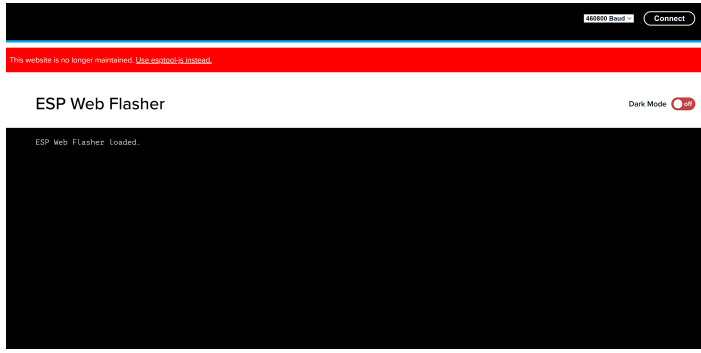


Figura 4.1: *Captura de pantalla de la aplicación web*

- Apagamos la *MagTag* con el microinterruptor, y la conectamos al bus USB. Mantenemos presionado el botón `BOOT0` y encendemos la *MagTag*. Ya podemos soltar el botón.

- En el navegador seleccionamos 460800 Baud (en la imagen 4.1 en la esquina superior derecha podemos observar que se encuentra del lado izquierdo el botón «Conectar»)...
- y presionamos Conectar (de haberse hecho la conexión con la MagTag correctamente aparecerá la siguiente pantalla)...

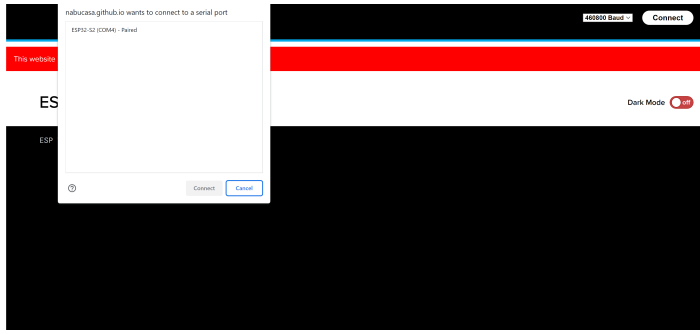


Figura 4.2: *Visualización de lista de dispositivo conectados*

- Seleccionamos ESP32-S2 (entre paréntesis aparece el nombre del puerto serial)...
- Seleccionamos Erase y le decimos OK al diálogo de confirmación de borrado... (al realizarse el enlase con la MagTag aparecen los botones “Erase”, “Program” & “Choose file” como se muestra en la imagen siguiente)

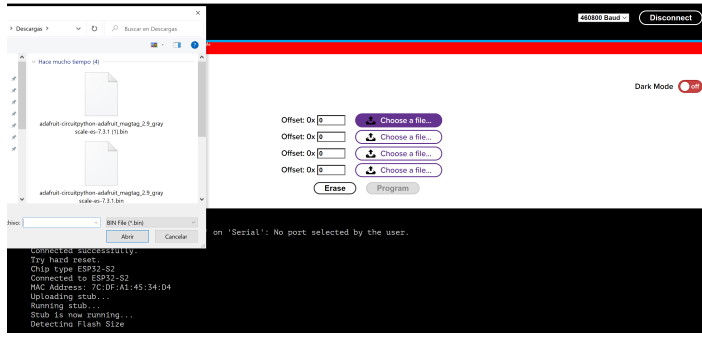


Figura 4.3: *Botones*

- Una vez borrada la memoria flash, procedemos a programar CircuitPyhton. Presionamos "Choose file", y en la lista seleccionamos el archivo .BIN que hemos descargado.
- Presionamos el botón de Program
- Cuando termine, presionamos Desconectar
- Y reiniciamos la MagTag con el botón de `RESET`.

Si todo ha salido bien, se podrá observar en el explorador de archivos un nuevo disco externo con nombre `CIRCUITPY` con una capacidad aproximada de 1MB:

4.3. Descargar los recursos

En el navegador ir a `https://github.com/trblnyx/magtag-co2`. Presionar el botón verde de `Code` y descargar el archivo ZIP. No es necesario crear una cuenta de *GitHub*, pero si deseas colaborar con el proyecto puedes crear una. Si quieres saber más detalles sobre *GitHub* hay mucha información disponible en la red: `https://docs.github.com/es` es un buen lugar para empezar.

