

LORA



Egileak

Jon M eta Mikel



Ziklo bukaerako proiektua: aurkibidea

1 Erronkaren planteamentua.....	3
2 Bloke-diagramak.....	4
2.1 Softwarearen hasierako bloke-diagrama.....	4
2.2 Hardwarearen hasierako bloke-diagram.....	5
2.3 Softwarearen amaierako bloke-diagrama.....	6
2.4 Hardwarearen amaierako bloke-diagrama.....	7
3 Haunarketa.....	8
4 Lora TTGO esp32.....	11
4.1 Sarrera.....	11
4.2 Lora liburutegiak.....	12
4.3 Lora probak.....	13
4.4 Lora Gateway konektatzeko programa.....	19
4.5 Lora neopixelarekin erabili.....	22
1.1 Tenperatura eta hezetasuna sentsorea.....	23
4.5.1 Sarrera.....	23
4.5.2 Eskema elektronikoak.....	24
4.5.3 Sentsorearen edo shieldaren pinout-a.....	24
4.5.4 Oinarrizko programa.....	24
4.5.5 Oinarrizko programa funtzioak bihurtuta.....	26
4.5.6 Material zerrenda.....	27
4.5.7 Gorabeherak.....	27
4.5.8 Estekak.....	27
4.6 CO2 sentsorea.....	28
4.6.1 Sarrera.....	28
4.6.2 Eskema elektronikoak.....	29
4.6.3 Sentsorearen edo shieldaren pinout-a.....	30
4.6.4 Oinarrizko programa.....	30
4.6.5 Oinarrizko programa funtzioak bihurtuta.....	31
4.6.6 Material zerrenda.....	32
4.6.7 Gorabeherak.....	32
4.6.8 Estekak.....	32
5 Arduino nano pinen esleipena.....	34
6 GateWay konfigurazioa.....	39
7 Elikatze iturria (Hi link).....	41
8 Egindako kalkulu guztiak.....	42
8.1 Elikatze-iturriaren kalkuluak.....	42
9 Eskema elektrikoak.....	43
LORA esp 32 eta DHT11 protoboard.....	43
LORA esp 32 eta CCS811 protoboard.....	43
LORA esp 32 ,CCS811 eta DHT11 protoboard.....	45
LORA esp 32 ,CCS811, DHT11 eta NEOPIXEL protoboard.....	46
10 10.Gorabeherak.....	47
11 PROGRAMA NAGUSIA.....	49
12 Egindako sistemaren argazkia.....	60



1 Erronkaren planteamentua

Erronka honetan bidali digute co2, tenperatura eta hezetasuna neurtzea eta honek datuak prozesatzeko Lora esp32 kontrolagailua. Datu honekin lortu behar duguna da WEB SERVER batean irakur ahal izatea eta ona heltzeko hainbat pausu pasatu behar dugu. Lehenik datuak GATEWAY batera bidaltzea eta irakurtzea, ostean RASPERRRY locala sortu.

Erronka bukatzen dugunean github plataformara dena dokumentatu eta programa osoak igo beharko dugu.

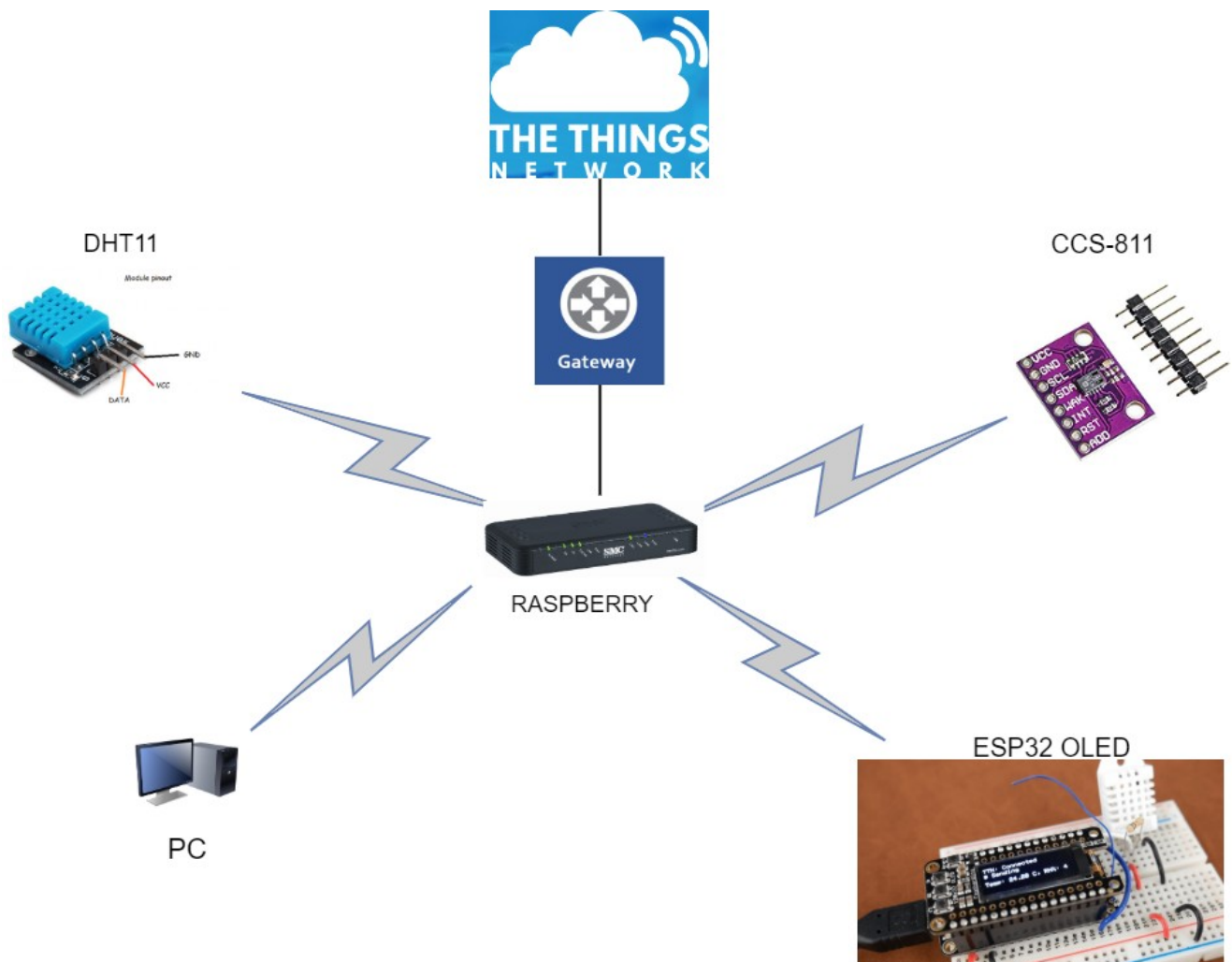


2 Bloke-diagramak

Lehen bertsioa izango da hau. Erronkaren amaieran igo egingo duzue azken bertsioa.

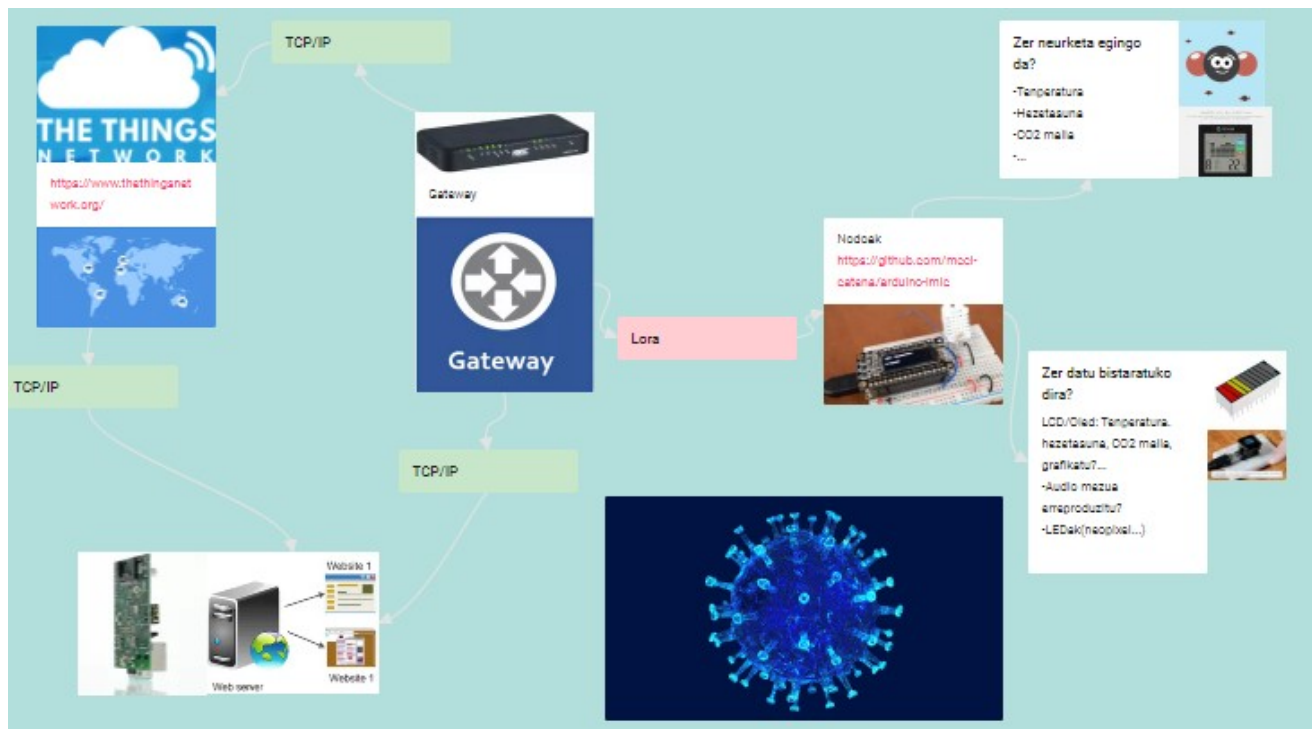
2.1 Softwarearen hasierako bloke-diagrama

Softwarea: geziekin, bloke bakoitza zelan eta zertarako erabiliko den azalduta azpian. Aldagaiak eta funtzioen izenak aurreikustea ere komeniko litzateke. Sekuentzia edo ordena inportantea dela kontutan izan.



