LORA

Egileak

Ander U. eta Xabi

1. orrialdea

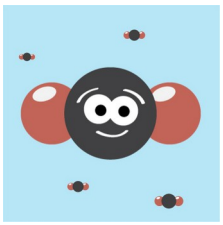
Ziklo bukaerako proiektua: aurkibidea

1 Erronkaren planteamendua

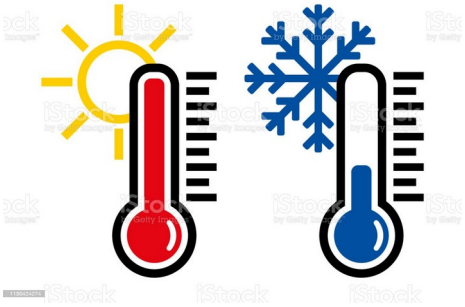
Iurreta ikastetxeko klaseetako datu batzuk lortzea eta balio horiek The Things Networks web orrialdean ikustea Lora Gateway baten bitartez; horrez gain, Raspberry batekin eskolako zerbitzarira konektatzen da eta horrela klase bakiotzeko balioak lortuko ditugu. Hurrengo datuak lortuko ditugu:

- Hezetasuna

- CO_2



- Tenperatura



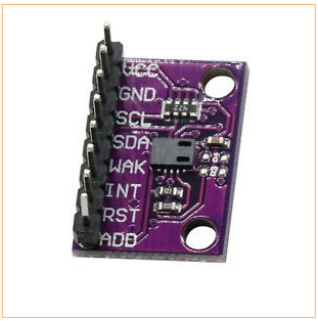
2 Hausnarketak

2.1 DHT11:

DHT11 tenperatura eta hezetasuna neurtzen dituen sentsore sinple bat da; nahiko merkea, baina modulu batean muntatuta ere aurki dezakezu (PCB batean muntatuta, errazago erabiltzeko), Arduinorentzat horrelako osagai elektronikoetan ohikoa den bezala. Plakaren kasuan, 5 kilo ommioko pull-up erresistentzia eta funtzionamenduaren berri ematen digun Led bat ditu. Fidagarritasun eta egonkortasun handia du, kalibratutako seinale digitalaren ondorioz.

2.2 CCS811:

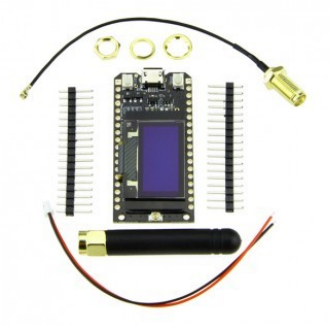
CCS811 kontsumo baxuko aire-kalitatearen sentsore bat da. Oxide metalezko sentsore bat (MOX) erabiltzen du osagai lurrunkor organiko (TVOC) eta CO2 eduki baliokideen guztizko baliokideak neurtzeko. Moduluak mikro kontrolagailu bat (MCU) barne hartzen du, gasaren irakurketak zuzentzeko, tenperaturaren eta giro-hezetasunaren arabera.



- Energia hornidura: 3.3V gomendatuak
- Energia kontsumoa: 60mW-tik gora
- ETVOC detekzio-eremua: 0 # 32768ppb
- ECO2 detektatzeko eremua: 400 # 32768ppm
- Baselinazko zuzenketa automatikoa geruza oxido sentikor metalikoaren denbora: 24H
- Konfigurazioa baimendu eta sentsore bat irakurri boterearen ondoren: 20 minutu gutxienez.

2.3 TTGO Lora oled:

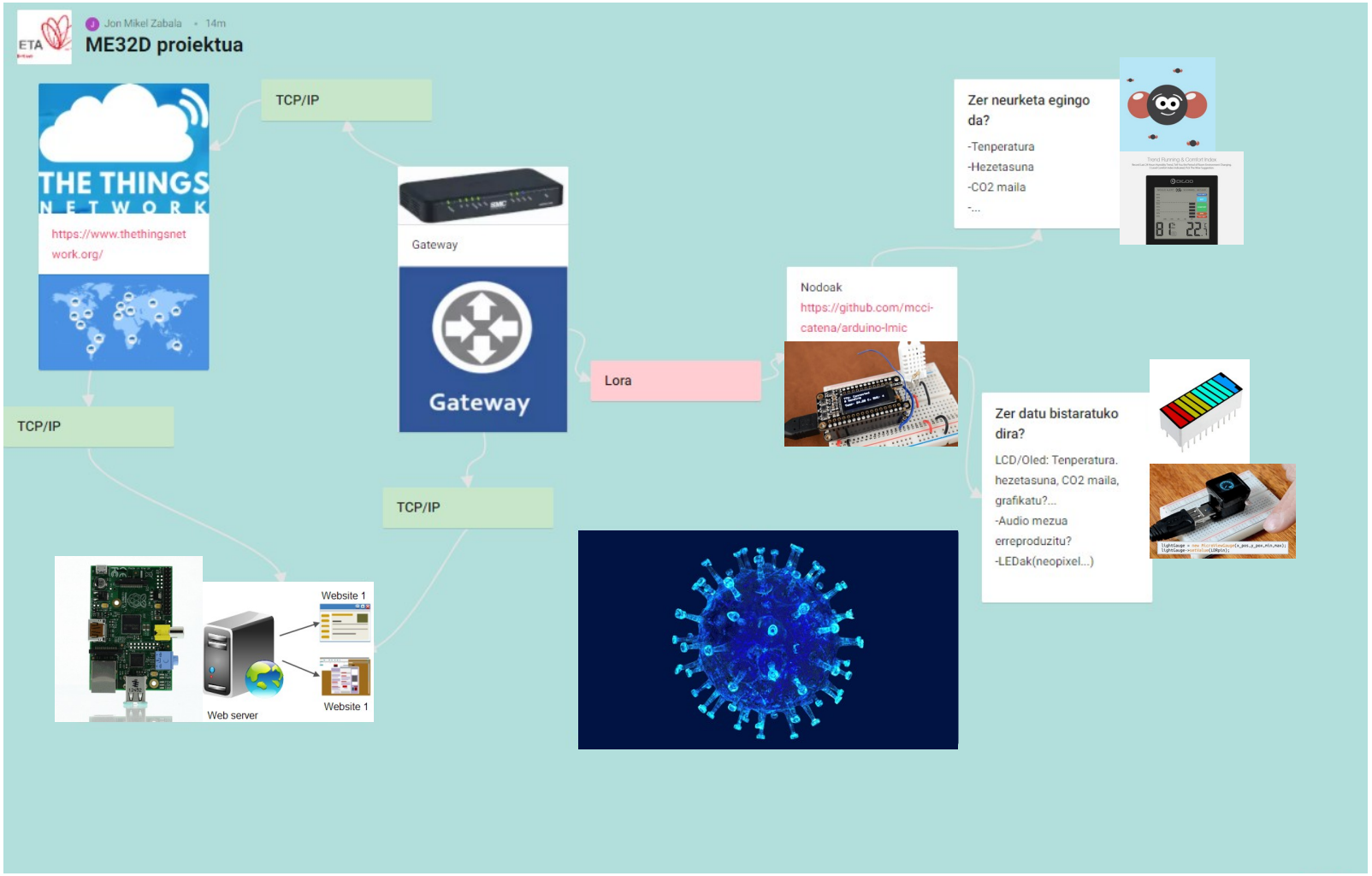
Wifi modulu ezagun hau bateragarria da Arduinorekin, eta hari gabeko konexioa eskaintzen du, modu guztiz errazean. ESP32 OLEDek bi gauzak plaka bakar eta sinple batean konbinatzen ditu: ESP32 modulua eta plaka berean dagoen OLED pantaila bat. Gainera, mikroUSB konektore bat dauka elikadurarako eta Arduinoko IDEtik zuzenean programatzeko. Plakak LoRa modulu bat ere badu, datuak distantzia handietara bi noranzkotan transmititzeko aukera ematen duena.



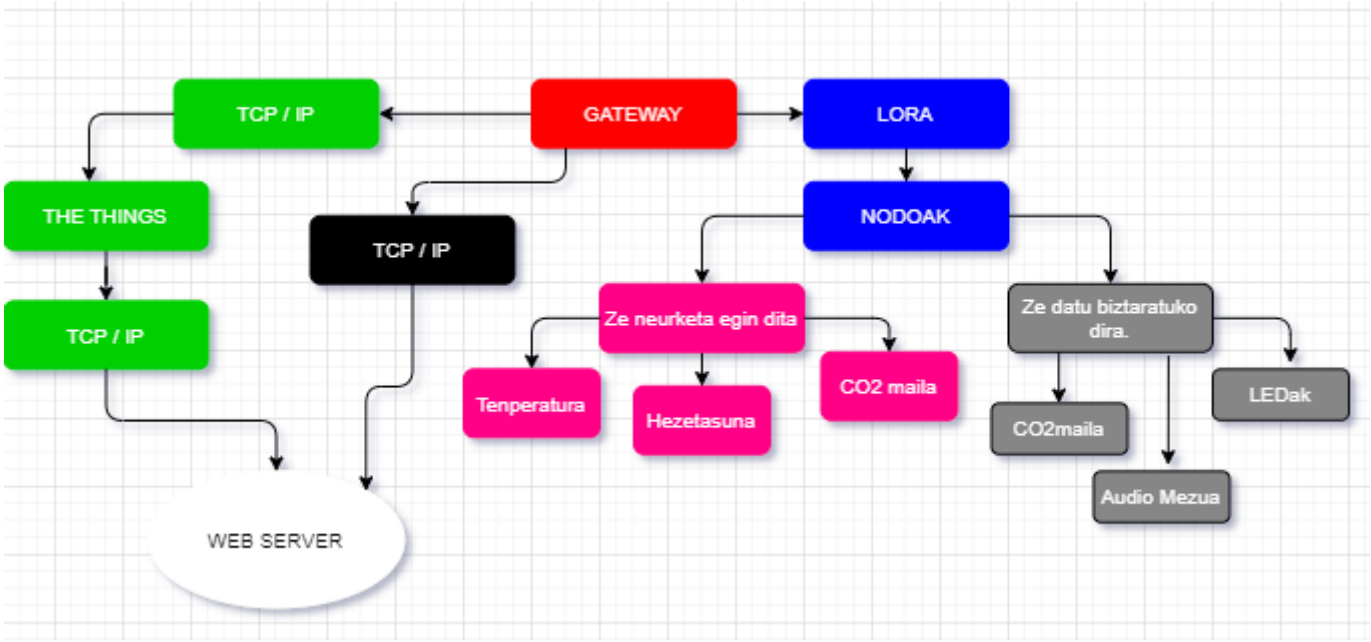
2.4 Hi-link (Elikatze iturria)

Hainbat arrazoirengatik elikatze iturri hau erabiltzea erabaki dugu. Batetik, korronte alternoko 220V-ko edozein entxufera konektatzeko aukera ematen digu. Bestetik, irteeran guk nahi dugun korronte zuzena lortzeko aukera daukagu, 3,3V. Gainera 0,9A ematen dizkigu irteeran, beraz, egokia da gure erronka aurrera atera ahal izateko.

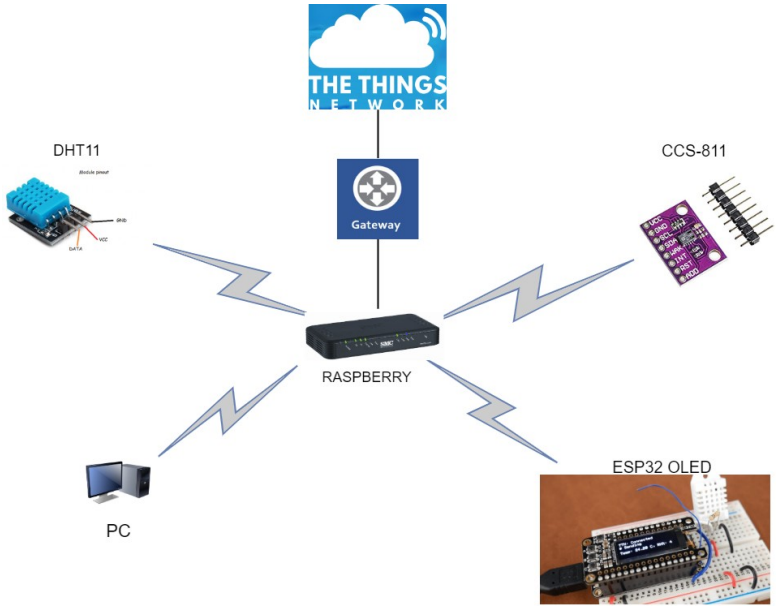
3 Bloke-diagramak



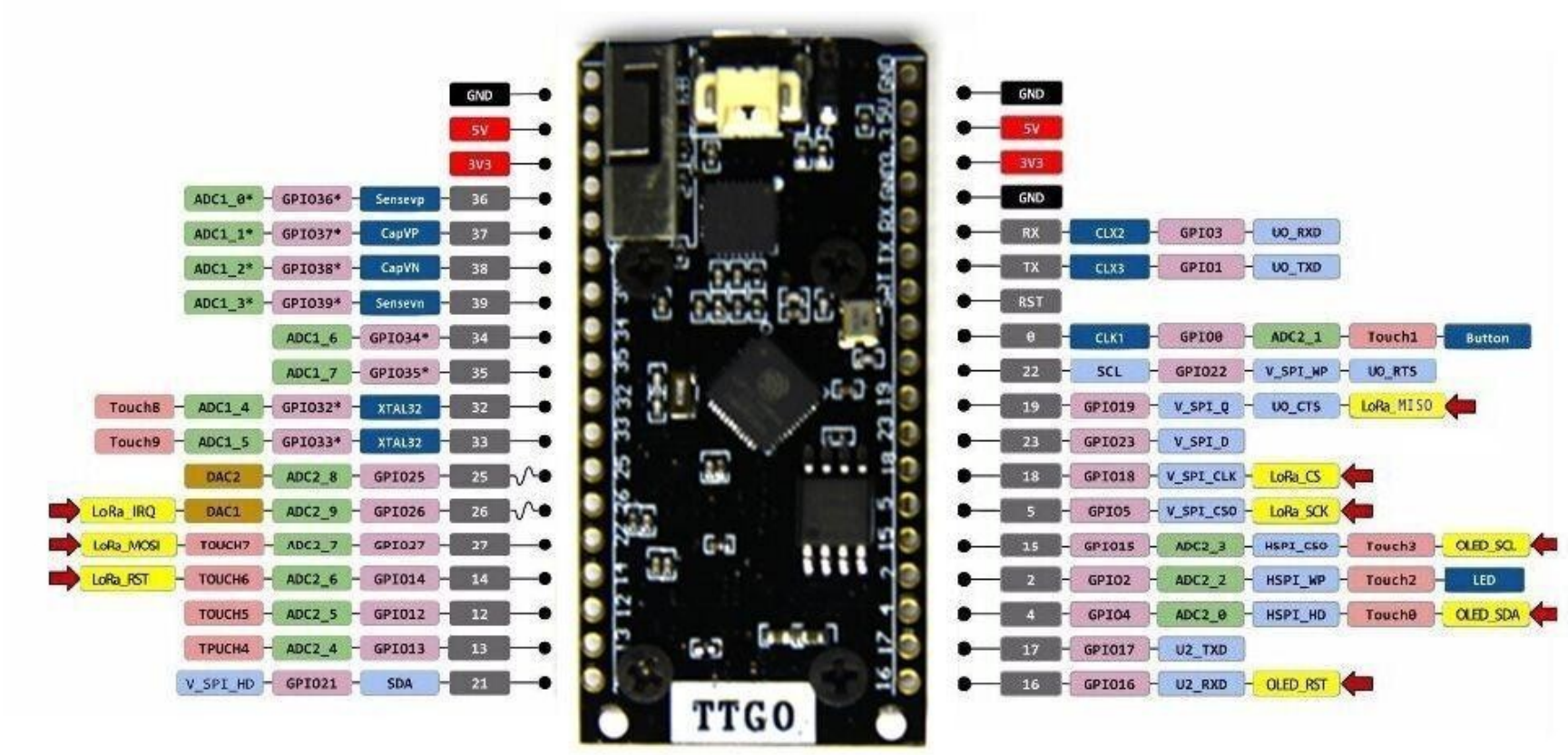
3.1 Softwarearen bloke-diagrama



3.2 Hardwarearen bloke-diagrama



4 ESP32 pinen esleipena



PLAKAKO ELIKADURA					KANPOKO ELIKADURA		ERABILERA
VIN	5V	3V3	GND	RESET	V	mA	
					3,3	20	ESP32
		X				2,5	DHT11
		X				1	CSS811
		X				460	NeoPixel
					220	100	Elikatze iturria

