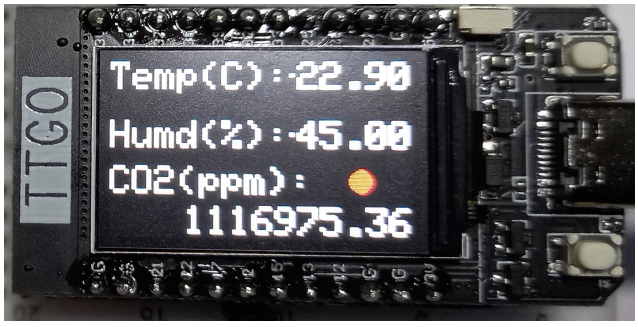
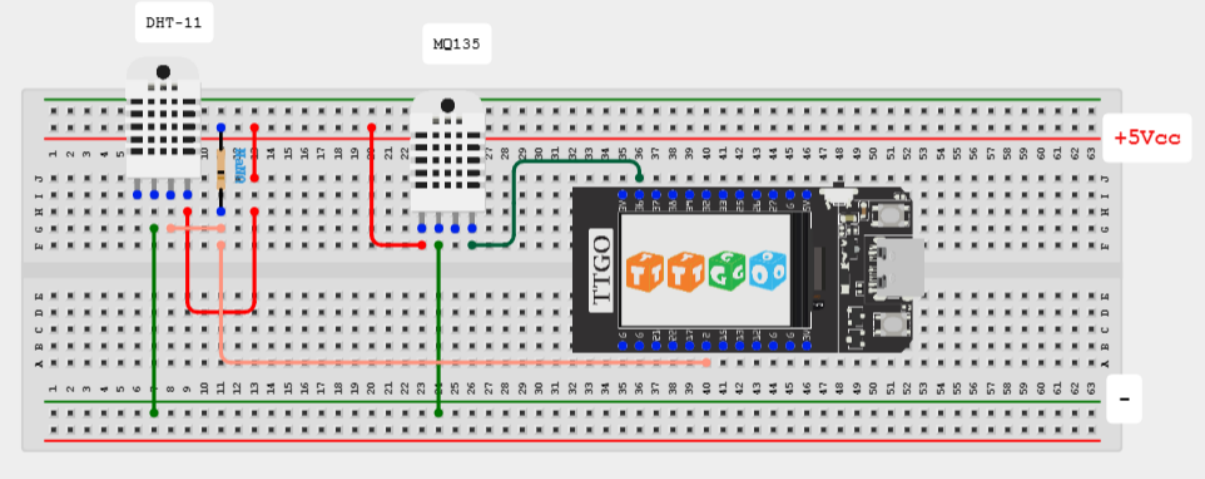
Medidor de CO2, temperatura y humedad con TTGO ESP32 LCD de 1,14 pulgadas

Utilizamos TTGO-Placa de Control LCD de 1,14 pulgadas para presentar los valores de CO2, Temperatura y humedad.



Dibujamos Led de color, según sea el valor del CO2.



Para medir la calidad del aire utilizamos el sensor MQ-135, que, como podemos ver en su ficha técnica, puede medir determinadas sustancias, tales como NH3, alcohol, benceno, humo, CO2, etc.

https://www.luisllamas.es/arduino-detector-gas-mq/

Los límites de seguridad de CO2 en partes por millón (ppm) para viviendas. se recomienda que oscile entre 400 y 800 ppm.

Vamos a medir la concentración de CO2 en partes por millón, la temperatura y la humedad y mostraremos los datos en la pantalla que lleva incorporada la TTGO-Placa de Control LCD de 1,14 pulgadas.

En la misma pantalla mostraremos un semaforo que simula leds (Rojo, Ambar, Verde), según el nivel de concentración de CO2.