Luego, en el explorador de archivos, descomprimir el archivo MagTag-co2-mail.zip. Entrar a la carpeta `magtag-co2-main/src`. Encontrarás dos carpetas: `bmps` y `fonts`; y dos archivos: `code.py` y `secrets.py`. Seleccionar todo y copiar en el disco CIRCUITPY.

En la pantalla de la *MagTag* aparece un error indicando que faltan las bibliotecas. Por lo tanto requerimos hacer el siguiente paso.

4.4. Instalar Bibliotecas

El Monitor de CO_2 utiliza varias bibliotecas de Python que se encuentran en `https://circuitpython.org/libraries`.

En la sección de `BUNDLES` se descarga el archivo ZIP. En el momento de ésta edición es la versión `adafruit-circuitpython-bundle-8.x-mpy-20230307.zip`.

Luego, utilizando el explorador de archivos, copiar las siguientes bibliotecas del archivo ZIP a la carpeta `lib` del disco externo `CIRCUITPY`:

Carpetas:

- adafruit_bitmap_font
- adafruit_display_text
- adafruit_imageload
- adafruit_io
- adafruit_magtag
- adafruit_minimqtt
- adafruit_portalbase

Archivos:

- adafruit_fakerequests.mpy
- adafruit_lis3dh.mpy
- adafruit_requests.mpy
- adafruit_scd30.mpy
- neopixel.mpy
- simpleio.mpy

Ahora sólo falta configurar el archivos de "secretos" que son los archivos donde se guardan tus credenciales de Adafruit y la forma de conectarse a la red WiFi que hayas considerado.

4.5. Archivo de secretos

En el explorador de archivos se selecciona el archivo secrets.py y cambia los valores necesarios (en realidad, son todos), para que coincidan con tu configuración. Necesitaremos la información de tu red WiFI, tu zona horaria, y tus credenciales de Adafruit IO. ¡Ah! Y un nombre o apodo que aparecerá al encender el dispositivo.

- **ssid**: El nombre de tu red WiFI
- **password**: La contraseña de tu red WiFI
- **timezone**: Tu zona horaria. Por ejemplo: `America/Mexico_City`. En `http://worldtimeapi.org/timezones` hay una lista completa.
- **aio_username**: Tu identificador de usuario de `io.adafruit.com`

- **aio_key**: Tu llave de identificación de `io.adafruit.com`
- **owner**: Tu nombre, o apodo, aparece cuando el *MagTag* enciende.

Una vez que ya hemos instalado los archivos y el software en el monitor, nos resta explicar cómo podemos visualizar los datos. Por supuesto, podemos verlos en la pantalla del monitor, pero seguramente quisiéramos trabajar con ellos para hacer algún historial o utilizarlos como fuentes para otros cálculos.

En el siguiente capítulo explicaremos como visualizar los datos en la "Nube".

Capítulo 5

Visualizar los datos con Adafruit IO

En este capítulo veremos cómo podemos visualizar los datos en la "Nube". Para esto, nos hemos preparado al instalar el software. En particular, entre las bondades de usar la nube es que nos releva de usar recursos para almacenar los datos, i.e., no necesitamos destinar una computadora para almacenar los datos, ni tampoco un servidor de datos, ni redes de telecomunicación. Solamente necesitamos un acceso a Internet. Usaremos Adafruit.io[1] que es un servicio en la nube. Uno puede conectarse a este servicio mediante Internet. De hecho, el software que hemos bajado en el capítulo anterior nos facilitará la conexión, dado que Adadruit está diseñado principalmente para almacenar y luego recuperar datos, ¡pero puede hacer mucho más que eso!