MEMORIAREN ERABILERA			
	FLASH (programa)	RAM (aldagaiak)	EEPROM (konfigurazioa/iraunkorra)
ERABILITAKOA			
HUTSIK			

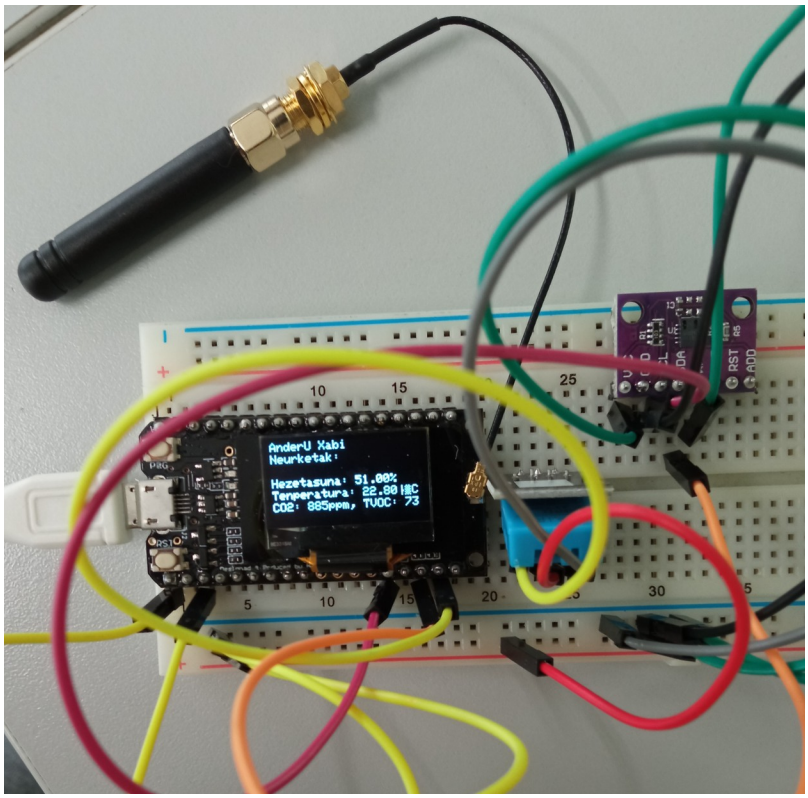
DIGITALA	KOMUNIKAZIOAK	GAINERAKOAK	ERABILERA
36	ADC1_0* / Sensevp		
37	ADC1_1* / CapVP		
38	ADC1_2* / CapVN		
39	ADC1_3* / Sensevn		
34	ADC1_6		
35	ADC1_7		
32	ADC1_4 / XTAL32		
33	ADC1_5 / XTAL32		
25	DAC2 / ADC2_8		
26	DAC1 / ADC2_9 / LoRa_ IRQ		
27	TPUCH7 / ADC2_7 / LoRa_ MOSI		
14	TPUCH6 / ADC2_6 / LoRa_ RST		
12	TPUCH5 / ADC2_5		
13	TPUCH4 / ADC2_4		NeoPixel Komunikazio pina
21	V_SPI_HD / SDA		
RX	CLX2 / UO_RDX		
TX	CLX3 / UO_TXD		
RST			
0	CLK1 / ADC2_1 / Button		
22	SCL / V_SPI_WP / UP_RTS		
19	V_SPI_Q / UO_CTS / LoRa_MISO		
23	V_SPI_D		
18	V_SPI_CLK / LoRa CS		
5	V_SPI_CSO / LoRa SCK		
15	HSPI_CSO / OLED_SCL		CCS811 SCL pina

DIGITALA	KOMUNIKAZIOAK	GAINERAKOAK	ERABILERA
2	HSPI_WP / LED		
4	HSPI_HD / OLED_SDA		CCS811 SDA pina
17	U2_TXD		DHT11 komunikazio pina
16	U2_RXD / OLED_RST		

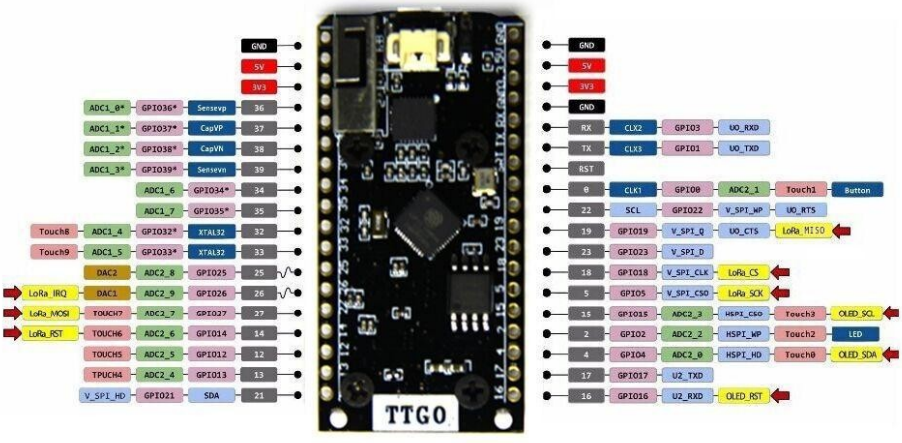
5 ESP32 TTGO LORA

5.1 Sarrera

ESP32-a gure erronkaren erdigunea izango da. Alde batetik sentsore ezberdinen neurketak jasoko ditu eta bestetik LORA bidez TTNra neurketa horiek bidaliko ditu. Horrez gain, pantaila bat dauka eta bertan balio ezberdinak erakutsiko ditugu, tenperatura, hezetazuna eta CO_2 maila.

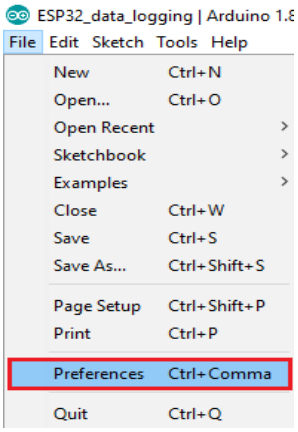


5.2 Sentsorearen edo shieldaren pinout-a

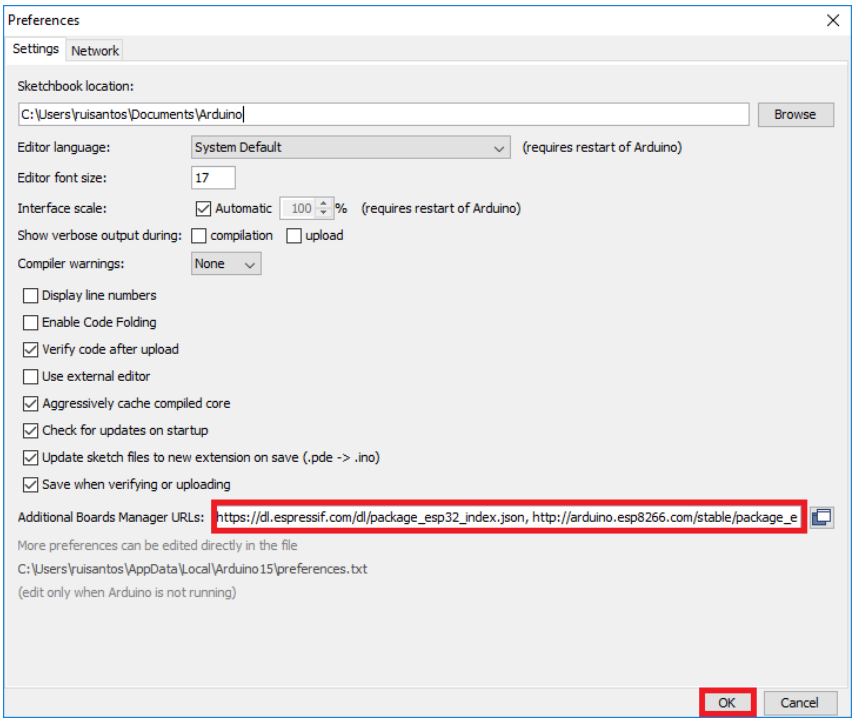


5.3 ESP32 arduinoan nola instalatu

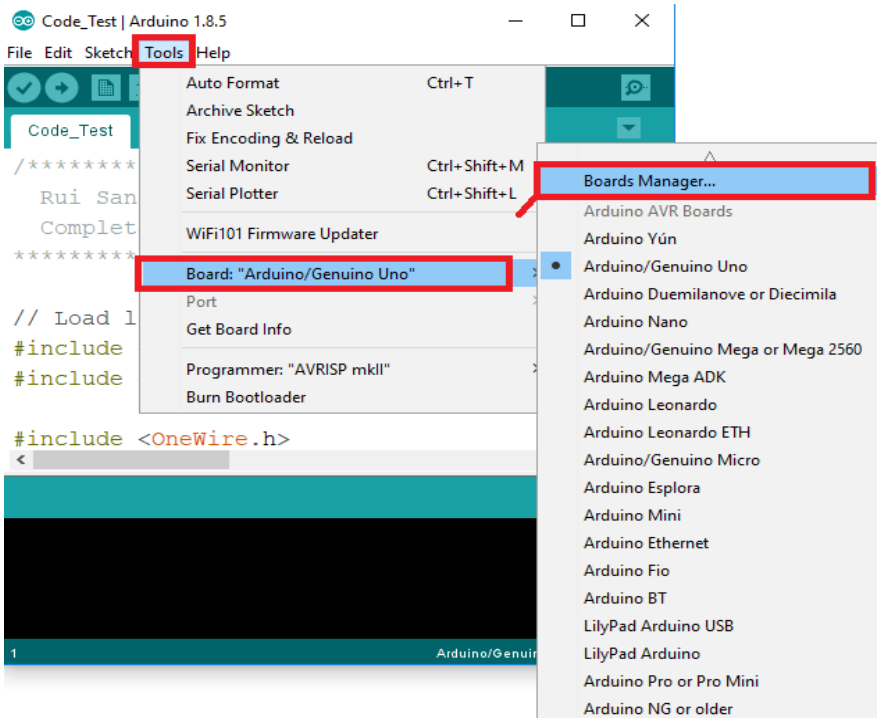
1. Arduinoko zure IDEan,joan Archivo > Preferencias



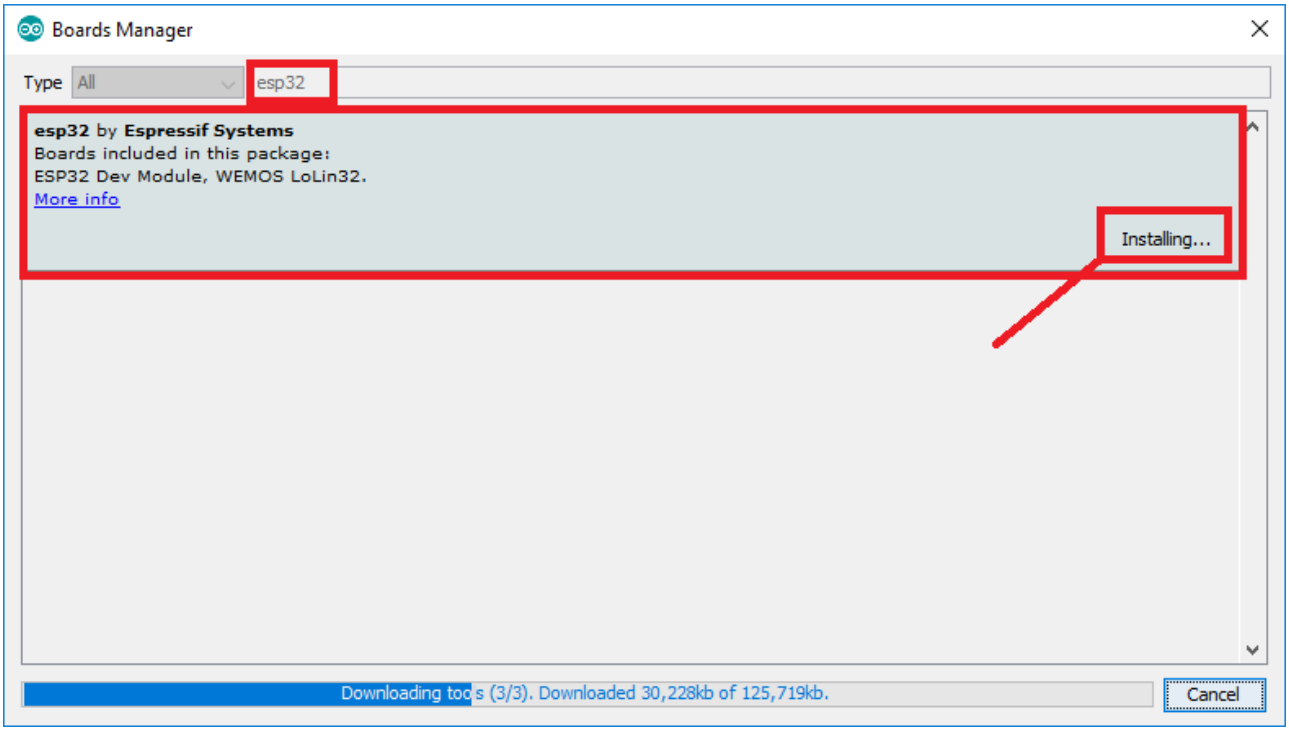
2. Sartu https://dl.espressif.com/dl/package_esp32_index.json "taula-administratzailearen URL gehigarriak" eremuan, hurrengo irudian agertzen den bezala. Gero, egin klik "onartu" botoian:



3. Ondoren “Herramientas” atalean, plaka jartzen duen eremuan, “Gestor de tarjetas”-ensartu.



4. Eta hor zaudenean ESP32 bilatu eta instalatu barik badago instalatu.10 minututan prest egongo da



5. Instalatu eta ondoren, plaka berri hori instalatuta egongo lirake.

Bertan utziko dizuet nik jarraitutako pusuak plaka hau instalatzeko :
https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/

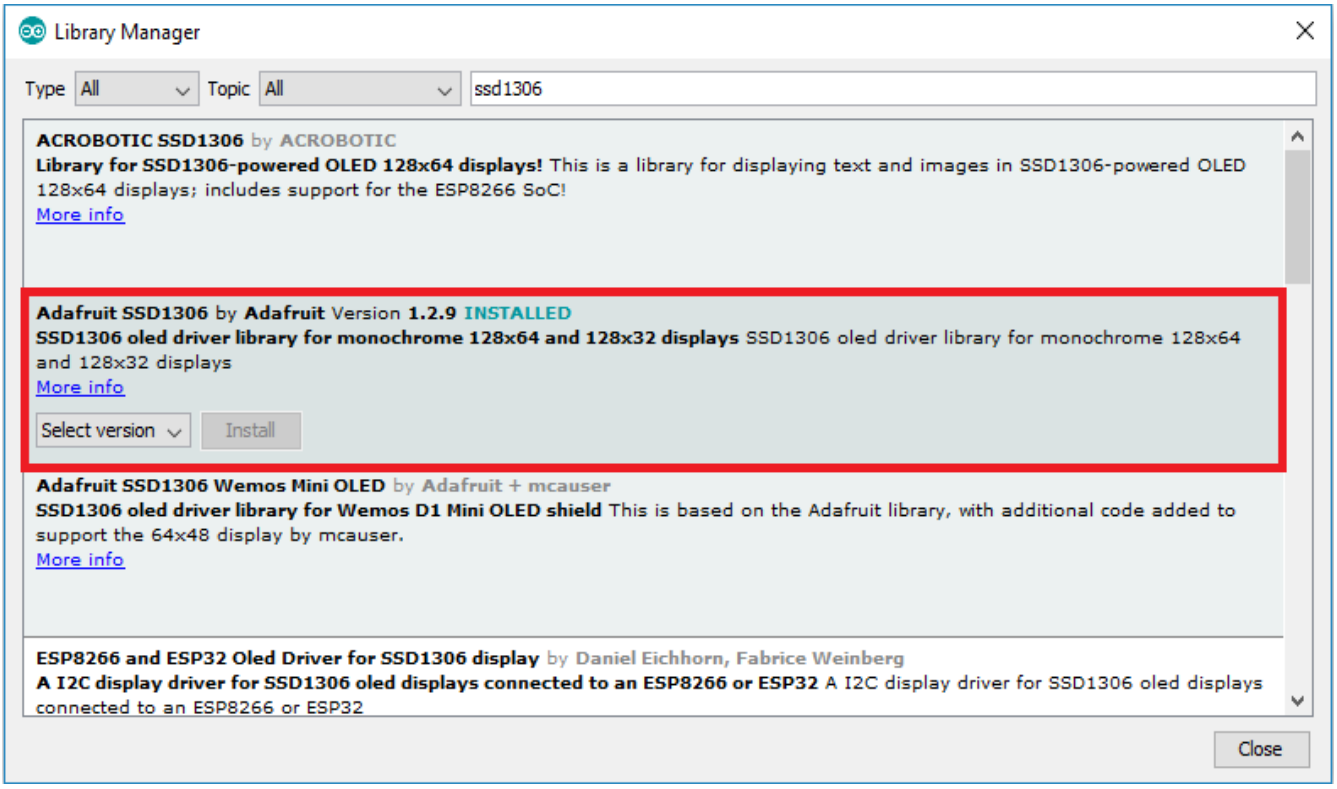
5.4 Oled liburutegien instalazioa

Liburutegi batzuk daude eskuragarri OLED pantaila ESP32 sistemarekin kontrolatzeko. Tutorial honetan Adafruit-eko bi liburutegi erabiliko ditugu: Adafruit_SSD1306 liburutegia eta Adafruit_GFX liburutegia.

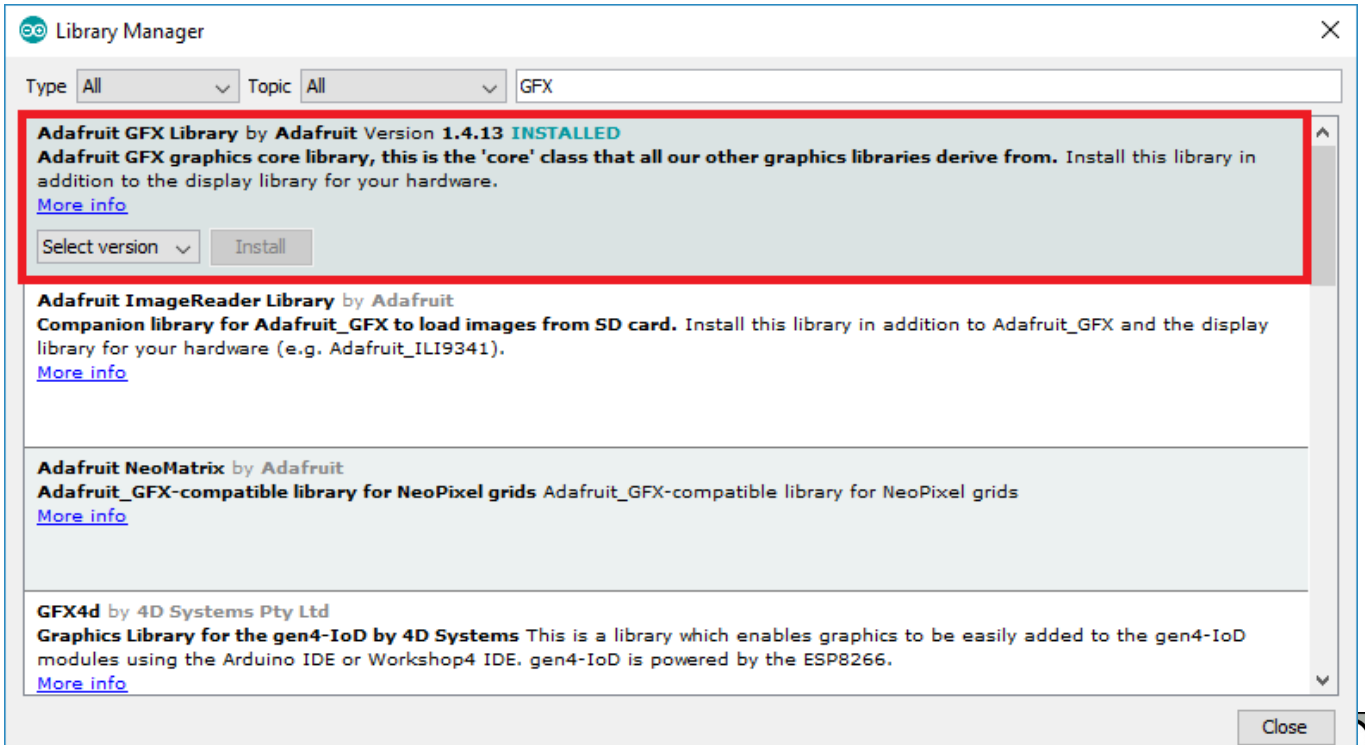
Jarraitu hurrengo urratsak liburutegi horiek instalatzeko.

1. Ireki ezazu Arduinoren IDE eta joan Sketch > Incluir biblioteca > Administrar bibliotecas. Liburutegiko administratzailea ireki beharko litzateke.

2. Idatzi "SSD1306" bilaketa-koadroan eta instalatu Adafruit-eko SSD1306 liburutegia.



3. Adafruit-eko SSD1306 liburutegia instalatu ondoren, idatzi "GFX" bilaketa-koadroan eta instalatu liburutegia.



5.5 Oinarrizko programa

Hartzailea

```
/*RECEPTOR*/
//Libraries for LoRa
#include <SPI.h>
#include <LoRa.h>

//Libraries for OLED Display
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//define the pins used by the LoRa transceiver module
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26

//433E6 for Asia
//866E6 for Europe
//915E6 for North America
#define BAND 866E6

//OLED pins
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

String LoRaData;

void setup() {

  //reset OLED display via software
  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  //initialize OLED
  Wire.begin(OLED_SDA, OLED_SCL);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C for
128x32
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }

  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(0,0);
  display.print("LORA RECEIVER ");
  display.display();
```

```
  //initialize Serial Monitor
  Serial.begin(115200);

  Serial.println("LoRa Receiver Test");

  //SPI LoRa pins
  SPI.begin(SCK, MISO, MOSI, SS);
  //setup LoRa transceiver module
  LoRa.setPins(SS, RST, DIO0);

  if (!LoRa.begin(BAND)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
  Serial.println("LoRa Initializing OK!");
  display.setCursor(0,10);
  display.println("LoRa Initializing OK!");
  display.display();
}

void loop() {

  //try to parse packet
  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    //received a packet
    Serial.print("Received packet ");

    //read packet
    while (LoRa.available()) {
      LoRaData = LoRa.readString();
      Serial.print(LoRaData);
    }

    //print RSSI of packet
    int rssi = LoRa.packetRssi();
    Serial.print(" with RSSI ");
    Serial.println(rssi);

   // Dsiplay information
   display.clearDisplay();
   display.setCursor(0,0);
   display.print("LORA RECEIVER");
   display.setCursor(0,20);
   display.print("Received packet:");
   display.setCursor(0,30);
   display.print(LoRaData);
   display.setCursor(0,40);
   display.print("RSSI:");
   display.setCursor(30,40);
   display.print(rssi);
   display.display();
  }
}
```

Igorlea

```
//EMISOR
//Libraries for LoRa
#include <SPI.h>
#include <LoRa.h>

//Libraries for OLED Display
#include <Wire.h>
#include <Adafruit_GFX.h>
```

```
#include <Adafruit_SSD1306.h>

//define the pins used by the LoRa transceiver module
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26

//433E6 for Asia
//866E6 for Europe
//915E6 for North America
#define BAND 866E6

//OLED pins
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

//packet counter
int counter = 0;

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

void setup() {

  //reset OLED display via software
  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  //initialize OLED
  Wire.begin(OLED_SDA, OLED_SCL);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C for
128x32
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }

  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(0,0);
  display.print("LORA SENDER ");
  display.display();

  //initialize Serial Monitor
  Serial.begin(115200);

  Serial.println("LoRa Sender Test");

  //SPI LoRa pins
  SPI.begin(SCK, MISO, MOSI, SS);
  //setup LoRa transceiver module
  LoRa.setPins(SS, RST, DIO0);

  if (!LoRa.begin(BAND)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
```

```
  Serial.println("LoRa Initializing OK!");
  display.setCursor(0,10);
  display.print("LoRa Initializing OK!");
  display.display();
  delay(2000);
}

void loop() {

  Serial.print("Sending packet: ");
  Serial.println(counter);

  //Send LoRa packet to receiver
  LoRa.beginPacket();
  LoRa.print("hello ");
  LoRa.print(counter);
  LoRa.endPacket();

  display.clearDisplay();
  display.setCursor(0,0);
  display.println("LORA SENDER");
  display.setCursor(0,20);
  display.setTextSize(1);
  display.print("LoRa packet sent.");
  display.setCursor(0,30);
  display.print("Counter:");
  display.setCursor(50,30);
  display.print(counter);
  display.display();

  counter++;

  delay(10000);
}
*/
```

5.6 Material zerrenda

Izena	Kopurua	Oharrak edota ezaugarri bereziak
ESP32 TTGO LORA	1	- Pantalla Oled - Antena de 2dBi SMA hariarekin.

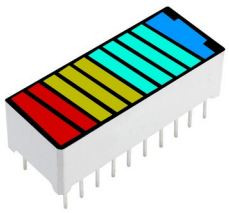
5.7 Gorabeherak

Ez dugu arazorik izan.

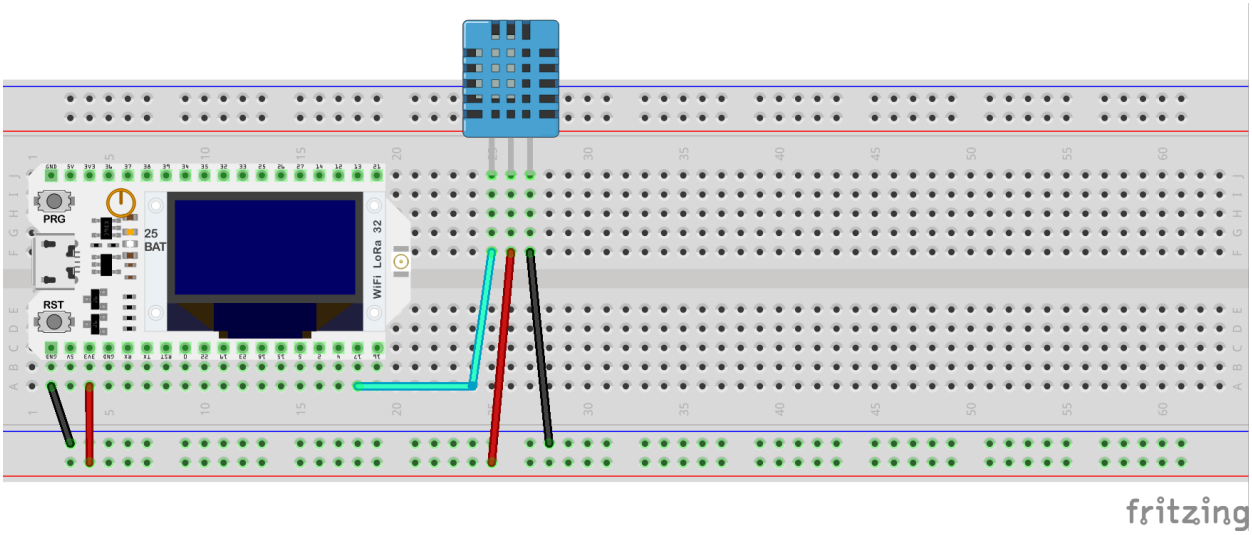
6. DHT11

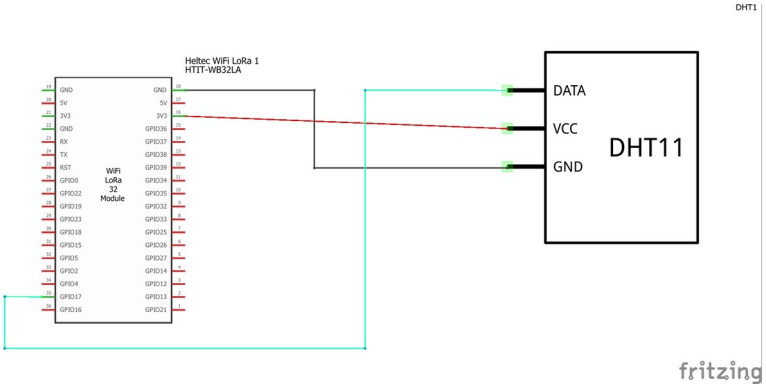
6.1 Sarrera

DHT11 tenperatura eta hezetasun erlatiboko sentsore digital bat da, kostu txikikoa eta erabilera errazekoa. Hezetasun-sentsore kapazitibo bat eta inguruko airea neurtzeko tenperatura sentsore bat integratzen ditu, eta datuak seinale digital baten bidez erakusten ditu datu-pinean.