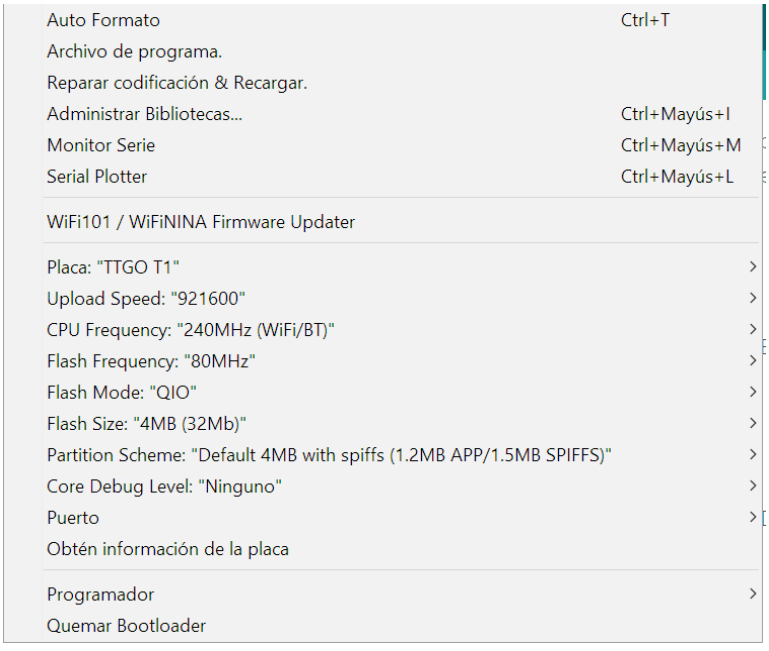
Hardware.

Número	Componente
1	TTGO-Placa de Control LCD de 1,14 pulgadas
1	Sensor de calidad de aire MQ-135
1	Sensor de temperatura y humedad DHT-11
1	Resistencia 10K ohmios
1	Fuente de alimentación de 5v

El software.

- Arduino IDE
- #include <Wire.h>
- #include <TFT_eSPI.h> // Libreria LCD
- #include <SPI.h>
- #include <DHT.h> // Librería DHT-11
- #include "MQ135.h" // Librería MQ135
- **calibrarMQ135.ino**
- **TTGO_CO2_monitorOLED.ino**

Configurar entorno Arduino IDE para cargar scrip



Funcionamiento del circuito

El sensor MQ-135 mide la concentración de CO2 en partes por millón, y el sensor DHT-11 mide la temperatura y la humedad.

Estos tres datos se envían a la pantalla LCD de la placa TTGO ESP32, para mostrar los valores.

Según la concentración de CO2, muestra un circulo de color (rojo; CO2>= 800, amarillo: CO2>= 551 y <= 799 y verde: CO2<= 550). Estos valores se pueden ajentar en el scrip.

Preparación y ajuste de la referencia de resistencia RZERO para el contenido de CO2 atmosférico

Es conveniente calibrar el sensor MQ-135 para que mida los valores correctos que utilizaremos en el siguiente paso. El sensor debe funcionar a 5 voltios durante a 2 horas proximadamente cuando es nuevo para eliminar las impurezas.

Después, lo mejor es hacer el ajuste de referencia de la resistencia RZEO en el circuito final, ya que se hace con las tensiones de alimentación y funcionamiento normalmente utilizadas y la medición es óptima.

El objetivo de esta calibración es ajustar la resistencia del nivel de CO2 atmosférico para que el sensor MQ-135 mida una concentración entre 300 y 450 ppm, que corresponde a la concentración normal de este gas en el ambiente, cambiando el valor de RZERO hasta que tengamos los valores normales.

Podemos utilizar el siguiente scrip:

calibrarMQ135.ino

```
#include "MQ135.h"
#define RZERO 1

MQ135 gasSensor = MQ135(A0);
int val;
int sensorPin = A0;
int sensorValue = 0;

void setup() {
  Serial.begin(115200);
  pinMode(sensorPin, INPUT);
}

void loop() {
  val = analogRead(A0);
  Serial.print ("raw = ");
  Serial.println (val);
  float zero = gasSensor.getRZero();
  Serial.print ("rzero: ");
  Serial.println (zero);
  float ppm = gasSensor.getPPM();
  Serial.print ("ppm: ");
  Serial.println (ppm);
  delay(2000);
```

```
}
```