5.1. Fuentes de datos

Las fuentes de datos (feeds) son el núcleo del sistema Adafruit IO y con ellos sabe cómo y para quién desplegar los datos. Las fuentes de datos contienen información codificada, que se conoce como metadatos, sobre los datos que enviamos a Adafruit IO. Esto incluye la configuración de si los datos son públicos o privados, a qué tipo de licencia deben ser asociado los datos del monitor almacenados y una descripción general de toda la información que

[1] `https://learn.adafruit.com/welcome-to-adafruit-io/what-is-adafruit-io`

se envía desde el monitor hacia la Nube. La fuente de datos también contiene los datos referentes al registro del monitor que se envían a Adafruit IO desde el dispositivo. Para cada conjunto único de datos que se envíe al sistema se debe crear una fuente de datos. En la configuración que proponemos, el Monitor de CO_2 crea una fuente de datos con cinco registros:

1. **co2ppm**
   La cantidad de CO_2 en el ambiente, medida en partes por millón (ppm).
2. **relative-humidity**
   El porcentaje de Humedad Relativa en el ambiente.
3. **temperature**
   La temperatura ambiente en $°C$
4. **vbat**
   El voltaje de la batería.
5. **rssi**
   La intensidad de la señal de la red inalámbrica, en dBm.

Como se observa los primeros tres datos son los que nos interesan para monitorizar el edificio y la calidad del aire. Con ellos podemos responder tanto a la necesidad proveer información para la climatización del edificio y para la calidad del aire para evitar contagios por COVID-19

5.2. Tableros

Adafruit.io puede manejar y visualizar múltiples fuentes de datos. Los tableros son una característica integrada en Adafruit IO que te permite trazar, graficar, medir, registrar y mostrar tus datos. Puedes ver tus tableros desde cualquier parte del mundo.

5.2.1. Crear un tablero

Cuando inicies sesión en tu cuenta io.adafruit.com, serás redirigido a tu lista de tableros. La primera vez sólo tendrás el Tablero de Bienvenida.

Para crear un tablero hay que dar *click* en el botón de Nuevo Tablero (+ New Dashboard). Luego hay que ingresar Medición Monitor CO_2 como nombre, y Tablero de Medición del Monitor CO_2 en la descripción del tablero nuevo y hacer *click* en el botón de Crear cuando hayas terminado. Una

vez que se haya creado tu tablero, dar *click* en el nombre de tu nuevo tablero para cargarlo. Ahora deberías ver tu nuevo tablero en blanco.

5.2.2. Agregar bloques

Los bloques son artefactos (*widget*) que pueden ser agregados a tu tablero. Hay algunos bloques que se pueden usar como salidas y otros que se pueden usar como entradas. Para agregar un nuevo bloque, hay que abrir el menú de ajustes, identificado con el ícono de un engrane, en la parte superior derecha del tablero, y seleccionar Crear nuevo bloque (+ Create new block).

Se presentará una lista de tipos de bloques para elegir. Para este tablero utilizaremos el bloque de Medidor (Gauge) para la fuente de datos **co2ppm**.

Al darle click a Medidor (Gauge), aparecerá el diálogo para conectar una fuente de datos. Seleccionamos **co2ppm** y presionamos el botón de Siguiente (Next step). Enseguida aparecerá el diálogo de Configuración del bloque (Block settings). Escribimos CO2 como Título (Block Title), el número 2000 en Valor Máximo (Gauge Max Value), ppm en Etiqueta (Gauge Label), el número 800 en Advertencia de valor alto (High Warning Value), y el número 0 (cero) en Decimales (Decimal Places). Cuando hayas terminado presiona el botón Crear Bloque (Create Block).

En el tablero aparecerá el bloque de medición del CO_2.

Repetimos el procedimiento para añadir el voltaje de la batería. Creamos un nuevo bloque del tipo Medidor (Gauge), lo conectamos a la fuente de datos **vbat**, y en el diálogo de ajustes escribimos lo siguiente: VBat para el Título, 5 para el Valor Máximo, la letra V como Etiqueta, 3.3 para Advertencia de valor bajo (Low Warning Value), y 1 para Decimales. Cuando hayas terminado presiona el botón Crear Bloque.

En el tablero aparecerá el bloque de medición del voltaje de la batería.

5.2.3. Tablero de gráfico de líneas

En los tableros también se pueden colocar bloques que muestren los valores del monitor a través del tiempo mediante una gráfica de líneas.

Veamos si lo has entendido:

Tienes que crear un Nuevo Tablero, con nombre Gráficos Monitor CO_2 y descripción **Tablero de Gráficos de Líneas**.