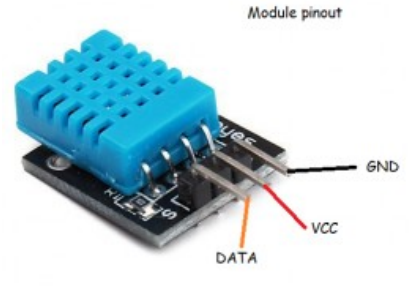


6.2 Eskema elektronikoak





6.3 Sentsorearen shieldaren pinout-a eta ezaugarriak



3.3V
Korronte maximoa ⟶ 2.5mA
Hezetasuna neurtzeko %20 - %90
Temperatura neurtzeko edukia ⟶ 0º - 5ºC

6.4 Oinarrizko programa

```
void DHT11sentsoreasetup (void){
    Serial.println(F("DHTxx test!"));

  dht.begin();
}
/* Funtzioa: void DHT11sentsorealoop (void)
 * Zeregina: funtzio honek tenperatura eta hezetasuna irakurtzen du
 * Erabilitako liburutegiak: - DHT sentsoreen liburutegia: https://github.com/adafruit/
DHT-sensor-library
                            - Adafruit Unified sensor Lib:
https://github.com/adafruit/Adafruit sensor
 * Liburutegia nondik hartuta: Arduino bertatik
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 *  */
void DHT11sentsorealoop (void){
    // Itxaron neurketa egin baino lehen
  delay(2000);

  float h = dht.readHumidity();
  // Irakurri tenperatura Celsius
  float t = dht.readTemperature();
  // Irakurri tenperatura Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  // Egiaztatu irakurketaren batek huts egin duen eta goiz atera (berriro saiatzeko).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Ezin izan da DHT sentsoretik irakurri!"));
    return;
  }
  // Kalkulatu bero-indizea Fahrenheiten (balio lehenetsia)
  float hif = dht.computeHeatIndex(f, h);
  // Kalkulatu bero-indizea Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Hezetasuna: "));
  Serial.print(h);
  Serial.print(F("%  Tenperatura: "));
  Serial.print(t);
  Serial.print(F("Â°C "));
  Serial.print(f);
```

```
  Serial.print(F("Â°F  Sentzasio termikoa: "));
  Serial.print(hic);
  Serial.print(F("Â°C "));
  Serial.print(hif);
  Serial.println(F("Â°F"));
}
```

6.5 Oinarrizko programa funtzioak bihurtuta

Lehen orrialdea

```
// Liburutegi hauek behar ditu Arduinok:
// - DHT sentsoreen liburutegia: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified sensor Lib: https://github.com/adafruit/Adafruit sensor

#include "DHT.h"


#define DHTPIN 17 // PIN digitala DHT sentsorera konektatuta

// DHT mota
#define DHTTYPE DHT11 // DHT 11

DHT dht(DHTPIN, DHTTYPE);

void DHT11sentsorealoop (void);
void DHT11sentsoreasetup (void);

//abiarazi DHT sentsorea


void setup() {
 Serial.begin(9600);
 DHT11sentsoreasetup();
}
void loop() {
DHT11sentsorealoop();
}
```

Bigarren horrialdea

```
void DHT11sentsoreasetup (void){
    Serial.println(F("DHTxx test!"));

  dht.begin();
}
/* Funtzioa: void DHT11sentsorealoop (void)
 * Zeregina: funtzio honek tenperatura eta hezetasuna irakurtzen du
 * Erabilitako liburutegiak: - DHT sentsoreen liburutegia: https://github.com/adafruit/DHT-sensor-library
                             - Adafruit Unified sensor Lib: https://github.com/adafruit/Adafruit sensor
 * Liburutegia nondik hartuta: Arduino bertatik
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 *  */
void DHT11sentsorealoop (void){
    // Itxaron neurketa egin baino lehen
  delay(2000);

  float h = dht.readHumidity();
  // Irakurri tenperatura Celsius
  float t = dht.readTemperature();
```

```
  // Irakurri tenperatura Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  // Egiaztatu irakurketaren batek huts egin duen eta goiz atera (berriro saiatzeko).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Ezin izan da DHT sentsoretik irakurri!"));
    return;
  }
  // Kalkulatu bero-indizea Fahrenheiten (balio lehenetsia)
  float hif = dht.computeHeatIndex(f, h);
  // Kalkulatu bero-indizea Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Hezetasuna: "));
  Serial.print(h);
  Serial.print(F("%  Tenperatura: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F  Sentzasio termikoa: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
}
```

6.6 Material zerrenda

Bill of Materials: DHT11.fzz

T:\AnderU_Xabi\LORA\FRITZIN ESKEMAK\DHT11\DHT11.fzz
jueves, marzo 4 2021, 14:00:14

Assembly List

Label	Part Type	Properties
DHT1	DHT11 Humitidy and Temperature Sensor (3 pins)	Variante variant 1
Heltec WiFi LoRa 1	Heltec WiFi LoRa 32	module WiFi LoRa 32 V1.0; Alfileres 36; Variante variant 1; Número de componente HTIT-WB32LA

Shopping List

Amount	Part Type	Properties
1	DHT11 Humitidy and Temperature Sensor (3 pins)	Variante variant 1
1	Heltec WiFi LoRa 32	module WiFi LoRa 32 V1.0; Alfileres 36; Variante variant 1; Número de componente HTIT-WB32LA

Exported with Fritzing 0.9.3- http://fritzing.org

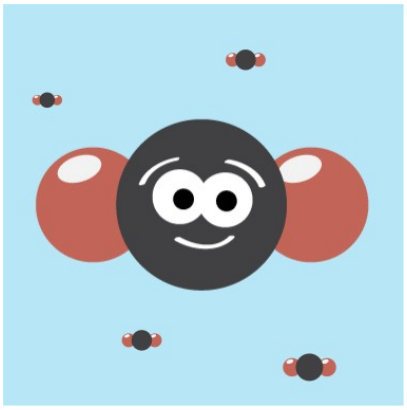
6.7 Gorabeherak

Arazorik ez.

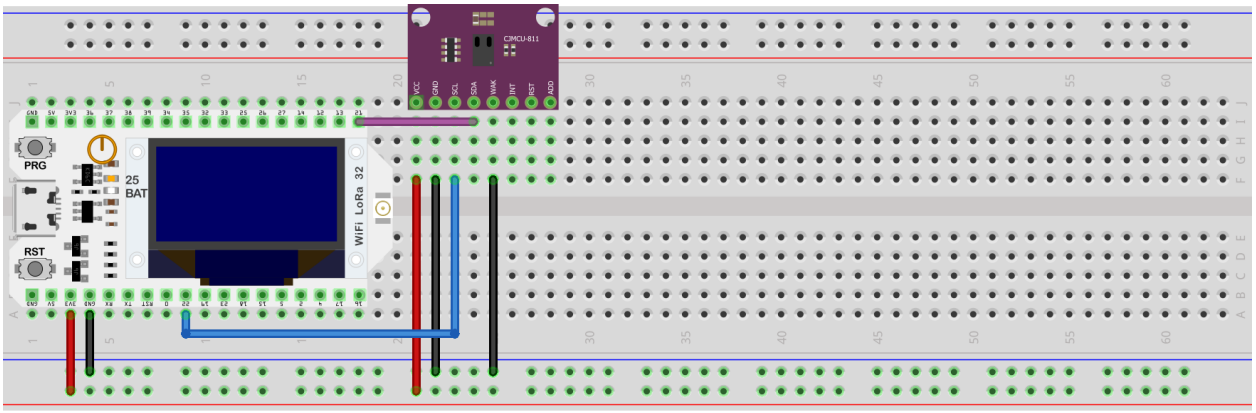
7 CJMCU-811 (CCS811)

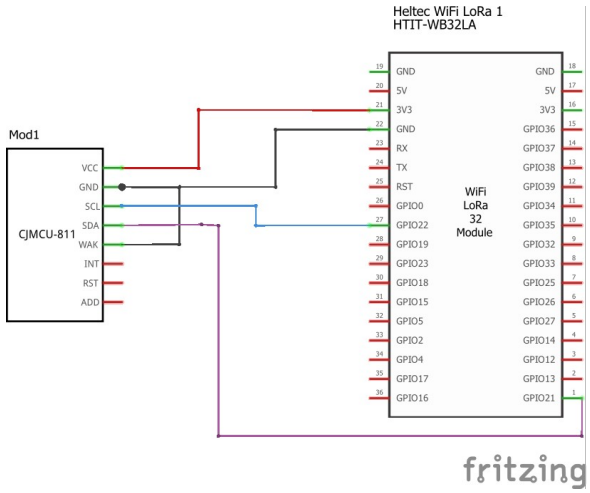
7.1 Sarrera

CCS811 sentsorea gas digitaleko sentsore bat da, eta konposatu organiko lurrunkor guztien gama zabala (TVOC) detektatzen du, karbono dioxidoaren (eCO2) eta metal oxidoaren (MOX) maila baliokideak barne.

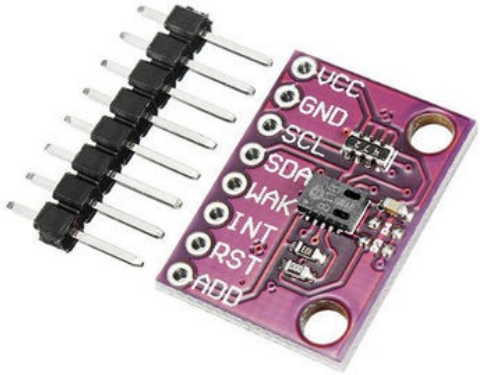


7.2 Eskema elektronikoak





7.3 Sentsorearen edo shieldaren pinout-a



7.4 Oinarrizko programa

```
void CCS811sentsoreasetup (void){
 Serial.println("CCS811 test");

 if(!ccs.begin()){
 Serial.println("Failed to start sensor! Please check your wiring.");
 while(1);
 }

 // Wait for the sensor to be ready
 while(!ccs.available());
}

/* Funtzioa: void CCS811sentsorealoop (void)
 * Zeregina: funtzio honek CO2 maila irakurtzen du
 * Erabilitako liburutegiak: "Adafruit_CCS811.h"
 * Liburutegia nondik hartuta: Arduino bertatik
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */
 void CCS811sentsorealoop (void){
 if(ccs.available()){
 if(!ccs.readData()){
 Serial.print("CO2: ");
 Serial.print(ccs.geteCO2());
 Serial.print("ppm, TVOC: ");
 Serial.println(ccs.getTVOC());
 }
 else{
 Serial.println("ERROR!");
 while(1);
 }
 }
 delay(500);
 }
```

7.5 Oinarrizko programa funtzioak bihurtuta

Lehen orrialdea

```
#include <lmic.h>
#include <hal/hal.h>
```

```
#include <SPI.h>.
#include "Adafruit_CCS811.h"

Adafruit_CCS811 ccs;

void CCS811sentsorealoop (void);
void CCS811sentsoreasetup (void);

void setup() {
 Serial.begin(9600);
 CCS811sentsoreasetup();
}

void loop() {
 CCS811sentsorealoop();
}
```

Bigarren orrialdea

```
void CCS811sentsoreasetup (void){
    Serial.println("CCS811 test");

  if(!ccs.begin()){
    Serial.println("Failed to start sensor! Please check your wiring.");
    while(1);
  }

  // Wait for the sensor to be ready
  while(!ccs.available());
}

/* Funtzioa: void CCS811sentsorealoop (void)
 * Zeregina: funtzio honek CO2 maila irakurtzen du
 * Erabilitako liburutegiak: "Adafruit_CCS811.h"
 * Liburutegia nondik hartuta: Arduino bertatik
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 *  */
 void CCS811sentsorealoop (void){
    if(ccs.available()){
    if(!ccs.readData()){
      Serial.print("CO2: ");
      Serial.print(ccs.geteCO2());
      Serial.print("ppm, TVOC: ");
      Serial.println(ccs.getTVOC());
    }
    else{
      Serial.println("ERROR!");
      while(1);
    }
  }
  delay(500);
 }
```

7.6 Material zerrenda

Bill of Materials: CCS811.fzz

T:\AnderU_Xabi\LORA\FRITZIN ESKEMAK\CCS811\CCS811.fzz
jueves, febrero 25 2021, 08:42:14

Assembly List

Label	Part Type	Properties
Heltec WiFi LoRa 1	Heltec WiFi LoRa 32	module WiFi LoRa 32 V1.0; Alfileres 36; Variante variant 1; Número de componente HTIT-WB32LA
Mod1	CJMCU-811	Variante Variant 1

Shopping List

Amount	Part Type	Properties
1	Heltec WiFi LoRa 32	module WiFi LoRa 32 V1.0; Alfileres 36; Variante variant 1; Número de componente HTIT-WB32LA
1	CJMCU-811	Variante Variant 1

Exported with Fritzing 0.9.3- http://fritzing.org

7.7 Gorabeherak

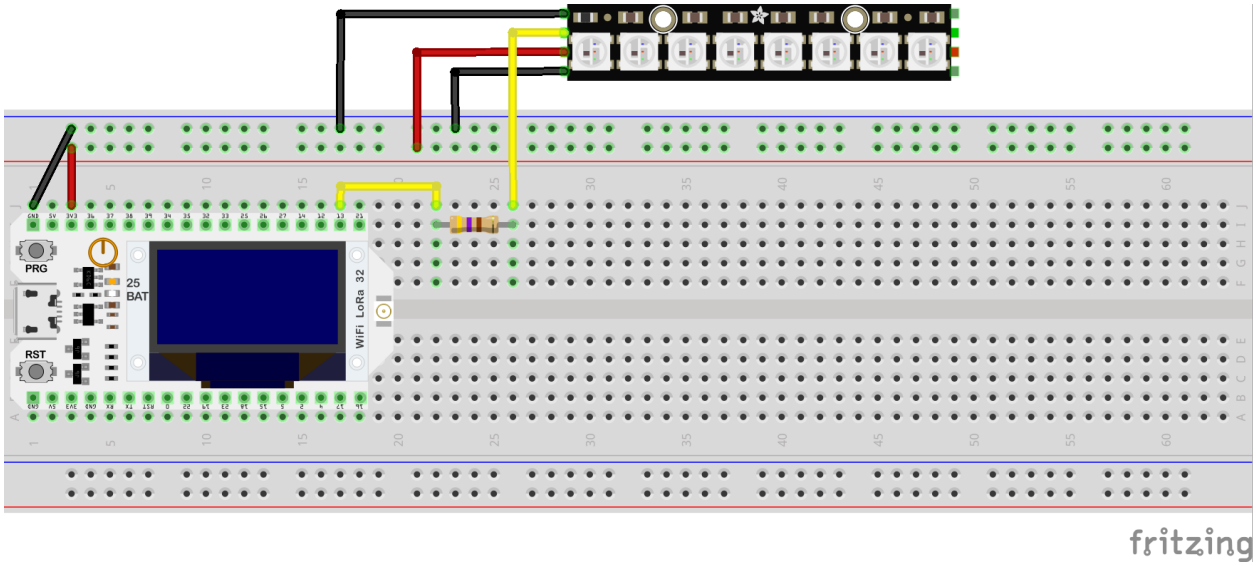
Wak pina hasieran ez genekien GNDra konektatu behar genuenik.