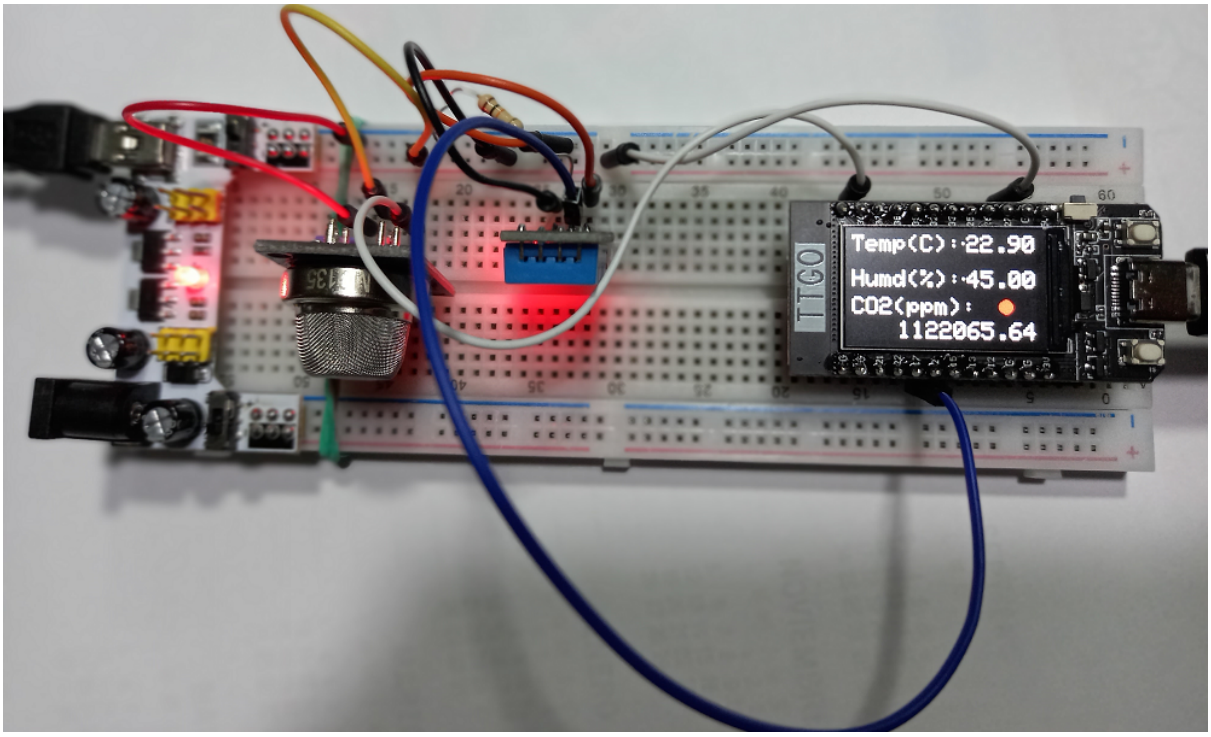
En el siguiente scrip, tenemos la configuración de la ESP32:

TTGO_CO2_monitorOLED.ino

```
/* ************************************************************************
 *  TTGO_CO2_monitorOLED
 *  Monitorización de CO2, temperatura y humedad con MQ-135 y DHT-11,
 *  con TTGO-Placa de Control LCD de 1,14 pulgadas
 *  Basado en:
 *
https://www.az-delivery.de/es/blogs/azdelivery-blog-fur-arduino-und-raspberry-pi/uberwachung-von-luftqualitat-temperatur-und-luftfeuchtigkeit-mit-mq-135-und-dht11?utm_source=AZ-MAILING+DEUTSCH&utm_campaign=24952e330f-EMAIL_CAMPAIGN_7_31_2021_Blog_COPY_01&utm_medium=email&utm_term=0_569b1a8f94-24952e330f-19163868&goal=0_569b1a8f94-24952e330f-19163868&mc_cid=24952e330f&mc_eid=0b09565020
 *
 *  En  https://github.com/Xinyuan-LilyGO/TTGO-T-Display/blob/master/TFT_eSPI/TFT_eSPI.h
 *  podemos ver las funciones para controlar el LCD
 *
 *  Jarp 2022  tecnouniab+esp32@gmail.com
 */
/*************   Librerias necesarias para el proyecto    **************************/
#include <Wire.h>
#include <TFT_eSPI.h>   // Libreria LCD
#include <SPI.h>
#include <DHT.h>        // Librería DHT-11
#include "MQ135.h"      // Librería MQ135
// Los pines TFT se han configurado en la biblioteca TFT_eSPI en el archivo de configuración de usuario TTGO_T_Display.h
// #define TFT_MOSI            19
// #define TFT_SCLK            18
// #define TFT_CS               5
// #define TFT_DC              16
// #define TFT_RST             23
// #define TFT_BL               4   // Pin de control de retroiluminación de la pantalla
TFT_eSPI tft = TFT_eSPI(135, 240);  // Fija resolución LCD en pixels
//TFT_eSPI tft = TFT_eSPI();        // Tambien podemos sin los valores y los muestra por defecto


int cx1,cy1,linea1, linea11 ,linea2, linea3; //para pixel y de cada linea a visualizar

/******    Parámetros y variables del sensor de temperatura y humedad DHT11   *****/
#define DHTPIN 2  // pin 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
float t, h;

/******************    Parámetros y variables del sensor MQ-135    ****************/
#define RZERO 1
MQ135 gasSensor = MQ135(A0);
int val;
int sensorPin = A0; //pin 36
int sensorValue = 0;
float ppm, zero;

void setup( ){
  Serial.begin(115200);
  dht.begin();                                      // iniciar DHT-11
   tft.init();
```

```
  tft.setRotation(1);                       // Horientación pantalla 0:Vertical, 1:Horizontal
  tft.fillScreen(TFT_BLACK);                // Pone pantalla en negro
  tft.setTextColor(TFT_WHITE, TFT_BLACK);       // Color texto y fondo
  tft.setTextDatum(TC_DATUM);               // Alineamiento datos en pantalla

 pinMode(sensorPin, INPUT);                                          // MQ-135 pin data
 Serial.println (tft.width());              // Dimensiones de pantalla
 Serial.println (tft.height());
}

/***********************    LOOP BLOCK    ************************************/
void loop( ) {

 t = dht.readTemperature();                                          // Lee temperatura
 h = dht.readHumidity();                                          // Lee Humedad

 if (isnan(h) || isnan(t)) {                                        // Si no lee temperatura y humedad
   Serial.println("Fallo de lectura del sensor DHT !!!");
 }

 val = analogRead(A0);                                               // lee CO2
 zero = gasSensor.getRZero();                                       // Valor de calibración para el sensor
MQ-135
 ppm = gasSensor.getPPM();                                           // Fórmula en la biblioteca para
obtener las ppm de CO2
 Serial.print( "T = " );                                          // Mostrar valores por el puerto serie
 Serial.print(t);
 Serial.print(" ºC, H = ");
 Serial.print(h);
 Serial.print( "%, " );
 Serial.print ("raw = ");
 Serial.print (val);
 Serial.print (", rzero: ");
 Serial.print (zero);
 Serial.print (", ppm: ");
 Serial.println (ppm);
 delay (1000);
 Visualizar_Datos();
}

void Visualizar_Datos() {                                            // Visualiza datos
 linea1=0; //0 45 90                // Valor coordenada_y 1ª linea
 linea2=45;                        // Valor coordenada_y 2ª linea
 linea11=80;                        // Valor coordenada_y 3ª linea
 linea3= 110;                       // Valor coordenada_y dato 3ª linea
 cx1= 200;                         // Valor coordenada_x 3ª linea para led
 cy1= linea11 +10;                  // Valor coordenada_y 3ª linea para led
// tft.fillScreen(TFT_BLACK);

 tft.setTextSize(3);                   // Tamaño texto
 tft.setTextColor(TFT_YELLOW);          // Color texto amarillo
 tft.drawString("Temp(C):", 12, linea1); // Muestra texto
 tft.setTextColor(TFT_WHITE);           // Color texto blanco
 tft.drawFloat(t, 2, 560, linea1);       // Muestra número decimal

 tft.setTextColor(TFT_YELLOW);
 tft.drawString("Humd(%):", 12, linea2);
 tft.setTextColor(TFT_WHITE);
 tft.drawFloat(h, 2, 555, linea2);

 tft.setTextColor(TFT_YELLOW);
  tft.drawString("CO2(ppm):", 20, linea11);
```

```
    if (ppm >= 800) {                    // Si la concentración de CO2 es igual o superior a 800
    tft.fillCircle(cx1, cy1 , 9,  TFT_RED); // Dibuja circulo relleno en rojo, alarma de CO2
   }
   if (ppm >= 551 && ppm <= 799) {
    tft.fillCircle(cx1, cy1, 9,  TFT_YELLOW);
   }

   if (ppm <= 550)  {
    tft.fillCircle(cx1, cy1, 9,  TFT_GREEN);
   }
   tft.setTextColor(TFT_WHITE);
   tft.drawFloat(ppm, 2, 550, linea3);
}
```



Mejoras:

- Enviar datos mediante Wifi a IoT, Telegram al pulsar botón incorporado en la placa o requerir el envio con un bot.
- Añadir zumbador de alarma.
- LCD apagado hasta pulsar botón incorporado en la placa para ahorrar batería y alargar la vida de LCD