Selecciona el tablero recién creado y añade un nuevo bloque, pero esta vez selecciona el tipo Gráficos de Línea (Line Graph). Conéctalo a la fuente

de datos co2ppm.

En el diálogo de Configuración del Bloque (Block settings) escribe lo siguiente: CO_2 para el Título (Block Title), ppm para la etiqueta del eje de las Y (Y-Axis Label), y 0 (cero) para Decimales (Decimal places). Presiona el botón de Crear bloque cuando hayas terminado.

En el tablero aparecerá el bloque de gráfico de líneas para el CO_2. A veces es necesario recargar la página para que aparezca la gráfica en el bloque recién creado. La gráfica se actualizará automáticamente conforme se vayan recibiendo nuevos datos.

¿Podrás añadir las gráficas de las demás fuentes de datos?

En el siguiente capítulo añadimos algunas recomendaciones finales.

Capítulo 6

Recomendaciones finales

El lugar dónde se coloca el monitor debe estar cerca de una toma de corriente eléctrica para que esté conectado siempre a la corriente. El lugar específico en la habitación deberá estar al menos a 1 metro de cualquier ocupante. De estar cerca de una persona y debido a la sensibilidad del monitor, las mediciones no serán representativas de la calidad del aire en la habitación.

En caso de que el monitor se descalibre por alguna extraña razón, (se observa esta descalibración cuando la mediciones son anormalmente altas o bajas), se debe esperar al proceso de calibración automática. Es decir, se debe mantener encendido el monitor sin desconectarlo para que al cabo de 3 días aproximadamente este se calibrará de manera automática al tomar el promedio bajo como referencia de las mediciones. Para ello se debe colocar en un lugar donde se pueda alcanzar los 400 ppm estándares de la aire atmosférico. Este es un procedimiento estándar.

La vida útil de la batería es de mas de 300 ciclos, se recomienda cambiar cuando el voltaje entregado sea menor a 3.2 V a pesar de estar conectada a la corriente.

Monitorear los datos entregados por el monitor por lo menos una vez a la semana. De esta forma, en caso de presentarse una anomalía podrá ser atendida sin afectar en de forma grave el historial de datos obtenidos.

La red WiFI que seleccionemos en el apartado de archivos secretos debe ser una red estable y sin interrupciones, se debe procurar que la señal recibida por la *MagTag* no tenga interferencias.

Es posible utilizar el Monitor sin señal WiFi; pero se pierden la mayoría de las bondades del Monitor que hemos configurado.

En el momento de ésta edición la última versión de CircuitPython es la 8.0.3. Se puede instalar una versión más nueva siguiendo las instrucciones de instalación utilizando el número de versión más reciente.

Bibliografía

[1] http://www.acmor.org/publicaciones/monitores-de-co2-para-la-evaluaci-n-de-espacios-ventilados-en-el-regreso-a-la-presencialidad

Apéndice

Código del programa

La última versión siempre estará disponible en `https://github.com/trblnyx/magtag-co2`