8 NeoPixel

8.1 Sarrera

Neopixel, Adafruit Industriesek sortutako marka bat da, banaka zuzendu daitezkeen RGB LEDak aipatzeko, hau da, euren barnean zirkuitu logiko integratu bat duten LEDak, kateatutako LEDen sekuentzia batean LED bakoitzaren kolorea pin digital bakar batekin kontrolatzea ahalbidetzen duen zirkuitua. Banaka zuzendu daitezkeen LED guztiak ez dira NeoPixel-ak.

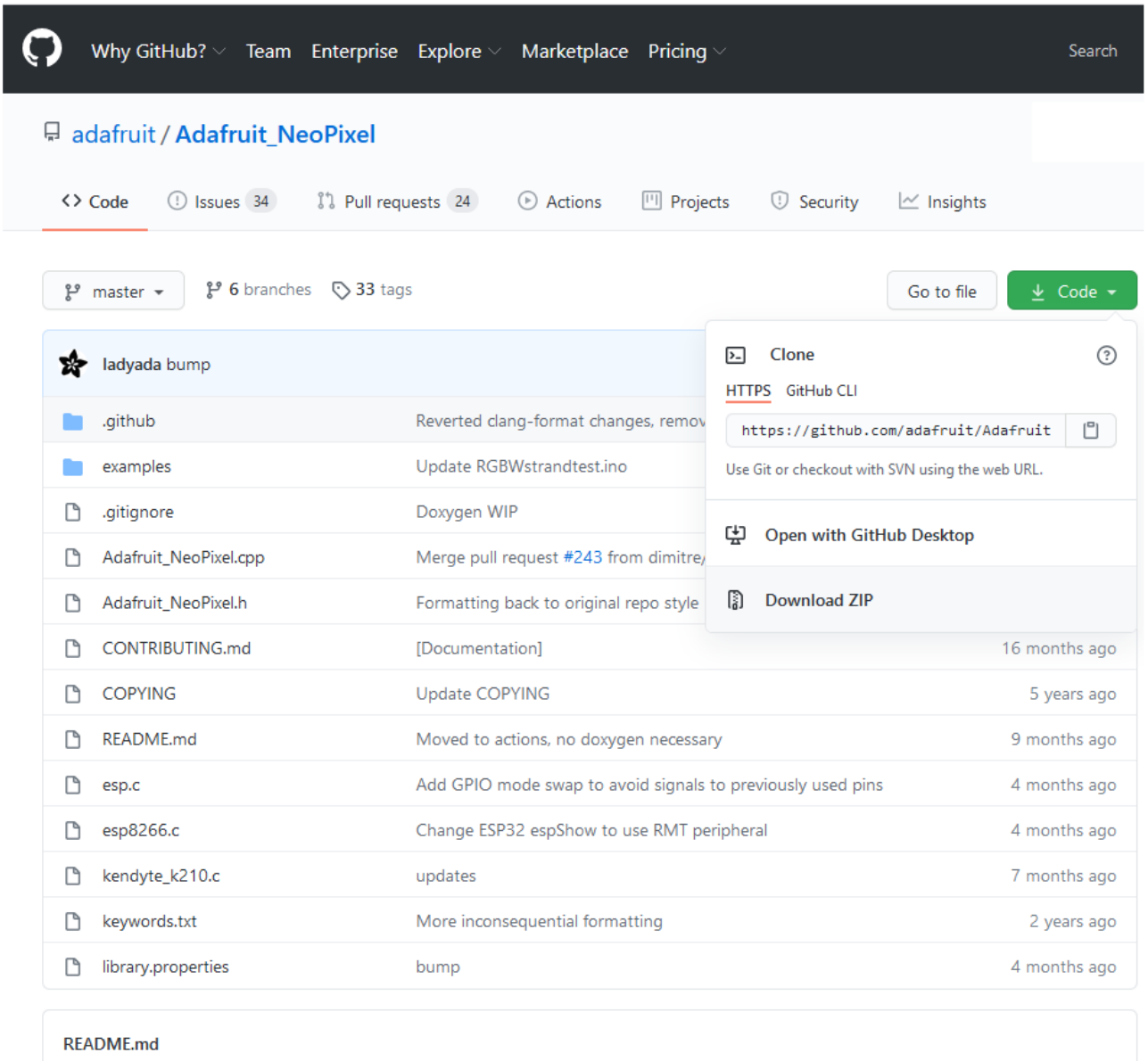
8.2 Eskema elektronikoak



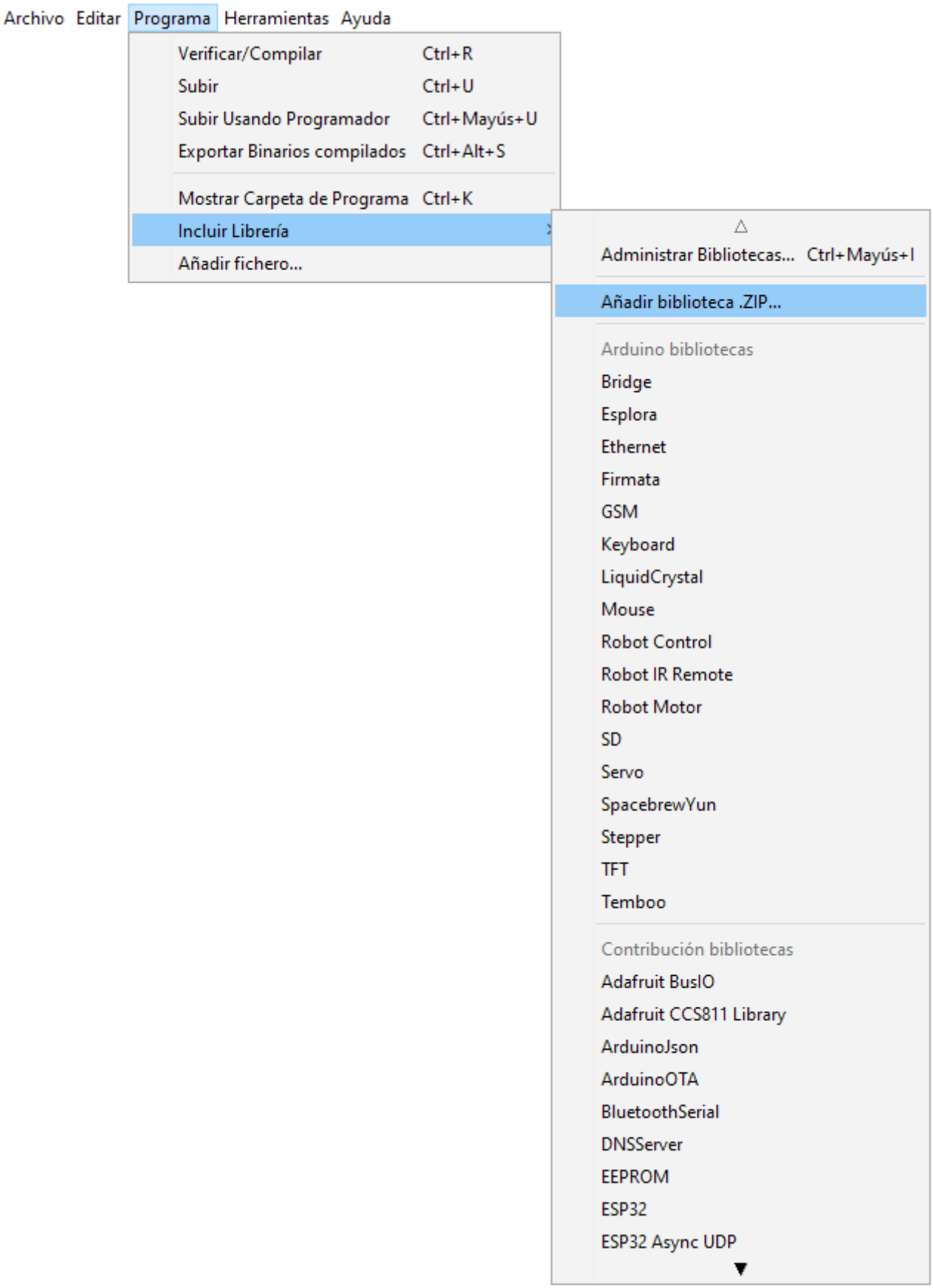
8.3 Sentsorearen edo shieldaren pinout-a

8.4 Liburutegia instalatu

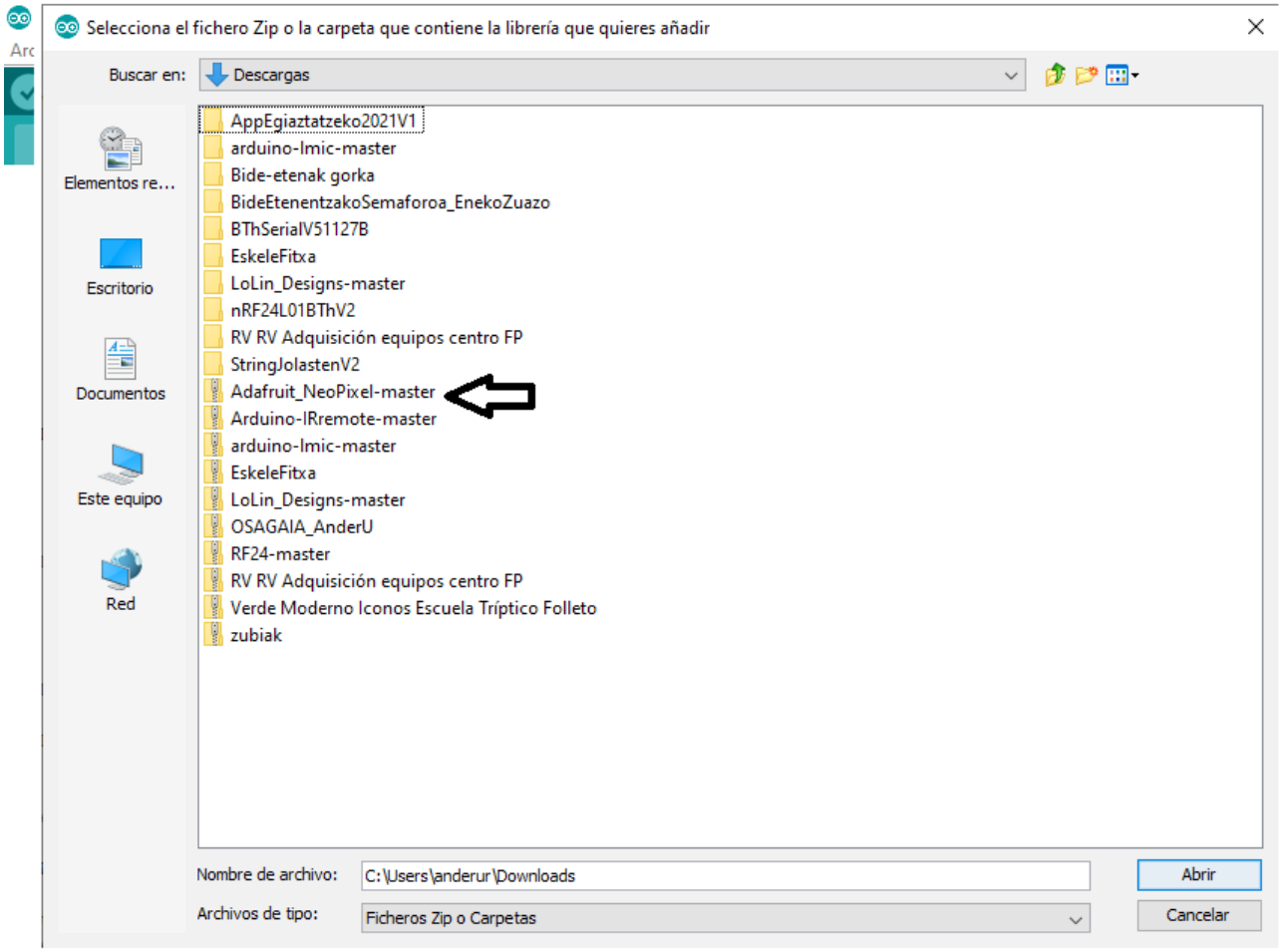
Liburutegia GitHub-etik deskargatu 7zip modura



Gehitu zip liburutegia



Aukeratu gehitu nahi duguna, aurretik jeitsitakoa.



8.5 Oinarrizko programa

```
// NeoPixel Ring simple sketch (c) 2013 Shae Erisson
// Released under the GPLv3 license to match the rest of the
// Adafruit NeoPixel library

#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
 #include <avr/power.h> // Required for 16 MHz Adafruit Trinket
#endif

// Which pin on the Arduino is connected to the NeoPixels?
#define PIN        13 // On Trinket or Gemma, suggest changing this to 1

// How many NeoPixels are attached to the Arduino?
#define NUMPIXELS 8 // Popular NeoPixel ring size

// When setting up the NeoPixel library, we tell it how many pixels,
// and which pin to use to send signals. Note that for older NeoPixel
// strips you might need to change the third parameter -- see the
// strandtest example for more information on possible values.
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

#define DELAYVAL 500 // Time (in milliseconds) to pause between pixels

void setup() {
  // These lines are specifically to support the Adafruit Trinket 5V 16 MHz.
  // Any other board, you can remove this part (but no harm leaving it):
#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
  clock_prescale_set(clock_div_1);
#endif
  // END of Trinket-specific code.
```

```
  pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)
}

void loop() {
  pixels.clear(); // Set all pixel colors to 'off'

    // pixels.Color() takes RGB values, from 0,0,0 up to 255,255,255
    // Here we're using a moderately bright green color:
    pixels.setPixelColor(0, pixels.Color(1, 1, 1));
    pixels.setPixelColor(1, pixels.Color(20, 0, 0));
    pixels.setPixelColor(2, pixels.Color(0, 20, 0));
    pixels.setPixelColor(3, pixels.Color(0, 0, 20));
    pixels.setPixelColor(4, pixels.Color(20, 20, 0));
    pixels.setPixelColor(5, pixels.Color(20, 0, 20));
    pixels.setPixelColor(6, pixels.Color(0, 20, 20));
    pixels.setPixelColor(7, pixels.Color(5, 5, 5));

    pixels.show();   // Send the updated pixel colors to the hardware.

    delay(DELAYVAL); // Pause before next pass through loop
  }
```

8.6 Material zerrenda

Bill of Materials: NeoPixel.fzz

T:/AnderU_Xabi/LORA/FRITZIN ESKEMAK/NeoPixel.fzz
jueves, marzo 4 2021, 12:42:47

Assembly List

Label	Part Type	Properties
Componente2	NeoPixel Stick	Variante variant 2; Número de componente 1426
Heltec WiFi LoRa 3	Heltec WiFi LoRa 32	Variante variant 1; Alfileres 36; module WiFi LoRa 32 V1.0; Número de componente HTIT-WB32LA
R3	470Ω Resistor	Resistencia 470Ω; paquete 0805 [SMD]; tolerancia ±5%

Shopping List

Amount	Part Type	Properties
1	NeoPixel Stick	Variante variant 2; Número de componente 1426
1	Heltec WiFi LoRa 32	Variante variant 1; Alfileres 36; module WiFi LoRa 32 V1.0; Número de componente HTIT-WB32LA
1	470Ω Resistor	Resistencia 470Ω; paquete 0805 [SMD]; tolerancia ±5%

Exported with Fritzing 0.9.3- http://fritzing.org

3.0.1 8.7 Gorabeherak

Ez dugu gorabeherarik eduki.