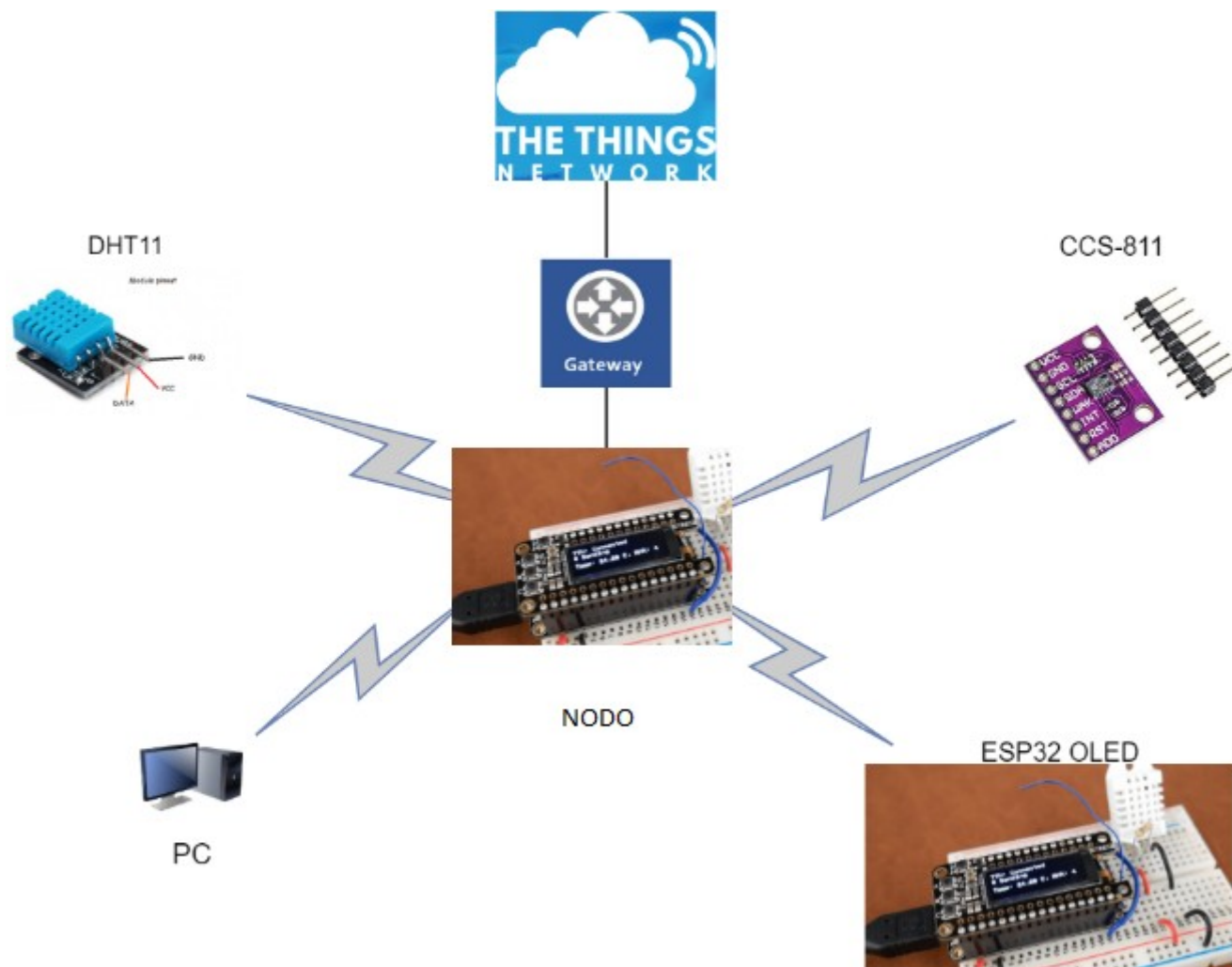
2.2 Hardwarearen hasierako bloke-diagrama

Hardwarea: geziekin, bloke bakoitza zelan eta zertarako erabiliko den azalduta azpian.



2.3 Softwarearen amaierako bloke-diagrama

Softwarea: geziekin, bloke bakoitza zelan eta zertarako erabiliko den azalduta azpian. Aldagaiak eta funtzioen izenak aurreikustea ere komeniko litzateke. Sekuentzia edo ordena inportantea dela kontutan izan.



Hardwarea: geziekin, bloke bakoitza zelan eta zertarako erabiliko den azalduta azpian.

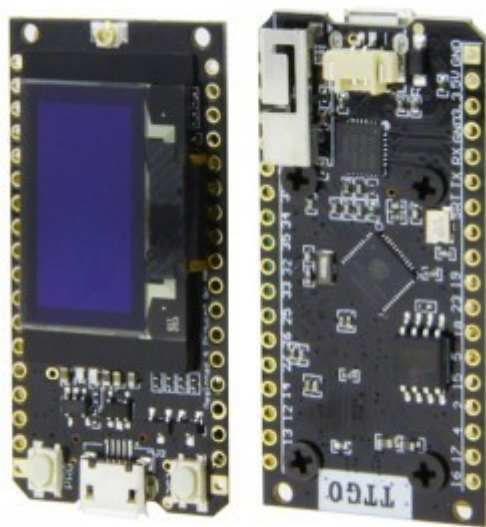
The diagram illustrates a LoRaWAN network architecture. At the top left, a box for 'THE THINGS NETWORK' includes the URL <https://www.thethingsnetwork.org/>. Below it, a 'Gateway' box features an image of a LoRa gateway device and a circular icon with four arrows pointing outwards. To the right of the gateway is a 'Lora' box. Below the gateway is a 'Web server' box with a large red 'X' over it, indicating it is not the correct path. The flow is as follows: 'Nodes' (represented by a box with a URL <https://github.com/moolenaar/arduino-lora> and an image of an Arduino board) send data to the 'Gateway' via 'LoRa'. The 'Gateway' then sends data to the 'Web server' via 'TCP/IP'. A separate box on the right lists various data types that can be sent: 'Zer neurteta egingo da?' (What can be sent?) including 'Temperatura', 'Hazetasuna', 'CO2 maila', and '...'. Another box below it lists 'Zer datu bideratuko dira?' (What data will be sent?) including 'LCD/Oled: Temperatura, hazetasuna, CO2 maila, grafikatu?...'. A third box at the bottom right shows a 'Web server' box with a large red 'X' over it, indicating it is not the correct path. A final box at the bottom right shows a 'Web server' box with a large red 'X' over it, indicating it is not the correct path.



3 Haunarketa

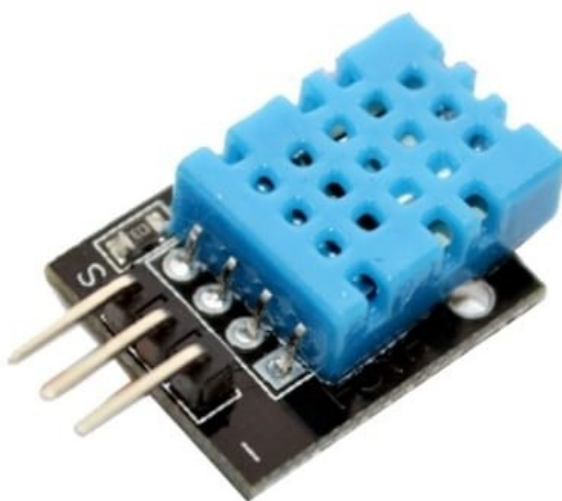
LORA ESP32

- Norabide bakarreko, norabide biko edo hedapen anitzeko lan-gaitasuna
- Interferentziekiko tolerantzia handia
- Sentsibilitate handia datuak jasotzeko (-168dB)
"Chirp" modulazioan oinarritua
- Kontsumo baxua (10 urtera arte bateria batekin)
- Luzera: 10-20 km
- Datuen transferentzia txikia (255 byte gehienez)
- Puntuz puntuko konexioa
- Lan maiztasunak: 868 Mhz European, 915 Mhz Amerikan eta 433 Mhz Asian



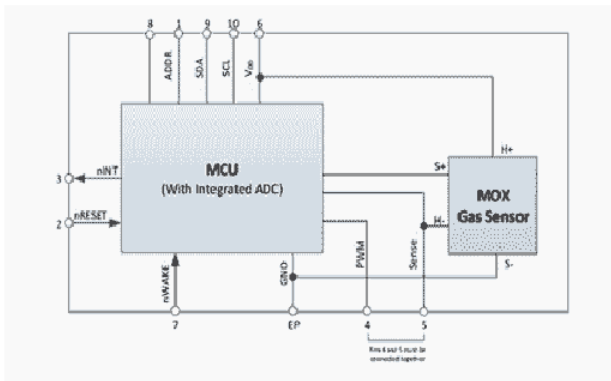
DH11

- DHT11 eta DHT22 (edo AM2302) sentsores familia bereko bi modelo dira, tenperatura eta hezetasuna aldi berean neurtzeko aukera ematen dutenak.
- Sentsores horiek barne-prozesadore bat dute, neurketa-prozesua egiten duena, eta neurketa seinale digital baten bidez ematen dute; beraz, oso erraza da neurketa Arduino bezalako mikroprozesadore batetik lortzea.
- Bi sentsoreek antzeko plastikozko kapsulatzea dute. Bi ereduak kolorearen arabera bereiz ditzakegu. DHT11ak karkasa urdin bat du, DHT22 sentsoresaren kasuan, berriz, kanpoaldea zuria da.