code.py

```
# Proyecto Monitor CO2
#   version MagTag
#
# (C) 2021, Hector Daniel Cortes Gonzalez <hdcg@ier.unam.mx>
# (C) 2021, Laboratorio de Tecnologias Abiertas, LaTA+
# (C) 2021, Instituto de Energias Renovables
# (C) 2021, Universidad Nacional Autonoma de Mexico
#
# CO2 Sensor: Sensirion SCD30
import time
tm=time.monotonic()
import board
import digitalio
led = digitalio.DigitalInOut(board.D13)
led.direction = digitalio.Direction.OUTPUT
led.value = True
import busio
import adafruit_scd30
import terminalio
import displayio
import alarm
import wifi
import adafruit_lis3dh
import adafruit_imageload
from adafruit_bitmap_font import bitmap_font
from adafruit_display_text import label
from adafruit_magtag.magtag import MagTag
from secrets import secrets

print("Start: ", tm)

SHORT_DELAY=0.128
DELAY=1.024
LONG_DELAY=8.192
MEASUREMENT_INTERVAL=2
```

```
POST_INTERVAL=180
CO2LOW=400
CO2MID=700
CO2HIGH=1000
PLAY_TONE=2
TONE_FRECUENCY=880
TONE_DURATION=0.333
SPLASH_IMAGE="/bmps/splash.bmp"
BIG_FONT="/fonts/DejaVuSans-Bold-75.pcf"
BAR_FONT="/fonts/DejaVuSans-18.pcf"

print("*" * 40)
print("MonitorCO2 MagTag version")
print("(C) 2021, hdcg@ier.unam.mx")
print("(C) 2021, LaTA+")
print("(C) 2021, IER-UNAM")
print("(C) 2021, UNAM")
print("*" * 40)
#
# Begin with SCD30, makes no sense no sensor working
#
# SCD-30 has tempremental I2C with clock stretching, datasheet recommends
# starting at 50KHz
i2c = busio.I2C(board.SCL, board.SDA, frequency=50000)
scd30 = adafruit_scd30.SCD30(i2c)
if scd30.measurement_interval != MEASUREMENT_INTERVAL :
  scd30.measurement_interval = MEASUREMENT_INTERVAL
  print("Setting MSI=", scd30.measurement_interval)
if not scd30.self_calibration_enabled :
  scd30.self_calibration_enabled=True
  print("Setting ASC=", scd30.self_calibration_enabled)
#
# Accel also uses i2c
#
lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c, address=0x19)
lis3dh.range = adafruit_lis3dh.RANGE_2_G
if lis3dh.acceleration.y > 0 :
  rotate=270
else :
  rotate=90
lis3dh.data_rate = 0
#
# Splash image IER-UNAM on PowerOn
#
if not alarm.wake_alarm:
  print("powerOnSelfTest:")
  magtag=MagTag(default_bg=SPLASH_IMAGE)
  magtag.display.rotation=rotate
  magtag.add_text(
    text_font=BAR_FONT,
    text_position=(
      magtag.graphics.display.width - 2, 2
    ),
    text_anchor_point=(1.0, 0.0),
  )
  COLORS=[(255, 0, 0), (255,127,0), (255, 255, 0), (0, 255, 0), (0, 255, 255), (0, 0,
  magtag.peripherals.neopixel_disable = False
  magtag.peripherals.neopixels.auto_write=False
  magtag.set_text(secrets["owner"])
  magtag.peripherals.play_tone(TONE_FRECUENCY, TONE_DURATION)
  while magtag.display.busy:
    for i in range(8):
      for j in range(3,0,-1):
        magtag.peripherals.neopixels[j]=magtag.peripherals.neopixels[j-1]
      magtag.peripherals.neopixels[0]=COLORS[i]
      magtag.peripherals.neopixels.show()
      time.sleep(SHORT_DELAY)
  for i in range(3):
    for j in range(3,0,-1):
      magtag.peripherals.neopixels[j]=magtag.peripherals.neopixels[j-1]
    magtag.peripherals.neopixels.show()
```

```
    time.sleep(SHORT_DELAY)
  magtag.peripherals.neopixel_disable = True
  led.value = False
  magtag.exit_and_deep_sleep(POST_INTERVAL)
#
# WokeUp!
#
magtag = MagTag()
display = magtag.display
display.rotation = rotate
#
# Make UI
#
# Make a background color fill
color_bitmap = displayio.Bitmap(display.width, display.height, 1)
color_palette = displayio.Palette(1)
color_palette[0] = 0xFFFFFF
backGround = displayio.TileGrid(color_bitmap, pixel_shader=color_palette)

bigFont=bitmap_font.load_font(BIG_FONT)

co2Bar = label.Label(
  bigFont,
  text="Error",
  color=0,
  anchor_point=(1.0, 0.5),
  anchored_position = (3 * display.width // 4 - 2, display.height // 2),