9. HLK-PM03 Elikatze iturria

9.1 Sarrera

HLK-PM03 AC DC bihurgailua elikadura-iturri deitzen den gailu elektroniko bat da. Elektronikan, korronte alternoa irteera batean edo gehiagotan korronte zuzen bihurtzen du.

HLK-PM03 AC DC bihurgailua espazio trinkoetan inplementatu daiteke. Hau da elikatze iturri horren itxura

Energia elektrikoa VCA izatetik VCD izatera aldatzean, hainbat gailu elektroniko elikatu ditzake, Adib. CD motorra, led-ak, wifi modulua, sentsoreak, etab.

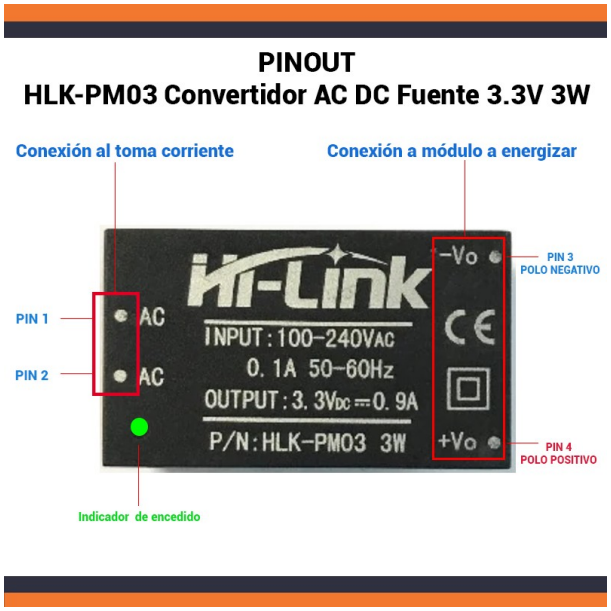
9.2 Zehaztasun teknikoak

Tentsioa sartzea (AC: 90 ~ 264 V).
Zarata-maila baxua
Gainkargaren eta zirkuitulaburren aurkako babesa.
Eraginkortasun handia, potentzia-dentsitate handia
Produktua EMCren eta segurtasun-probaren baldintzak betetzeko diseinatuta dago.
Energia-kontsumo txikia, ingurumenaren babesa, karga-galerarik gabe < 0,1 W.

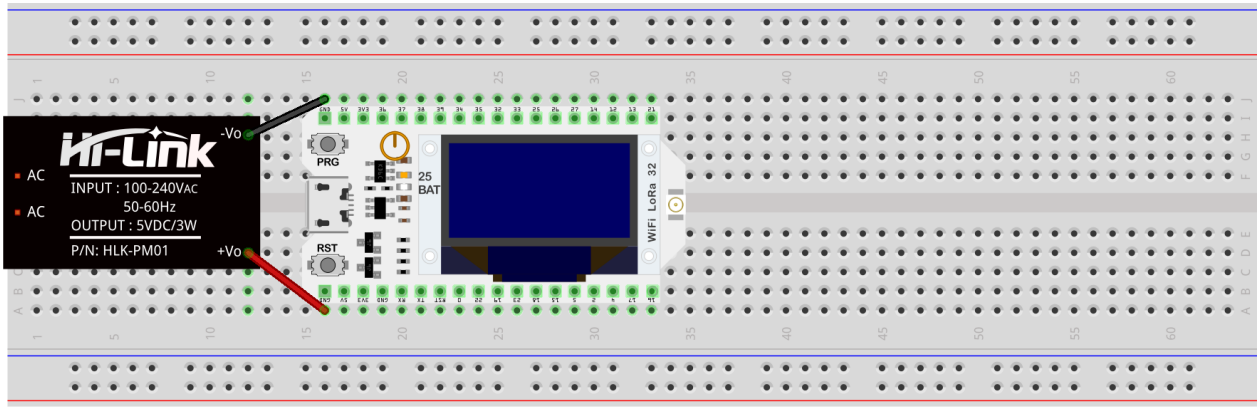
Segurtasun-ezaugarriak
PCB plaka, kobrezko aurpegi bikoitzarekin, 94-V0 suteen sailkapen-mailarako materiala.
Segurtasun-arauak: UL1012, EN60950, UL60950 betetzen ditu.
Gidatzea eta erradiazioa: Espainiako Konstituzioaren segurtasun-araudiaren araua eta 22. artikulua betetzen ditu.

Tenperaturaren segurtasunaren diseinua.
Giro-tenperaturan, potentzia horren kondentsadoreek bihurgailu nagusiaren barne-azalera, gehieneko tenperaturakoa, ez da 90 ° C-tik gorakoa.
Karkasaren gainazaleko tenperatura maximoa ez da 60 ° C-tik gorakoa.

9.3 Pinout



9.4 Eskema elektrikoa



9.5 Estekak

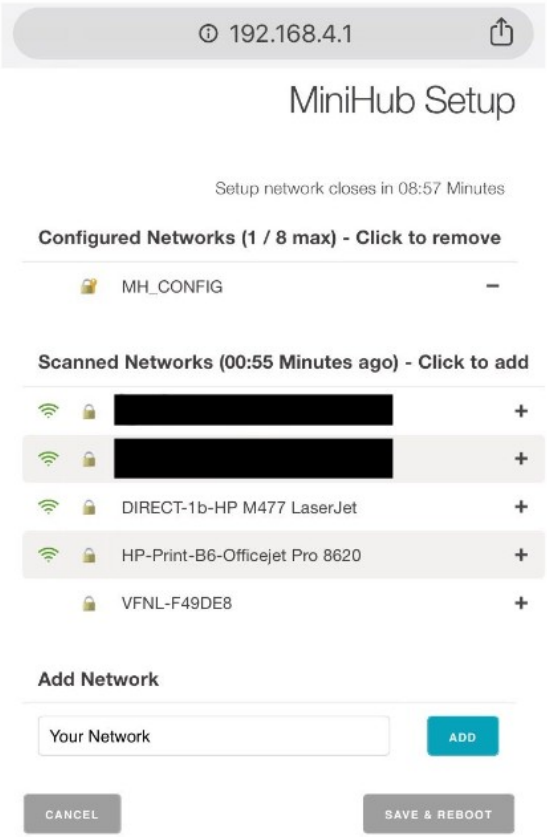
Datasheet: https://www.mikrocontroller.net/attachment/349613/HLK-PM03.pdf

10 GateWay konfigurazioa

GateWay bat LoRa transmisio modulua duen gailua da eta informazioa Internet eta berarekin komunikatzen diren nodoen artean birbidaltzen du eta alderantziz.

GateWay-a konfiguratzeko tutorial edo manual bat aurkitu dugu interneten.
Hainbat pasu segi ditugu, honako hauek:

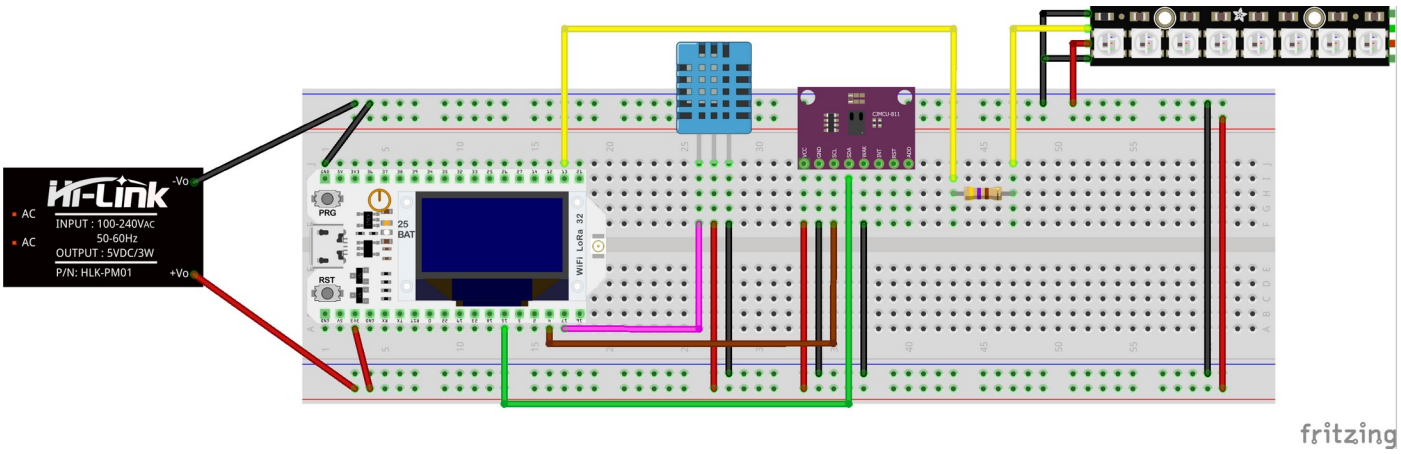
- Lehenengo pausua, gure GateWay-a elikatu ondoren Reset botoia 5 segunduz pultsatzea da, goiko aldean duen LEDa azkar parpadeatzen(berde-gorri) jarri arte
- Ondoren Setup botoia 10 segunduz pultsauta mantendu LEDa gorriz parpadeatzen jarri harte.
- Hau lortu eta gero GateWay-a orain WiFi AP bat erakusten du, SSID MINIHUB-xxxxxx dena, non xxxxxx GateWay-aren IDA da, 6 digito.
- GateWay-aren pasahitza gailuaren atzeko aldean ikus dezakegu.
- Azkenik, 192.168.4.1 sarrera sartu eta hor Wifia aukeratu dugu



- Konfigurazioa zuzena bada, Gateway-a berdez kliskatuko du segundo batzuetan, sare horretara konektatzen den bitartean.
- Konfigurazioa zuzena bada, pasabideak BERDE < ->GORRIAN kliskatuko du segundo batzuetan, CUPS azken puntura konektatzen den bitartean, eta LNS trafikoko azken puntura konektatzeko beharrezkoa den informazioa lortzen du.

- Konfigurazioa zuzena izan bada, Led berde finkoan egongo da, eta horrek esan nahi du GateWay-a konektatuta dagoela.

11 Fritzing eskema osoa



Material zerrenda

Bill of Materials: EskemaOsoaErronka.fzz

T:\AnderU_Xabi\LORA\FRITZIN ESKEMAK\EskemaOsoaErronka.fzz
jueves, marzo 4 2021, 12:46:20

Assembly List

Label	Part Type	Properties
Componente1	NeoPixel Stick	Variante variant 2; Número de componente 1426
DHT2	DHT11 Humitidy and Temperature Sensor (3 pins)	Variante variant 1
Heltec WiFi LoRa 2	Heltec WiFi LoRa 32	Variante variant 1; module WiFi LoRa 32 V1.0; Alfileres 36; Número de componente HTIT-WB32LA
HLK-PM2	HLK-PM01	Variante variant 1; Alfileres 4; editable pin labels false; label HLK-PM01
Mod2	CJMCU-811	Variante Variant 1
R2	470Ω Resistor	paquete 0805 [SMD]; Resistencia 470Ω; tolerancia ±5%

Shopping List

Amount	Part Type	Properties
1	NeoPixel Stick	Variante variant 2; Número de componente 1426
1	DHT11 Humitidy and Temperature Sensor (3 pins)	Variante variant 1
1	Heltec WiFi LoRa 32	Variante variant 1; module WiFi LoRa 32 V1.0; Alfileres 36; Número de componente HTIT-WB32LA
1	HLK-PM01	Variante variant 1; Alfileres 4; editable pin labels false; label HLK-PM01
1	CJMCU-811	Variante Variant 1
1	470Ω Resistor	paquete 0805 [SMD]; Resistencia 470Ω; tolerancia ±5%

12 Arduino programa

.h orrialdea

```
// Liburutegi hauek behar ditu Arduinok:
// - DHT sentsoreen liburutegia: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified sensor Lib: https://github.com/adafruit/Adafruit sensor
/*LIBURUTEGIAK*/
//DHT11
```

```
#include "DHT.h"
//CCS811
#include "Adafruit_CCS811.h"
//TTN
#include <lmic.h>
#include <hal/hal.h>
#include <SPI.h>

//LORA
#include <SPI.h>
#include <LoRa.h>

//OLED Display Pantaila
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//NeoPixel LED
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
 #include <avr/power.h> // Required for 16 MHz Adafruit Trinket
#endif

/*KONSTANTEAK*/
//DHT11 PIN
#define DHTPIN 17     // PIN digitala DHT sentsorera konektatuta
// DHT mota
#define DHTTYPE DHT11   // DHT 11

// LoRaWAN NwkSKey, network session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const PROGMEM u1_t NWKSKEY[16] = { 0x58, 0xF2, 0x01, 0xF7, 0xE5, 0xC1, 0x7F, 0x2E, 0x16,
0xB3, 0x04, 0x30, 0x99, 0x4D, 0xAB, 0x86 };

// LoRaWAN AppSKey, application session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const u1_t PROGMEM APPSKEY[16] = { 0x58, 0x9B, 0x5B, 0xDA, 0xDC, 0x3B, 0x20, 0xBC, 0x39,
0x8A, 0x48, 0xD5, 0xEE, 0x84, 0x99, 0x0A };

// LoRaWAN end-device address (DevAddr)
static const u4_t DEVADDR = 0x26013B1C ; // <-- Change this address for every node!

// Schedule TX every this many seconds (might become longer due to duty
// cycle limitations).
const unsigned TX_INTERVAL = 60;

// Pin mapping
const lmic_pinmap lmic_pins = {
    .nss = 18,
    .rxtx = LMIC_UNUSED_PIN,
    .rst = 14,
    .dio = {26, 33, 32},
};

//Lora komunikazio pinak
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26
```

```
//433E6 for Asia
//866E6 for Europe
//915E6 for North America
#define BAND 866E6

//OLED pinak
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in

//NeoPixelLed Pinak
#define PIN        13

#define NUMPIXELS 8 // Zenbat NeoPixel

/*OBJETUAK*/
Adafruit_CCS811 ccs;
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

//abiarazi DHT sentsorea
DHT dht(DHTPIN, DHTTYPE);

//NeoPixel
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

/*ALDAGAIAK*/
//static uint8_t mydata[] = "Hello, world!";
static osjob_t sendjob;

/*FUNTZIOAK*/
//DHT
void DHT11sentsoreasetup (void);
void DHT11sentsorealoop (void);
//CCS811
void CCS811sentsoreasetup (void);
void CCS811sentsorealoop (void);
//TTN//
void os_getArtEui (u1_t* buf) { }
void os_getDevEui (u1_t* buf) { }
void os_getDevKey (u1_t* buf) { }
void ttnabpSetup(void);
void ttnabpLoop(void);
//PANTAILA
void PantailaSetup(void);
void PantailaLoop(void);
//NEOPIXEL
void NeoPixelSetup(void);
void NeoPixelLoop(void);
//BIDALKETA
void BidalketaLoop (void);
```