CCS811

- CCS811 AMS fabrikatzailearen sentsore bat da barneko airearen kalitatea neurtzeko, eta erraz erabil dezakegu Arduino bezalako prozesadore batekin batera.
- Barneko airearen kalitatea zehazteko, CCS811 MOX (Metal-Oxide) sentsore multigas bat da, karbono monoxidoa (Co) eta konposatu lurrunkorrak (VOCs) neurtzea barne hartzen duena, hala nola etanola, aminak edo hidrokarburo aromatikoak.
- Horien bidez, CCS811k karbono dioxido baliokidearen (eCO₂) kantitatea zehaztu dezake. Neurketa-tartea 400 eta 8192 ppm artekoa da eCO₂-n eta 0 eta 1187 ppb artekoa TVOCen. Datasheet-ak ez du neurketaren zehaztasunari buruzko daturik ematen.



4 Lora TTGO esp32

4.1 Sarrera

TTGO LoRa32 SX1276 OLED garapen plaka bat da, LoRa txip bat eta 0,96 hazbeteko OLED SSD1306 pantaila dituen. Gida honetan erakutsiko dizugu nola bidali eta jaso LoRa paketeak (puntuz puntuko komunikazioa) eta nola erabili OLED pantaila Arduino IDEekin.



4.2 Lora liburutegiak

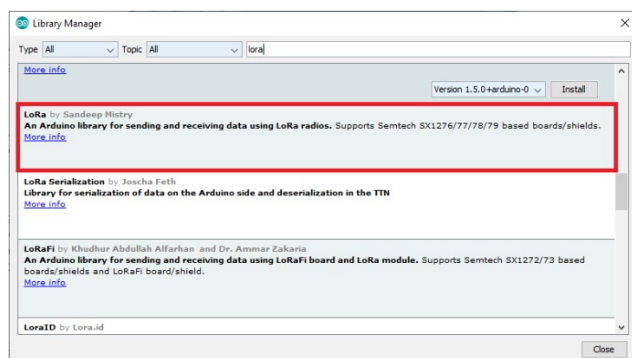
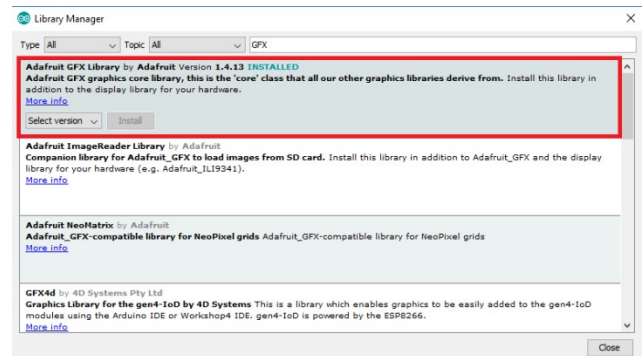
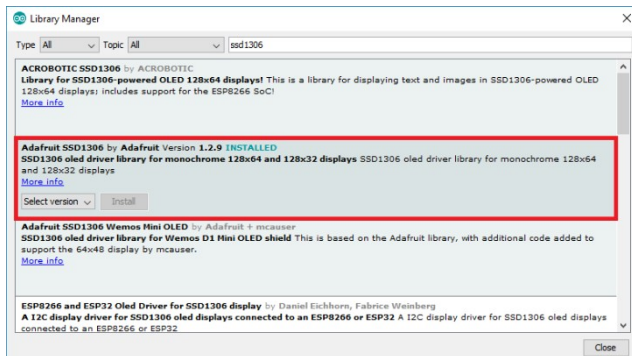
Lora esp32 erabili ahal izateko hainbat librería instalatu behar dira eta orain esteka bidez eta argazki bidez adieraziko dizuet.

Asteko lora funtzionatzeko hainbat librería instalatu behar dira eta esteka honetatik pausuak jarraitzen atera ditugu:

<https://randomnerdtutorials.com/ttgo-lora32-sx1276-arduino-ide/>

<https://github.com/matthijskooijman/arduino-lmic.git>

<https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/>



4.3 Lora probak

Lora probatzeko programa bat sartu diogu, nun programa hau 10segunduro mezuak bidaltzen ditu beste dispositibo batera.

```
/*  
  Rui Santos  
  Complete project details at https://RandomNerdTutorials.com/ttgo-lora32-sx1276-  
  arduino-ide/  
  */  
  
//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 DI00 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
```



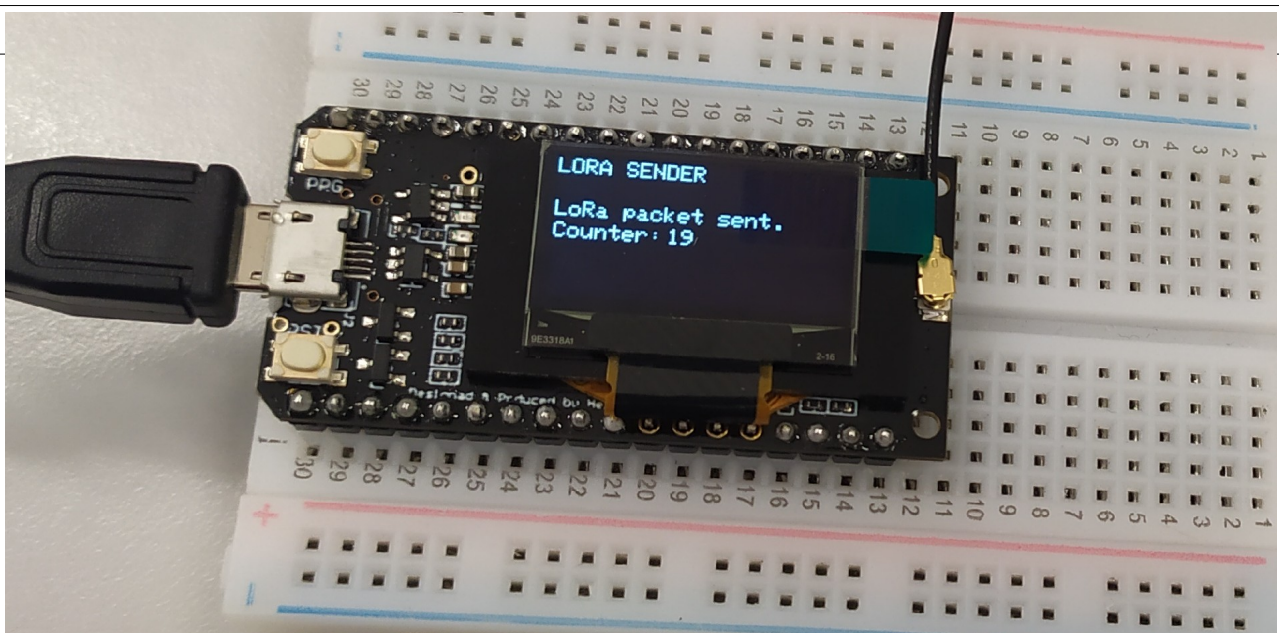
```

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, DI00);
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);}
```



Mezu honek jasotzeko beste programa bat kargatu behar da.

```
/*****
```

```
Rui Santos
```

```
Complete project details at https://RandomNerdTutorials.com/ttgo-lora32-sx1276-arduino-ide/
```

```
//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 DI00 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()) {

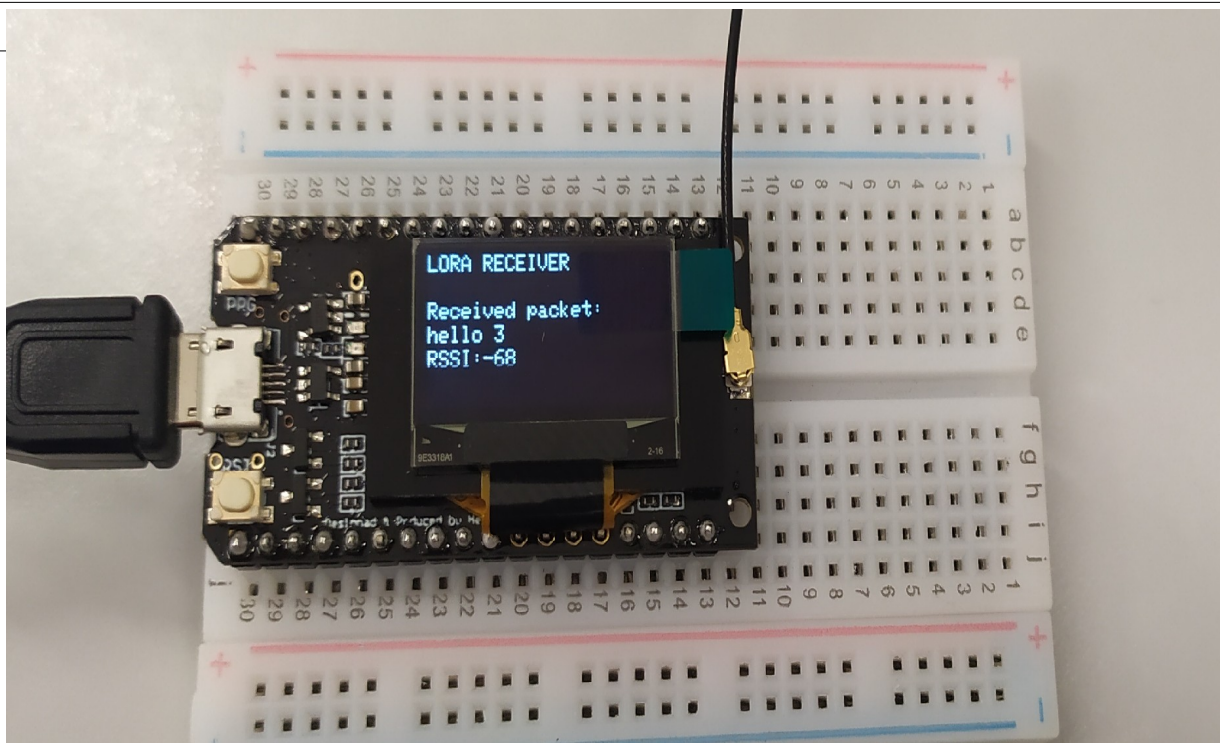
```



```

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

```



4.4 Lora Gateway konektatzeko programa

Programa hau erabiltzen dugu gure lora esp32 mezuak bidaltzeko gure gateway-eri, eta horrela ikusi ahal dugu gure dispositiboaren mezuak gateway aplikazioan.