)

barFont = bitmap_font.load_font(BAR_FONT)

centerBar = label.Label(
  barFont,
  text="CO2\nppm",
  color=0,
  anchor_point=(0.0, 0.5),
  anchored_position = (3 * display.width // 4 + 2, display.height // 2),
)

topBar = label.Label(
  barFont,
  text="1234567890" * 3,
  color=0,
  anchor_point=(0.0, 0.0),
  anchored_position = (2, 2),
)

bottomBar = label.Label(
  barFont,
  text="1234567890" * 3,
  color=0,
  anchor_point=(0.0, 1.0),
  anchored_position = (2, display.height - 2),
)

group = displayio.Group()
group.append(backGround)
group.append(co2Bar)
group.append(centerBar)
group.append(topBar)
group.append(bottomBar)

#
# First, get CO2 concentration
#
while not scd30.data_available:
  time.sleep(0.5)

co2ppm=int(scd30.CO2)
co2Bar.text=str(co2ppm)
```

```
#
# Network operations
#
try:
  magtag.network.connect()
except Exception as e:
  topBar.text = str(e)

if magtag.network._wifi.is_connected :
  print("My IP address is", wifi.radio.ipv4_address)
  topBar.text = "[%s] (%ddBm)" % (wifi.radio.ap_info.ssid, wifi.radio.ap_info.rssi)

  try:
    response=magtag.get_local_time()
    print(response)
  except Exception as e:
    print(str(e))

lt=time.localtime()

vbat = magtag.peripherals.battery
print("vbat=", vbat)

bottomBar.text = "%02d:%02d %.1fV %+.1fC %.0f%%HR" % (lt.tm_hour, lt.tm_min, vbat, sc
#
# transfer data
#
if magtag.network._wifi.is_connected:
  try:
    print("sending data...")
    magtag.network.push_to_io("magtag-co2ppm", co2ppm)
    magtag.network.push_to_io("magtag-temperature", scd30.temperature)
    magtag.network.push_to_io("magtag-relative-humidity", scd30.relative_humidity)
    magtag.network.push_to_io("magtag-vbat", vbat);
    magtag.network.push_to_io("magtag-rssi", wifi.radio.ap_info.rssi)
    print("data sent")
  except Exception as e:
    topBar.text = str(e)
#
# lights and sounds show
#
magtag.peripherals.neopixel_disable = False

light = magtag.peripherals.light
print("light=", light)

COLOR_INTENSITY=4+light//256
COLOR_GREEN=(0,COLOR_INTENSITY,0)
COLOR_YELLOW=(COLOR_INTENSITY,COLOR_INTENSITY,0)
COLOR_RED=(COLOR_INTENSITY,0,0)

display.show(group)
display.refresh()

if co2ppm>=CO2LOW and co2ppm<CO2MID:
  magtag.peripherals.neopixels.fill(COLOR_GREEN)
  if PLAY_TONE>2:
    magtag.peripherals.play_tone(TONE_FRECUENCY, TONE_DURATION)

if co2ppm>=CO2MID and co2ppm<CO2HIGH:
  magtag.peripherals.neopixels.fill(COLOR_YELLOW)
  if PLAY_TONE>1:
    magtag.peripherals.play_tone(TONE_FRECUENCY, TONE_DURATION)
    time.sleep(TONE_DURATION);
    magtag.peripherals.play_tone(TONE_FRECUENCY, TONE_DURATION)

if co2ppm>=CO2HIGH:
  magtag.peripherals.neopixels.fill(COLOR_RED)
  if PLAY_TONE>0:
    magtag.peripherals.play_tone(TONE_FRECUENCY, TONE_DURATION)
    time.sleep(TONE_DURATION);
```

```
    magtag.peripherals.play_tone(TONE_FRECUENCY, TONE_DURATION)
    time.sleep(TONE_DURATION);
    magtag.peripherals.play_tone(TONE_FRECUENCY, TONE_DURATION)

while magtag.display.busy:
  time.sleep(SHORT_DELAY)

magtag.peripherals.neopixel_disable = True
#
# Clean up and exit (er, deep sleep)
#
te=time.monotonic() - tm
print("TimeElapsed=", te)
if te>=POST_INTERVAL/2:
  ds = POST_INTERVAL
else:
  ds = POST_INTERVAL - te
print("DeepSleep=", ds)

led.value = False
magtag.exit_and_deep_sleep(ds)
```

secrets.py

```
# This file is where you keep secret settings, passwords, and tokens!
# If you put them in the code you risk committing that info or sharing it

secrets = {
  "ssid": "YOUR-WIFI-NETWORK",
  "password": "YOUR-WIFI-PASSWORD",
  "timezone": "YOUR/TIMEZONE",  # http://worldtimeapi.org/timezones
  "aio_username": "YOUR-AIO-USERNAME",
  "aio_key": "YOUR-AIO-KEY",
  "owner": "YOUR-NAME",
}
```