Orrialde nagusia

```
/*
 Programaren izena: ProgramaNagusia.ino
 Egilea: Ander Urigoitia eta Xabier Intxausti    Data: 2021/02/24
 Zeregina: Lora bidezko komunikazioa
 Adibide hau domeinu publikokoa da.
 Programaren egoera: Egiaztatzeko/Egiaztatuta
 Egiaztatutako plakak: UNO ?? Nano Bai MEGA ???
 */
#include "ProgramaNagusia.h"
```

```
void setup() {
  Serial.begin(115200);
  DHT11sentsoreasetup();
  PantailaSetup();
  CCS811sentsoreasetup();
  ttnabpSetup();
  NeoPixelSetup();

}
void loop() {
DHT11sentsorealoop();
CCS811sentsorealoop();
ttnabpLoop();
PantailaLoop();
NeoPixelLoop();
}
```

Bidalketa orrialdea

```
/* Funtzioa: void BidalketaLoop (void)
 * Zeregina: funtzio honek tenperatura eta hezetasuna TTNra bidaltzen ditu
 * Erabilitako liburutegiak:
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 *  */
void BidalketaLoop (void){

  float h = dht.readHumidity();
  float t = dht.readTemperature();

unsigned long balioa =((t*10000000)+(h*10000)+(ccs.geteCO2())); //bidaliko dugun zenbakizko
txurroaren simulazioa, balioa adibide bat da TTTHHCCCC
String payload = String(balioa);                                    //txurroa string batera pasatu
static uint8_t payloadChar[10];                                     //karaktereen arraya sortu
payload.toCharArray((char *)payloadChar, sizeof(payloadChar));      //stringa char array bihurtu


// Prepare upstream data transmission at the next possible time. Bidalketa egiteko lekua,
payloadChar aldagaia bidaltzen du
LMIC_setTxData2(1, payloadChar, sizeof(payloadChar)-1, 0);
Serial.println(F("Packet queued"));
}
```

CCS811Sentsorea orrialdea

```
void CCS811sentsoreasetup (void){
    Serial.println("CCS811 test");

  if(!ccs.begin()){
    Serial.println("Failed to start sensor! Please check your wiring.");
    while(1);
  }

  // Wait for the sensor to be ready
  while(!ccs.available());
}//void CCS811sentsoreasetup (void) AMAIERA

/* Funtzioa: void CCS811sentsorealoop (void)
 * Zeregina: funtzio honek CO2 maila irakurtzen du
 * Erabilitako liburutegiak: "Adafruit_CCS811.h"
 * Liburutegia nondik hartuta: Arduino bertatik
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 *  */
 void CCS811sentsorealoop (void){
```

```
    if(ccs.available()){
    if(!ccs.readData()){
      Serial.print("CO2: ");
      Serial.print(ccs.geteCO2());
      Serial.print("ppm, TVOC: ");
      Serial.println(ccs.getTVOC());
    }
    else{
      Serial.println("ERROR!");
      while(1);
    }
  }
  delay(500);
 }//void CCS811sentsorealoop (void) AMAIERA
```

DHT11 Sentsorea orrialdea

```
void DHT11sentsoreasetup (void){
    Serial.println(F("DHTxx test!"));

  dht.begin();
}//void DHT11sentsoreasetup (void) AMAIERA
/* Funtzioa: void DHT11sentsorealoop (void)
 * Zeregina: funtzio honek tenperatura eta hezetasuna irakurtzen du
 * Erabilitako liburutegiak: - DHT sentsoreen liburutegia: https://github.com/adafruit/DHT-sensor-
library
                            - Adafruit Unified sensor Lib: https://github.com/adafruit/Adafruit
sensor
 * Liburutegia nondik hartuta: Arduino bertatik
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 *  */
void DHT11sentsorealoop (void){
    // Itxaron neurketa egin baino lehen
  delay(2000);

  float h = dht.readHumidity();
  // Irakurri tenperatura Celsius
  float t = dht.readTemperature();
  // Irakurri tenperatura Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  // Egiaztatu irakurketaren batek huts egin duen eta goiz atera (berriro saiatzeko).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Ezin izan da DHT sentsoretik irakurri!"));
    return;
  }
  // Kalkulatu bero-indizea Fahrenheiten (balio lehenetsia)
  float hif = dht.computeHeatIndex(f, h);
  // Kalkulatu bero-indizea Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Hezetasuna: "));
  Serial.print(h);
  Serial.print(F("%  Tenperatura: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F  Sentzasio termikoa: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
}//void DHT11sentsorealoop (void) AMAIERA
```

NeoPixelLed orrialdea

```
void NeoPixelSetup (void){
#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
  clock_prescale_set(clock_div_1);
#endif
  pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)
}

/* Funtzioa: void NeoPixelLoop (void)
 * Zeregina: CO2 maila bakoitzean led bidez egoera adierazi
 * Erabilitako liburutegiak: <Wire.h>
                             <Adafruit_GFX.h>
                             <Adafruit_SSD1306.h>
 * Liburutegia nondik hartuta: GitHub
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 *  */
void NeoPixelLoop (void){
CCS811sentsorealoop();
  pixels.clear(); // Set all pixel colors to 'off'

    // pixels.Color() takes RGB values, from 0,0,0 up to 255,255,255
    // Here we're using a moderately bright green color:

    if ((ccs.geteCO2()>=400)&&(ccs.geteCO2()<700)){
      pixels.setPixelColor(0, pixels.Color(0, 20, 0));
      pixels.setPixelColor(1, pixels.Color(0, 20, 0));
      pixels.setPixelColor(2, pixels.Color(0, 20, 0));
      pixels.setPixelColor(3, pixels.Color(0, 20, 0));
      pixels.setPixelColor(4, pixels.Color(0, 20, 0));
      pixels.setPixelColor(5, pixels.Color(0, 20, 0));
      pixels.setPixelColor(6, pixels.Color(0, 20, 0));
      pixels.setPixelColor(7, pixels.Color(0, 20, 0));
    }
    else{
      if ((ccs.geteCO2()>=700)&&(ccs.geteCO2()<800)){
        pixels.setPixelColor(0, pixels.Color(20, 20, 0));
        pixels.setPixelColor(1, pixels.Color(20, 20, 0));
        pixels.setPixelColor(2, pixels.Color(20, 20, 0));
        pixels.setPixelColor(3, pixels.Color(0, 20, 0));
        pixels.setPixelColor(4, pixels.Color(0, 20, 0));
        pixels.setPixelColor(5, pixels.Color(0, 20, 0));
        pixels.setPixelColor(6, pixels.Color(0, 20, 0));
        pixels.setPixelColor(7, pixels.Color(0, 20, 0));
      }
      else{
        if ((ccs.geteCO2()>=800)&&(ccs.geteCO2()<900)){
        pixels.setPixelColor(0, pixels.Color(20, 20, 0));
        pixels.setPixelColor(1, pixels.Color(20, 20, 0));
        pixels.setPixelColor(2, pixels.Color(20, 20, 0));
        pixels.setPixelColor(3, pixels.Color(20, 20, 0));
        pixels.setPixelColor(4, pixels.Color(20, 20, 0));
        pixels.setPixelColor(5, pixels.Color(0, 20, 0));
        pixels.setPixelColor(6, pixels.Color(0, 20, 0));
        pixels.setPixelColor(7, pixels.Color(0, 20, 0));
        }
        else{
          if((ccs.geteCO2()>=900)&&(ccs.geteCO2()<1000)){
        pixels.setPixelColor(0, pixels.Color(20, 20, 0));
        pixels.setPixelColor(1, pixels.Color(20, 20, 0));
        pixels.setPixelColor(2, pixels.Color(20, 20, 0));
        pixels.setPixelColor(3, pixels.Color(20, 20, 0));
        pixels.setPixelColor(4, pixels.Color(20, 20, 0));
        pixels.setPixelColor(5, pixels.Color(20, 20, 0));
```

```
        pixels.setPixelColor(6, pixels.Color(20, 20, 0));
        pixels.setPixelColor(7, pixels.Color(20, 20, 0));
          }
          else{
            if((ccs.geteCO2()>=1000)&&(ccs.geteCO2()<1200)){
        pixels.setPixelColor(0, pixels.Color(20, 0, 0));
        pixels.setPixelColor(1, pixels.Color(20, 0, 0));
        pixels.setPixelColor(2, pixels.Color(20, 0, 0));
        pixels.setPixelColor(3, pixels.Color(20, 20, 0));
        pixels.setPixelColor(4, pixels.Color(20, 20, 0));
        pixels.setPixelColor(5, pixels.Color(20, 20, 0));
        pixels.setPixelColor(6, pixels.Color(20, 20, 0));
        pixels.setPixelColor(7, pixels.Color(20, 20, 0));
            }
            else{
              if((ccs.geteCO2()>=1200)&&(ccs.geteCO2()<1500)){
        pixels.setPixelColor(0, pixels.Color(20, 0, 0));
        pixels.setPixelColor(1, pixels.Color(20, 0, 0));
        pixels.setPixelColor(2, pixels.Color(20, 0, 0));
        pixels.setPixelColor(3, pixels.Color(20, 0, 0));
        pixels.setPixelColor(4, pixels.Color(20, 0, 0));
        pixels.setPixelColor(5, pixels.Color(20, 20, 0));
        pixels.setPixelColor(6, pixels.Color(20, 20, 0));
        pixels.setPixelColor(7, pixels.Color(20, 20, 0));
              }
              else{
        pixels.setPixelColor(0, pixels.Color(20, 0, 0));
        pixels.setPixelColor(1, pixels.Color(20, 0, 0));
        pixels.setPixelColor(2, pixels.Color(20, 0, 0));
        pixels.setPixelColor(3, pixels.Color(20, 0, 0));
        pixels.setPixelColor(4, pixels.Color(20, 0, 0));
        pixels.setPixelColor(5, pixels.Color(20, 0, 0));
        pixels.setPixelColor(6, pixels.Color(20, 0, 0));
        pixels.setPixelColor(7, pixels.Color(20, 0, 0));
              }
            }
          }
        }
      }
    }

    pixels.show();   // Send the updated pixel colors to the hardware.
}
```

PantailanIkusi orrialdea

```
void PantailaSetup(void){
    //reset OLED display via software
  pinMode(OLED_RST, OUTPUT);
  digitalWrite(OLED_RST, LOW);
  delay(20);
  digitalWrite(OLED_RST, HIGH);

  //initialize OLED
  Wire.begin(OLED_SDA, OLED_SCL);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C for 128x32
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }

  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(0,0);
  display.print("LORA SENDER ");
```

```
  display.display();

  Serial.println("LoRa Sender Test");

  //SPI LoRa pins
  SPI.begin(SCK, MISO, MOSI, SS);
  //setup LoRa transceiver module
  LoRa.setPins(SS, RST, DIO0);

  if (!LoRa.begin(BAND)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
  Serial.println("LoRa ongi abiarazten");
  display.setCursor(0,10);
  display.print("LoRa ongi abiarazten");
  display.display();
  delay(2000);
//DHT11
      Serial.println(F("DHTxx test!"));

  dht.begin();

  //CCS811
      Serial.println("CCS811 test");

  if(!ccs.begin()){
    //Serial.println("Failed to start sensor! Please check your wiring.");
    while(1);
  }

  // Wait for the sensor to be ready
  while(!ccs.available());
}

void PantailaLoop(void){
  // Itxaron neurketa egin baino lehen
  delay(2000);

  float h = dht.readHumidity();
  // Irakurri tenperatura Celsius
  float t = dht.readTemperature();
  // Irakurri tenperatura Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  if(ccs.available()){
    if(!ccs.readData()){
      Serial.print("CO2: ");
      Serial.print(ccs.geteCO2());
      Serial.print("ppm, TVOC: ");
      Serial.println(ccs.getTVOC());
    }
    else{
      Serial.println("ERROR!");
      while(1);
    }
  }
  delay(500);

  display.clearDisplay();
  display.setCursor(0,0);
  display.println("AnderU Xabi");
  display.setCursor(0,10);
  display.setTextSize(1);
```

```
  display.print("Neurketak:");
    display.setCursor(0,30);
  display.print("Hezetasuna: ");
  //display.setCursor(50,30);
  display.print(h);
  display.print("% ");
  display.setCursor(0,40);
  display.print("Tenperatura: ");
  display.print(t);
  display.print("°C ");
  display.setCursor(0,50);
  display.print("CO2:");
  display.print(ccs.geteCO2());
  display.print("ppm,TVOC:");
  display.print(ccs.getTVOC());

  display.display();

  // Egiaztatu irakurketaren batek huts egin duen eta goiz atera (berriro saiatzeko).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Ezin izan da DHT sentsoretik irakurri!"));
    return;
  }
  // Kalkulatu bero-indizea Fahrenheiten (balio lehenetsia)
  float hif = dht.computeHeatIndex(f, h);
  // Kalkulatu bero-indizea Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Hezetasuna: "));
  Serial.print(h);
  Serial.print(F("%  Tenperatura: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F  Sentzasio termikoa: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
}
```