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

// 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] = { 0x0E, 0x47, 0xF6, 0x3E, 0xF4, 0x6E, 0x65,
0x07, 0x9B, 0x41, 0x20, 0x16, 0x69, 0x2C, 0x09, 0xBD };
// 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] = { 0x1B, 0x54, 0x8E, 0xAC, 0x66, 0x0F, 0x32,
0x0A, 0xAB, 0x21, 0x09, 0x04, 0x46, 0x1C, 0x08, 0x6A };
// LoRaWAN end-device address (DevAddr)
static const u4_t DEVADDR = 0x26011E4D ; // <-- Change this address for every node!
// These callbacks are only used in over-the-air activation, so they are
// left empty here (we cannot leave them out completely unless
// DISABLE_JOIN is set in config.h, otherwise the linker will complain).
void os_getArtEui (u1_t* buf) { }
void os_getDevEui (u1_t* buf) { }
void os_getDevKey (u1_t* buf) { }
static uint8_t mydata[] = "Hello, world!";
static osjob_t sendjob;
// 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},
};
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 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);
        Serial.println(F("Packet queued"));
    }
    // Next TX is scheduled after TX_COMPLETE event.
}

void setup() {
    Serial.begin(115200);
    Serial.println(F("Starting"));
    #ifdef VCC_ENABLE
    // For Pinoccio Scout boards
    pinMode(VCC_ENABLE, OUTPUT);
    digitalWrite(VCC_ENABLE, HIGH);
    delay(1000);

```




```

#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 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
#ifdef 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_SF9;

```



```

    // 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 loop() {
    os_runloop_once();
}

```

4.5 Lora neopixelarekin erabili

Lehenik eta behin neopixela arduino nano batekin konprobatu dogu ea ondo dabilela.

Hemen daukagu deskargatu beha diren librería eta informazioa.

https://github.com/adafruit/Adafruit_NeoPixel

<https://www.youtube.com/watch?v=gJm11ME-Un8>

```

// 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 100 // 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'

    // The first NeoPixel in a strand is #0, second is 1, all the way up
    // to the count of pixels minus one.
    for(int i=0; i<8; i++) { // For each pixel...

        // 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(i, pixels.Color(50, 0, 0));

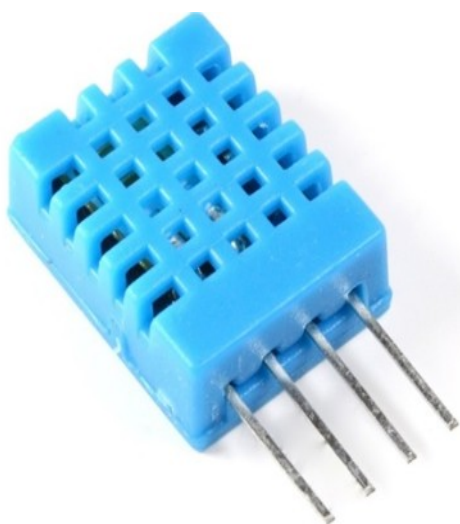
pixels.show(); // Send the updated pixel colors to the hardware.

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

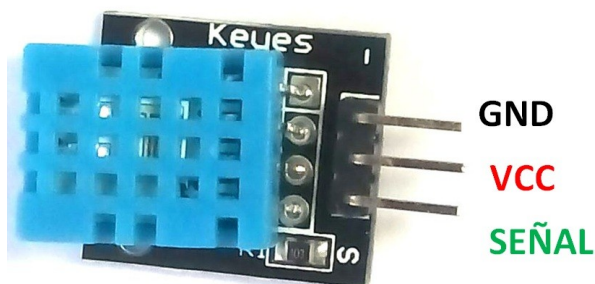
1.1 Temperatura eta hezetasuna sentsoarea

4.5.1 Sarrera

DHT11 temperatura eta hezetasun digitaleko sentsoare bat da, kostu txikikoa. Hezetasun-sentsore kapazitibo bat eta termistor bat erabiltzen ditu inguruko airea neurtzeko, eta datuak seinale digital baten bidez erakusten ditu datu-pinuan (ez dago sarrerako pinu analogikorik).



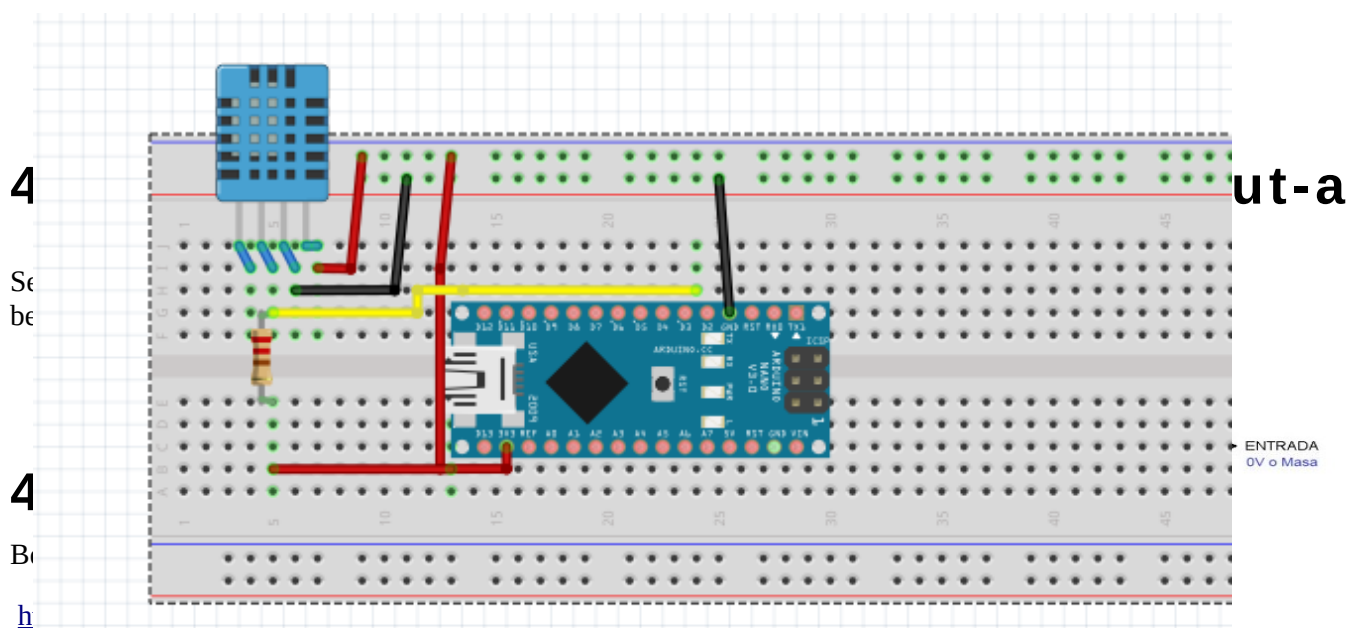
- Operazio-tentsioa: 3V-5VDC.
- Temperatura neurtzeko tartea: 0 eta 50 ° C bitartean.
- Temperatura neurtzeko doitasuna: ± 2.0 ° c
- Denbora-ebazpena: 0.1 ° C.
- Hezetasuna neurtzeko tartea: % 20tik % 90era Rh.
- Hezetasuna neurtzeko zehaztasuna: % 5 Rh.
- Hezetasunaren ebazpena: Rh% 1. Sensida-denbora: 1 seg.



<https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf>

<https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf>

4.5.2 Eskema elektronikoak



Tipo Todos Tema Todos DHT-sensor-library

AM232X
by Rob Tillaart
Arduino library for AM2320 AM2321 and AM2323 I2C temperature and humidity sensor. Supports AM2320, AM3231, AM2322. These sensors are similar to DHT12 with I2C interface.
[More info](#)

Versión 0.3.1 Instalar

DHT sensor library
by Adafruit Versión 1.4.1 **INSTALLED**
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
[More info](#)

/*
Programaren izena: DHTtester_JON_MIKEL



24. orrialdea
lan hau IURRETA LHIK sortu du eta [Creative Commons lizentzia](#) baten mende dago.



```

Egilea: JON eta MIKEL    Data: 22/02/2021
Zeregina: DHT sentsoarekin tenperatura eta hezetazuna neurtzea.
Adibide hau domeinu publikokoa da.
Programaren egoera: Egiaztatzeko/Egiaztatuta
Egiaztatutako plakak: hasiera batean Nano-arekin konprobatu dugu.
*/
/*-----( LIBURUTEGIAK )-----*/
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit\_Sensor
#include <DHT.h>
#include <DHT_U.h>
#include "DHT.h"
#define DHTPIN 2      // DHT sentsoareari lotutako Digital pin

//Erabili pinak 3, 4, 5, 12, 13 edo 14.
//Pin 15ek funtziona dezake, baina DHT deskonektatu egin behar da programazioan.

/*-----( KONSTANTEAK )-----*/
#define DHTTYPE DHT11  // DHT 11
// #define DHTTYPE DHT22  // DHT 22  (AM2302), AM2321
// #define DHTTYPE DHT21  // DHT 21  (AM2301)

// Connect pin 1 (ezkerrean) of the sensor to +5V
//3.3V logika duen taula bat Arduino Due batek pin 1 konektatzen badu
//5V ordez, 3.3V!
//Zure DHTPIN DOKTOREARI
//X/Connect pin 4 (eskuinaldean)
//Connect a 10K resistor from pin 2 (datuak) to pin 1 (boterea)

//DHT sentsoarea.
//Liburutegi honen bertsio zaharragoek hirugarren parametro bat eraman zuten.
//Parametro hau ez da beharrezkoa prozesadore azkarragoentzat.
//DHT irakurgailuaren algoritmoa prozesu azkarragoetan lan egitera egokitzen da.
//DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  Serial.println(F("DHTxx test!"));

  dht.begin();
}

void loop() {
  // Wait a few seconds between measurements.
  delay(2000);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  // Compute heat index in Fahrenheit (the default)
  float hif = dht.computeHeatIndex(f, h);
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