TTNFuntziak orrialdea

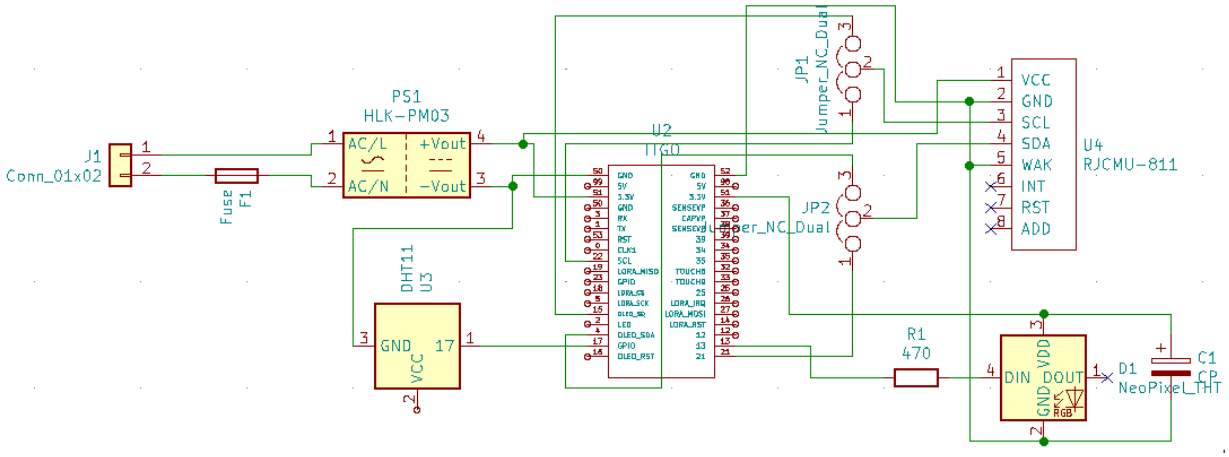
```
void onEvent (ev_t ev) {
    Serial.print(os_getTime());
    Serial.print(": ");
    switch(ev) {
        case EV_SCAN_TIMEOUT:
            Serial.println(F("EV_SCAN_TIMEOUT"));
            break;
        case EV_BEACON_FOUND:
            Serial.println(F("EV_BEACON_FOUND"));
            break;
        case EV_BEACON_MISSED:
            Serial.println(F("EV_BEACON_MISSED"));
            break;
        case EV_BEACON_TRACKED:
            Serial.println(F("EV_BEACON_TRACKED"));
            break;
        case EV_JOINING:
            Serial.println(F("EV_JOINING"));
            break;
        case EV_JOINED:
            Serial.println(F("EV_JOINED"));
            break;
```

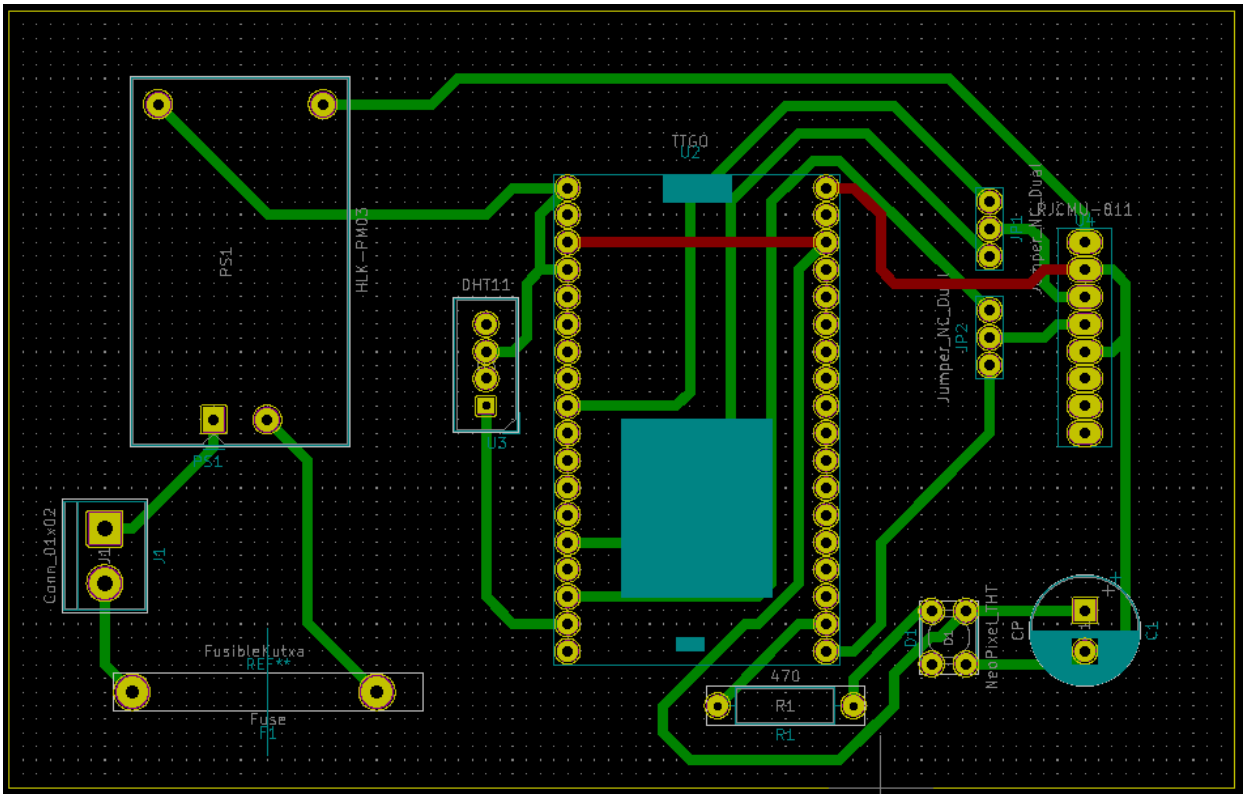
```
        case EV_RFU1:
            Serial.println(F("EV_RFU1"));
            break;
        case EV_JOIN_FAILED:
            Serial.println(F("EV_JOIN_FAILED"));
            break;
        case EV_REJOIN_FAILED:
            Serial.println(F("EV_REJOIN_FAILED"));
            break;
        case EV_TXCOMPLETE:
            Serial.println(F("EV_TXCOMPLETE (includes waiting for RX windows)"));
            if (LMIC.txrxFlags & TXRX_ACK)
              Serial.println(F("Received ack"));
            if (LMIC.dataLen) {
              Serial.println(F("Received "));
              Serial.println(LMIC.dataLen);
              Serial.println(F(" bytes of payload"));
            }
            // Schedule next transmission
            os_setTimedCallback(&sendjob, os_getTime()+sec2osticks(TX_INTERVAL), do_send);
            break;
        case EV_LOST_TSYNC:
            Serial.println(F("EV_LOST_TSYNC"));
            break;
        case EV_RESET:
            Serial.println(F("EV_RESET"));
            break;
        case EV_RXCOMPLETE:
            // data received in ping slot
            Serial.println(F("EV_RXCOMPLETE"));
            break;
        case EV_LINK_DEAD:
            Serial.println(F("EV_LINK_DEAD"));
            break;
        case EV_LINK_ALIVE:
            Serial.println(F("EV_LINK_ALIVE"));
            break;
         default:
            Serial.println(F("Unknown event"));
            break;
    }
}//void onEvent (ev_t ev) AMAIERA

void do_send(osjob_t* j){
    // Check if there is not a current TX/RX job running
    if (LMIC.opmode & OP_TXRXPEND) {
        Serial.println(F("OP_TXRXPEND, not sending"));
    } else {
        // Prepare upstream data transmission at the next possible time.
        //LMIC_setTxData2(1, mydata, sizeof(mydata)-1, 0);
        BidalketaLoop();
        Serial.println(F("Packet queued"));
    }
    // Next TX is scheduled after TX_COMPLETE event.
}//void do_send(osjob_t* j) AMAIERA
void ttnabpSetup(void){
  Serial.println(F("Starting"));

    #ifdef VCC_ENABLE
    // For Pinoccio Scout boards
    pinMode(VCC_ENABLE, OUTPUT);
    digitalWrite(VCC_ENABLE, HIGH);
    delay(600000);
    #endif
```

```
    // LMIC init
    os_init();
    // Reset the MAC state. Session and pending data transfers will be discarded.
    LMIC_reset();

    // Set static session parameters. Instead of dynamically establishing a session
    // by joining the network, precomputed session parameters are be provided.
    #ifdef PROGMEM
    // On AVR, these values are stored in flash and only copied to RAM
    // once. Copy them to a temporary buffer here, LMIC_setSession will
    // copy them into a buffer of its own again.
    uint8_t appskey[sizeof(APPSKEY)];
    uint8_t nwkskey[sizeof(NWKSKEY)];
    memcpy_P(appskey, APPSKEY, sizeof(APPSKEY));
    memcpy_P(nwkskey, NWKSKEY, sizeof(NWKSKEY));
    LMIC_setSession (0x1, DEVADDR, nwkskey, appskey);
    #else
    // If not running an AVR with PROGMEM, just use the arrays directly
    LMIC_setSession (0x1, DEVADDR, NWKSKEY, APPSKEY);
    #endif

    #if defined(CFG_eu868)
    // Set up the channels used by the Things Network, which corresponds
    // to the defaults of most gateways. Without this, only three base
    // channels from the LoRaWAN specification are used, which certainly
    // works, so it is good for debugging, but can overload those
    // frequencies, so be sure to configure the full frequency range of
    // your network here (unless your network autoconfigures them).
    // Setting up channels should happen after LMIC_setSession, as that
    // configures the minimal channel set.
    // NA-US channels 0-71 are configured automatically
    LMIC_setupChannel(0, 868100000, DR_RANGE_MAP(DR_SF12, DR_SF7),  BAND_CENTI);      // g-band
    LMIC_setupChannel(1, 868300000, DR_RANGE_MAP(DR_SF12, DR_SF7B), BAND_CENTI);      // g-band
    LMIC_setupChannel(2, 868500000, DR_RANGE_MAP(DR_SF12, DR_SF7),  BAND_CENTI);      // g-band
    LMIC_setupChannel(3, 867100000, DR_RANGE_MAP(DR_SF12, DR_SF7),  BAND_CENTI);      // g-band
    LMIC_setupChannel(4, 867300000, DR_RANGE_MAP(DR_SF12, DR_SF7),  BAND_CENTI);      // g-band
    LMIC_setupChannel(5, 867500000, DR_RANGE_MAP(DR_SF12, DR_SF7),  BAND_CENTI);      // g-band
    LMIC_setupChannel(6, 867700000, DR_RANGE_MAP(DR_SF12, DR_SF7),  BAND_CENTI);      // g-band
    LMIC_setupChannel(7, 867900000, DR_RANGE_MAP(DR_SF12, DR_SF7),  BAND_CENTI);      // g-band
    LMIC_setupChannel(8, 868800000, DR_RANGE_MAP(DR_FSK,  DR_FSK),  BAND_MILLI);      // g2-band
    // TTN defines an additional channel at 869.525Mhz using SF9 for class B
    // devices' ping slots. LMIC does not have an easy way to define set this
    // frequency and support for class B is spotty and untested, so this
    // frequency is not configured here.
    #elif defined(CFG_us915)
    // NA-US channels 0-71 are configured automatically
    // but only one group of 8 should (a subband) should be active
    // TTN recommends the second sub band, 1 in a zero based count.
    // https://github.com/TheThingsNetwork/gateway-conf/blob/master/US-global_conf.json
    LMIC_selectSubBand(1);
    #endif

    // Disable link check validation
    LMIC_setLinkCheckMode(0);

    // TTN uses SF9 for its RX2 window.
    LMIC.dn2Dr = DR_SF7;

    // Set data rate and transmit power for uplink (note: txpow seems to be ignored by the
library)
    LMIC_setDrTxpow(DR_SF7,14);

    // Start job
```

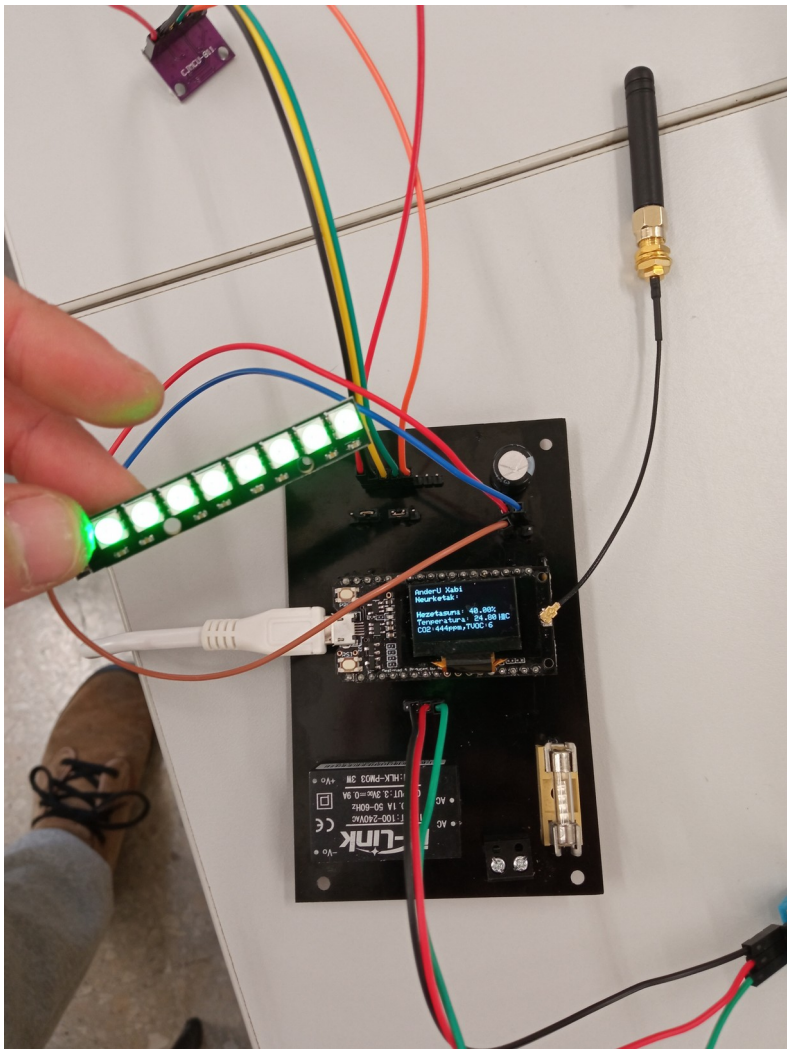
```
    do_send(&sendjob);
}//void ttnabpSetup(void) AMAIERA

void ttnabpLoop(void){
  os_runloop_once();
}//void ttnabpLoop(void) AMAIERA
```

13 KiCad plaka





3.1 13.1 Lortutakoa



14. Gorabeherak

Izan dugun gorabehera edo arazo nagusia plaka fresatu eta fisikoki montatzea izan da.
Beste arazo batzu ere izan ditugu:

- 3D makinan kutxa egitea ez dugu nahi genuen moduan lortu, neurri eta diseinuekin arazoak izan ditugu.
- DHT11 sentsoreak hainbat alditan ez digu balio adierazi, ezin zuela neurtu esanten zigun.
- CCS811 sentsorean batzuetan ez zuen aurrera egiten, arduinoa bertan blokeatuta geratzen zitzaigun

3.2 15. Hobetzeko proposamenak

- Praka fresatu eta arinago probatu.
- Arduino programa txukunago egiten saiatu