```



```

Serial.print(F("hezetasuna: "));
Serial.print(h);
Serial.print(F("%  Temperatura: "));
Serial.print(t);
Serial.print(F("°C "));
Serial.print(f);
Serial.print(F("°F  Bero-indizea: "));
Serial.print(hic);
Serial.print(F("°C "));
Serial.print(hif);
Serial.println(F("°F"));
}

```

```

hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.50°C 70.70°F  Bero-indizea: 20.98°C 69.77°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.50°C 70.70°F  Bero-indizea: 20.98°C 69.77°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F
hezetasuna: 49.00%  Temperatura: 21.60°C 70.88°F  Bero-indizea: 21.09°C 69.97°F

```

```

dht.begin();
}

/* Funtzioa: int DTH11sentsorea (int zeinPin)
 * Zeregina: DTH11 sentsorean temperatura eeta hezetasuna naurtzen du.
 * Erabilitako liburutegiak: DTH11DHT Sensor Library eta Adafruit
 * Liburutegia nondik hartuta: https://github.com/adafruit/Adafruit\_Sensor //
https://github.com/adafruit/DHT-sensor-library
 * Bueltatzen duena: ezer
 * Sartutako parametroak:
 *     int
 *
 * */
void DTH11sentsoreaLOOP (void){

    // Wait a few seconds between measurements.
    delay(2000);

    // Reading temperature or humidity takes about 250 milliseconds!
    // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
    float h = dht.readHumidity();
    // Read temperature as Celsius (the default)
    float t = dht.readTemperature();
    // Read temperature as Fahrenheit (isFahrenheit = true)
    float f = dht.readTemperature(true);

```




```
// Check if any reads failed and exit early (to try again).
if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
}

// Compute heat index in Fahrenheit (the default)
float hif = dht.computeHeatIndex(f, h);
// Compute heat index in Celsius (isFahreheit = false)
float hic = dht.computeHeatIndex(t, h, false);

Serial.print(F("hezetasuna: "));
Serial.print(h);
Serial.print(F("%  Temperatura: "));
Serial.print(t);
Serial.print(F("°C "));
Serial.print(f);
Serial.print(F("°F  Bero-indizea: "));
Serial.print(hic);
Serial.print(F("°C "));
Serial.print(hif);
Serial.println(F("°F"));
}
```

4.5.6 Material zerrenda

Izena	Kopurua	Oharrak edota ezaugarri bereziak
DHT11	1	Errezistentzia erabiltzen dugu. (PULL UP)
ARDUINO NANO	1	Sentsorea probetako erabiltzen dugu.

4.5.7 Gorabeherak

4.5.8 Estekak

<https://steve.fi/hardware/temperature/>

<https://www.zonamaker.com/electronica/intro-electronica/teoria/resistencias-de-pull-up-y-pull-down>

<https://forum.fritzing.org/t/heltec-dev-boards/8112>

<https://fritzing.org/projects/dht11-temp-and-humidity-sensor>

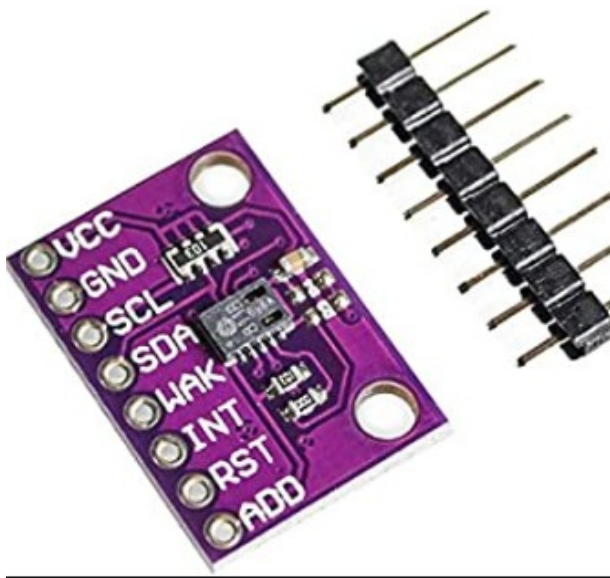


4.6 CO2 sentsorea

4.6.1 Sarrera

CCS811 sentsore bat da barneko airearen kalitatea neurtzeko, eta erraz erabil dezakegu Arduino bezalako prozesadore batekin batera.

Barneko airearen kalitatea zehazteko, CCS811 MOX (Metal-Oxide) sentsore multigas bat da, karbono monoxidoa (CO) eta konposatu lurrunkorrak (VOCs) neurtzea barne hartzen duena, hala nola etanola, aminak edo hidrokarburo aromatikoak. Horien bidez, CCS811k karbono dioxido baliokidearen (eCO₂) kantitatea zehaztu dezake. Neurketa-tartea 400 eta 8192 ppm artekoa da eCO₂-n eta 0 eta 1187 ppb artekoa TVOCen.



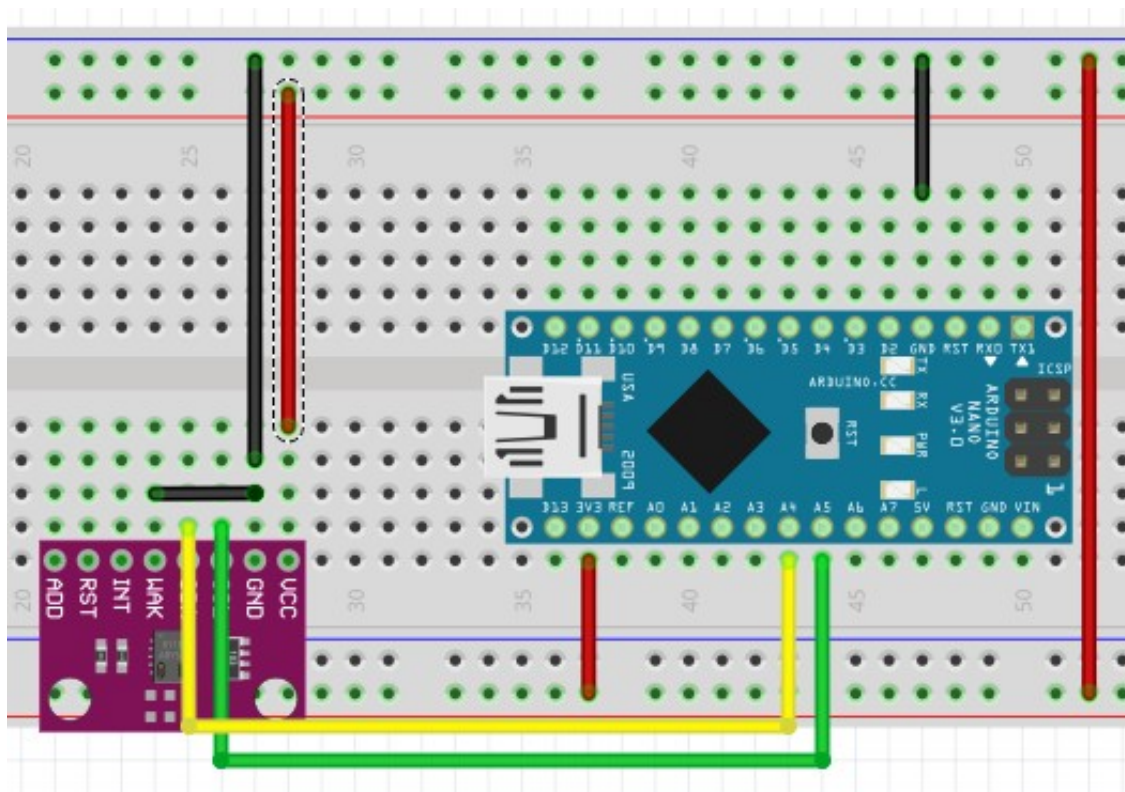
- Konposatu organiko lurrunkorra guztira (TVOC), milioi bakoitzeko 0 eta 1187 zati bitartean detektatzen dituen
- eCO₂ren detekzioa, 400etik 8,192ra milioiko.
- Bost funtzionamendu-modu
- MCU integratua
- I2C interfaze digital estandarra
- Kontsumo txikiko moduak
- Elikadura: 3.3V a 5V
- Neurriak: 21 x 18 x 3 milimetro
- Pisua: 1.2 gramo

Datasheet-ak ez du neurketaren zehaztasunari buruzko daturik ematen.

https://cdn.sparkfun.com/assets/2/c/c/6/5/CN04-2019_attachment_CCS811_Datasheet_v1-06.pdf



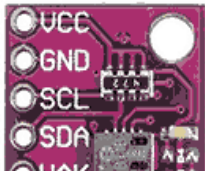
4.6.2 Eskema elektronikoak



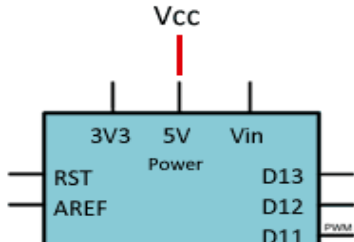
4.6.3 Sentsorearen edo shieldaren pinout-a

4

5V — Vcc
GND — GND
A5 — SCL
A4 — SDA
GND — GND



grai
fruit/Ada



Gestor de Librerías

Tipo: Todos Tema: Todos adafruit_ccs81

Adafruit CCS811 Library
by Adafruit Versión 1.0.5 **INSTALLED**
This is a library for the Adafruit CCS811 I2C gas sensor breakout. CCS811 is a gas sensor that can detect a wide range of Volatile Organic Compounds (VOCs) and is intended for indoor air quality monitoring.
[More info](#)

```
/*
Programaren izena: DHTtester_JON_MIKEL
Egilea: JON eta MIKEL   Data: 22/02/2021
Zeregina: DHT sentsorearekin tenperatura eta hezetazuna neurtzea.
Adibide hau domeinu publikokoa da.
Programaren egoera: Egiaztatzeko/Egiaztatuta
Egiaztatutako plakak: hasiera batean Nano-arekin konprobatu dugu.
*/
/*-----( LIBURUTEGIAK )-----*/
#include "Adafruit_CCS811.h" //https://github.com/adafruit/Adafruit_CCS811

Adafruit_CCS811 ccs;

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

  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 loop() {
  if(ccs.available()){
    if(!ccs.readData()){
      //Serial.println(ccs.calculateTemperature());
      Serial.print("°C, CO2: ");
      Serial.print(ccs.geteCO2());
    }
  }
}
```



```

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

```

COM3

CCS811 test

°C, CO2: 400ppm, TVOC: 0
 °C, CO2: 400ppm, TVOC: 0
 °C, CO2: 411ppm, TVOC: 1
 °C, CO2: 411ppm, TVOC: 1
 °C, CO2: 425ppm, TVOC: 3
 °C, CO2: 434ppm, TVOC: 5
 °C, CO2: 436ppm, TVOC: 5
 °C, CO2: 402ppm, TVOC: 0
 °C, CO2: 400ppm, TVOC: 0
 °C, CO2: 400ppm, TVOC: 0
 °C, CO2: 400ppm, TVOC: 0
 °C, CO2: 400ppm, TVOC: 0
 °C, CO2: 406ppm, TVOC: 0
 °C, CO2: 411ppm, TVOC: 1
 °C, CO2: 411ppm, TVOC: 1

☐ Autoscroll
 ☐ Mostrar marca temporal

4.6.5 Oinarrizko programa funtzioak bihurtuta

```

/* Funtzioa: void C02sentsorea (void)
 * Zeregina:CO2 sentsorean naurtzea du.
 * Erabilitako liburutegiak: DTH11DHT Sensor Library eta Adafruit
 * Liburutegia nondik hartuta://https://github.com/adafruit/Adafruit\_CCS811
 * Bueltatzen duena: ezer
 * Sartutako parametroak:
 *     int
 *

```



```

* */
void CO2sentsoreaSETUP (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 CO2sentsorea (void)
* Zeregina:CO2 sentsorean naurtzea du.
* Erabilitako liburutegiak: DHT11DHT Sensor Library eta Adafruit
* Liburutegia nondik hartuta://https://github.com/adafruit/Adafruit CCS811
* Bueltatzen duena: ezer
* Sartutako parametroak:
*     int
* */
void CO2sentsoreaLOOP (void){
    if(ccs.available()){
        if(!ccs.readData()){
            //Serial.println(ccs.calculateTemperature());
            Serial.print("CO2: ");
            Serial.print(ccs.geteCO2());
            Serial.println("ppm");
            Serial.print("TVOC: ");
            Serial.println(ccs.getTVOC());
        }
        else{
            Serial.println("ERROR!");
            while(1);
        }
    }
    delay(500);
}

```

4.6.6 Material zerrenda

Izena	Kopurua	Oharrak edota ezaugarri bereziak
CO2 (Sentsorea)	1	
ARDUINO NANO	1	Sentsorea probetako erabiltzen dugu.

4.6.7 Gorabeherak

4.6.8 Estekak

<https://www.luisllamas.es/medir-calidad-del-aire-y-co2-con-ccs811-y-arduino/>

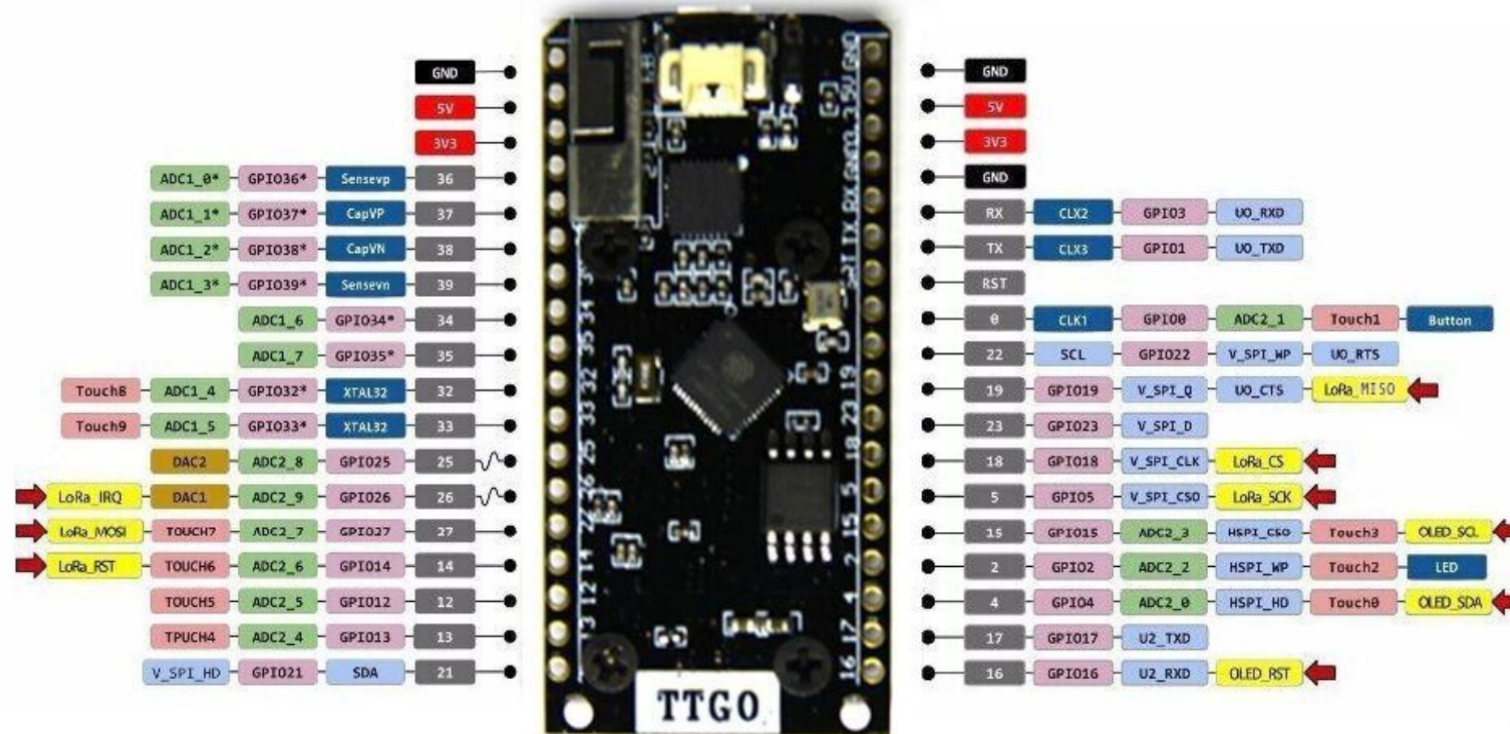


<https://tienda.bricogeek.com/sensores-gas/1508-sensor-de-calidad-del-aire-ccs811-voc-y-co2.html>

<http://digitalconcepts.net.au/fritzing/index.php?op=parts>



5 Arduino nano pinen esleipena



1.8. ESP-32 hardware eta software baliabideak

DIGITALA	KOMUNIKAZIOAK	GAINERAKOAK	ERABILERA
GPI0/GPO0			
GPI1/GPO1	UART TxD		Ordenagailuarekin komunikatzeko (Serial) Datuak bidali
GPI2/GPO2			
GPI3/GPO3	UART RxD		Ordenagailuarekin komunikatzeko (Serial) Datuak jaso
GPI4/GPO4	SDA		OLED_SDA ETA CJMCU-811 (CO2_Sentsorea)
GPI5/GPO5	SPI CS0		LORA_SCK ETA CJMCU-811 (CO2_Sentsorea)
GPI12/GPO12			
GPI13/GPO13			CJMCU-2812-8(Led Neopixel)
GPI14/GPO14			LORA_RST
GPI15/GPO15	SCL		OLED_SCL
GPI16/GPO16	U2 RXD		OLED_RST
GPI17/GPO17	U2 TXD		DHT11 (Tenperatura eta Hezetasuna)

DIGITALA	KOMUNIKAZIOAK	GAINERAKOAK	ERABILERA
GPI18/GPO18	SPI CLK		LORA_CS
GPI19/GPO19			LORA_MISO
GPI21/GPO21	SDA		CJMCU-811 (CO2_Sentsorea)
GPI22/GPO22	SCL		CJMCU-811 (CO2_Sentsorea)
GPI23/GPO23			
GPI24/GPO24			
GPI25/GPO25			
GPI26/GPO26			LORA_IRQ
GPI27/GPO27			LORA_MOSI
GPI32/GPO32			
GPI33/GPO33			
GPI34/GPO34			
GPI35/GPO35			

DIGITALA	KOMUNIKAZIOAK	GAINERAKOAK	ERABILERA
GPI36/GPO36			
GPI37/GPO37			
GPI38/GPO38			
GPI39/GPO39			



6 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 pultsautu mantendu LEDa gorri 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.




192.168.4.1











MiniHub Setup

Setup network closes in 08:57 Minutes

Configured Networks (1 / 8 max) - Click to remove

	MH_CONFIG	-
---	-----------	---

Scanned Networks (00:55 Minutes ago) - Click to add

		<div></div>	+
		<div></div>	+
		DIRECT-1b-HP M477 LaserJet	+
		HP-Print-B6-Officejet Pro 8620	+
		VFNL-F49DE8	+

Add Network

Your Network

ADD

CANCEL

SAVE & REBOOT



- Azkenik, 192.168.4.1 sarrera sartu eta hor Wifia aukeratu duguKonfigurazioa 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.



7 Elikatze iturria (Hi link)

Sarrera

Serieko elikadura-modulu ultra-txikia Potentzia ultra txikikoa, serieko hornidura-modulu txikia, 3W bolumen txiki bat da, elikadura-efizientzia handia, Shenzhen Hola-Link Electronics Co, Ltd enpresak diseinaturia. Abantaila hauek ditu: sarrera globaleko tentsio-tartea, tenperatura-igoera txikia, energia-kontsumo txikia, eraginkortasun handia, fidagarritasun handia eta segurtasun-isolamendu handia. Asko erabili da etxe adimendunetan, automatizazio-kontrolan, komunikazio-ekipoetan, tresnerian eta beste industria batzuetan.



Ezaugarriak

- Sarrerako tentsio nominala: 100-240 VCA
- Sarrerako tentsio-maila: 90-264 VCA
- Gehieneko sarrera-boltajea: 270 Vac
- Gehieneko sarrera-korrontea: 0.2tik
- Sarrerako gainjarpena: 10-a
- Hasiera motela: 50 ms
- Tentsio txikiko sarrera-eraginkortasuna: $v_{in} = 110$ VCA Irteera osoa,% 69ko irteerako irteera osoa

Estekak

<https://datasheetspdf.com/datasheet/HLK-PM01.html>



8 Egindako kalkulu guztiak

8.1 Elikatze-iturriaren kalkuluak

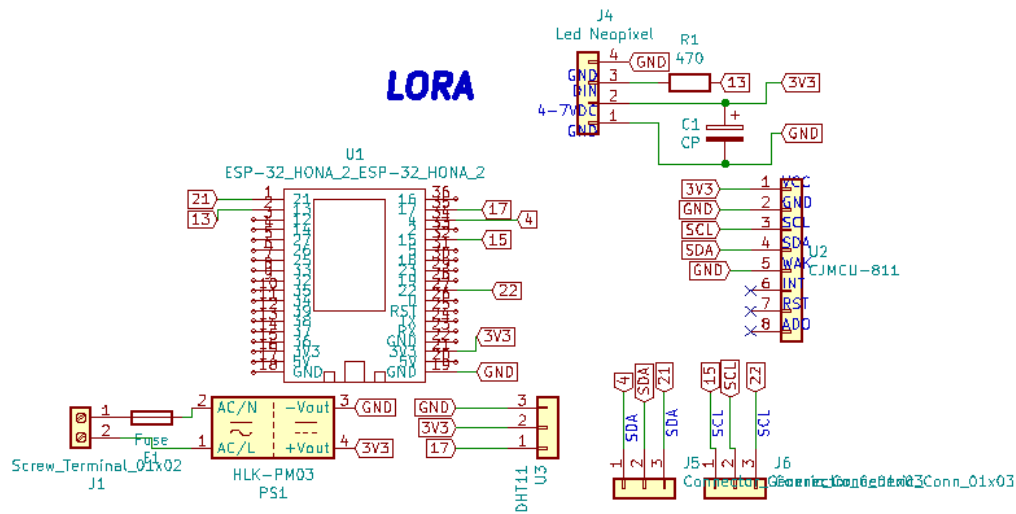
Gure osagai guztiak zenbateko elikadura behar dituen eta zenbat mA erabiltzen dituen. Taula honetan adierazten dugu.

PLAKAKO ELIKADURA					KANPOKO ELIKADURA		ERABILERA
VI N	5V	3V3	GND	RESE T	V	mA	
		X			3V3	20mA	TT-GO-ESP32
		X				4.5mA	CCS811 (CO2 Sentsorea)
		X				2.5mA	DHT11 (Tenperatura eta Hezetasuna Sentsorea)
		X				900mA	Neopixel
12 voltio					220	1100%20220mA	Bateria DC



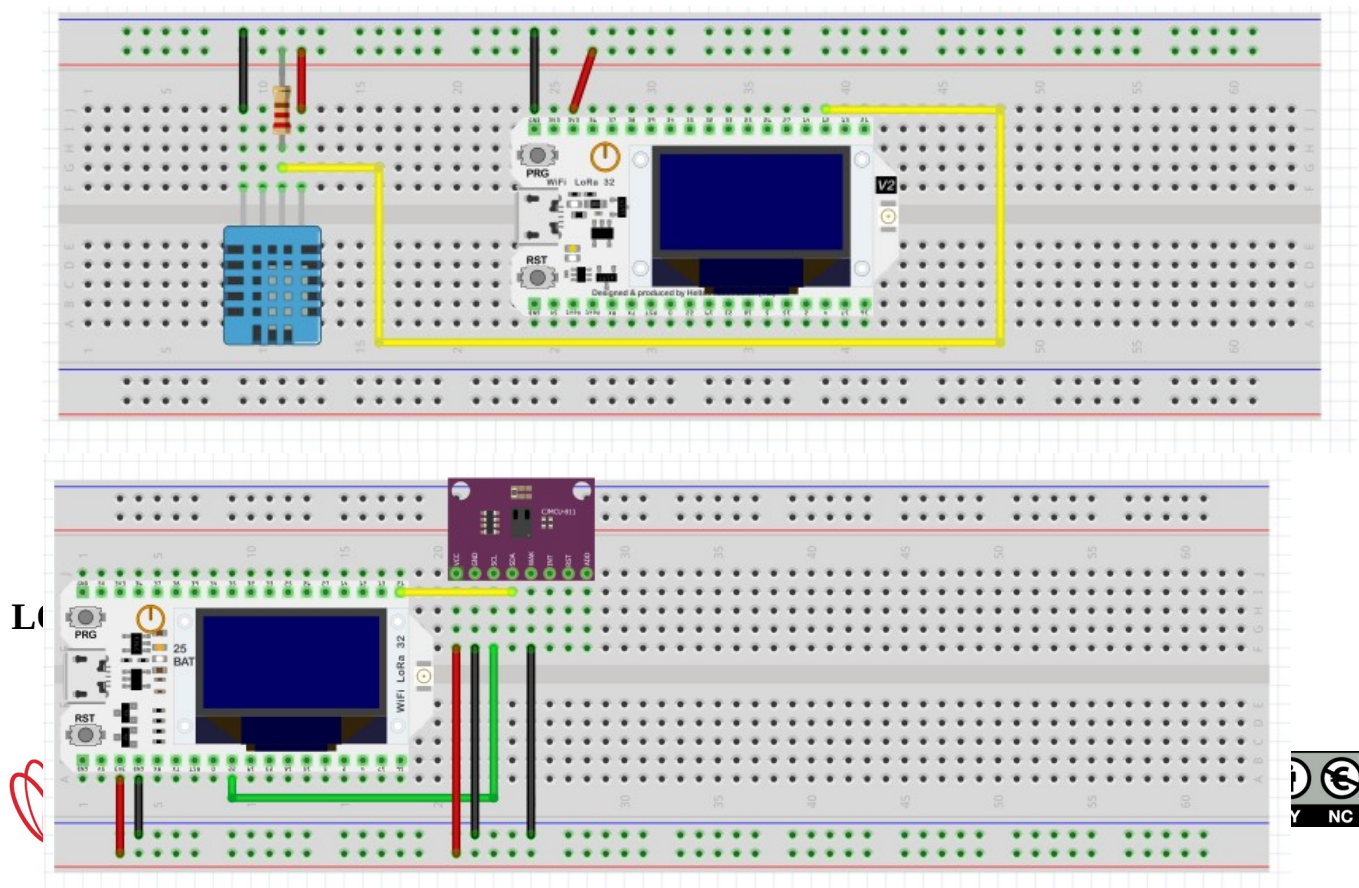
9 Eskema elektrikoak

Kicad Eskema:



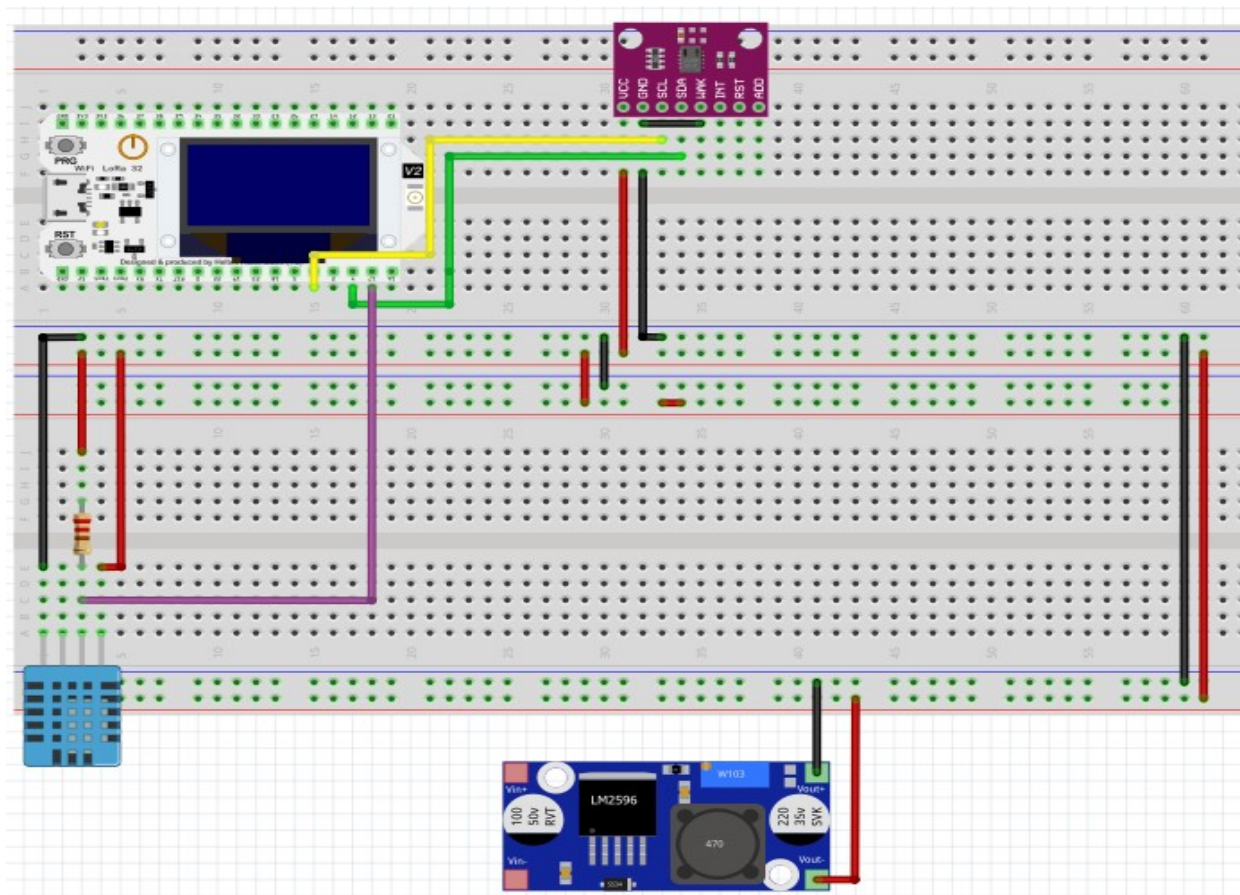
Fritzing eskemak

LoRa esp 32 eta DHT11 protoboard.

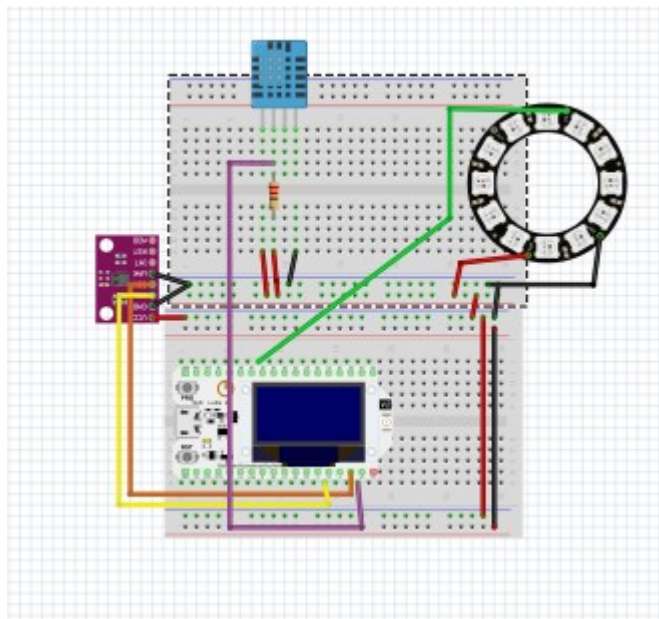




LORA esp 32 ,CCS811 eta DHT11 protoboard.



LORA esp 32 ,CCS811, DHT11 eta NEOPIXEL protoboard.



Material zerrenda

Erronka osoan erabiliko dugun material oso a.

Izena	Kopurua	Oharrak edota ezaugarri bereziak
DHT-11	1	3 pin
CO2 sentsoarea	1	
Neopixel	1x8	
Elikatze iturria DC	1	
ESP32 kontrolagailua	1	Nodo



10 10.Gorabeherak

- LORA ESP32-ko oled pantaila, ez dut lortzen beti datuak agertzea. Uste dut plakaren arazoa dela bai golperenbat hartu duelako, bai txarto soldeatuta dagoelako ... batzutan funtzionatzen du eta beste batzutan ez. Programa berdinarekin probatzen dugu.
- Lehenengo PCB placa txarto fresatuta, pin asko nahasiak eta orain ez doa ezer. Osagai batenbat apurtuta egon ahal da.





11 PROGRAMA NAGUSIA

Sentsoren bidalketa:

```
/*
Programaren izena: DHTtester_JON_MIKEL
Egilea: JON eta MIKEL   Data: 22/02/2021
Zeregina: DHT sentsorearekin tenperatura eta hezetazuna neurtzea.
Adibide hau domeinu publikokoa da.
Programaren egoera: Egiaztatzen/Egiaztatuta
Egiaztatutako plakak: hasiera batean Nano-arekin konprobatu dugu.
*/

#include "LoraGatewayHeader.h"
void setup() {
    Serial.begin(115200);
    DTH11sentsoreaSETUP ();
    OledPantallaSETUP();
    CO2sentsoreaSETUP ();
    tntSETUP ();
    NeopixelSETUP();
}

void loop() {
    DTH11sentsoreaLOOP ();
    OledPantallaLOOP ();
    CO2sentsoreaLOOP ();
    tntLOOP ();
    NeopixelLOOP ();
}
```

Bidalketa:

```
/* Funtzioa: void CO2sentsorea (void)
* Zeregina: ditugun datuak tnt-ra bidaltzea
* Erabilitako liburutegiak:
* Liburutegia nondik hartuta:
* Bueltatzen duena: ezer
* Sartutako parametroak:
*
* */
void BidalketaLOOP (void){

    float h = dht.readHumidity();
    // Read temperature as Celsius (the default)
    float t = dht.readTemperature();

    unsigned long balioa = ((t*10000000)+(h*10000)+(ccs.getCO2())); //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"));
}
```

CO2 sentsorea

```
/* Funtzioa: void CO2sentsorea (void)
 * Zeregina:CO2 sentsorean naurtzea du.
 * Erabilitako liburutegiak: DHT11DHT Sensor Library eta Adafruit
 * Liburutegia nondik hartuta://https://github.com/adafruit/Adafruit\_CCS811
 * Bueltatzen duena: ezer
 * Sartutako parametroak:
 *     int
 *
 * */
void CO2sentsoreaSETUP (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 CO2sentsorea (void)
 * Zeregina:CO2 sentsorean naurtzea du.
 * Erabilitako liburutegiak: DHT11DHT Sensor Library eta Adafruit
 * Liburutegia nondik hartuta://https://github.com/adafruit/Adafruit\_CCS811
 * Bueltatzen duena: ezer
 * Sartutako parametroak:
 *     int
 *
 * */
void CO2sentsoreaLOOP (void){
    if(ccs.available()){
        if(!ccs.readData()){
            //Serial.println(ccs.calculateTemperature());
            Serial.print("CO2: ");
            Serial.print(ccs.geteCO2());
            Serial.println("ppm");
            Serial.print("TVOC: ");
            Serial.println(ccs.getTVOC());
        }
        else{
            Serial.println("ERROR!");
            while(1);
        }
    }
    delay(1000);
}
```



DTH11 sentsorea:

```
/* Funtzioa: int DTH11sentsorea (int zeinPin)
 * Zeregina: DTH11 sentsorean temperatura eeta hezetasuna naurtzen du.
 * Erabilitako liburutegiak: DTH11DHT Sensor Library eta Adafruit
 * Liburutegia nondik hartuta: https://github.com/adafruit/Adafruit\_Sensor //
https://github.com/adafruit/DHT-sensor-library
 * Bueltatzen duena: ezer
 * Sartutako parametroak:
 *     int
 *
 * */
void DTH11sentsoreaSETUP (void){
    Serial.println(F("DHTxx test!"));

    dht.begin();
}

/* Funtzioa: int DTH11sentsorea (int zeinPin)
 * Zeregina: DTH11 sentsorean temperatura eeta hezetasuna naurtzen du.
 * Erabilitako liburutegiak: DTH11DHT Sensor Library eta Adafruit
 * Liburutegia nondik hartuta: https://github.com/adafruit/Adafruit\_Sensor //
https://github.com/adafruit/DHT-sensor-library
 * Bueltatzen duena: ezer
 * Sartutako parametroak:
 *     int
 *
 * */
void DTH11sentsoreaLOOP (void){

    // Wait a few seconds between measurements.
    delay(1000);

    // Reading temperature or humidity takes about 250 milliseconds!
    // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
    float h = dht.readHumidity();
    // Read temperature as Celsius (the default)
    float t = dht.readTemperature();
    // Read temperature as Fahrenheit (isFahrenheit = true)
    float f = dht.readTemperature(true);

    // Check if any reads failed and exit early (to try again).
    if (isnan(h) || isnan(t) || isnan(f)) {
        Serial.println(F("Failed to read from DHT sensor!"));
        return;
    }

    // Compute heat index in Fahrenheit (the default)
    float hif = dht.computeHeatIndex(f, h);
    // Compute heat index in Celsius (isFahreheit = false)
    float hic = dht.computeHeatIndex(t, h, false);

    Serial.print(F("hezetasuna: "));
    Serial.print(h);
    Serial.print(F("%  Temperatura: "));
    Serial.print(t);
    Serial.print(F("°C "));
    Serial.print(f);
    Serial.print(F("°F  Bero-indizea: "));
    Serial.print(hic);
    Serial.print(F("°C "));
    Serial.print(hif);
```



```
Serial.println(F("°F"));
```

```
}
```

Lora gateway:

```
/*-----( LIBURUTEGIAK )-----*/
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit\_Sensor
#include <DHT.h>
#include <DHT_U.h>
#include "DHT.h"
#define DHTPIN 17 // DHT sentsoareari lotutako Digital pin
#include "Adafruit_CCS811.h" // https://github.com/adafruit/Adafruit\_CCS811

#include <lmic.h>
#include <hal/hal.h>
#include <SPI.h>
//Libraries for LoRa
#include <LoRa.h>
//Libraries for OLED Display
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

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

/*-----( KONSTANTEAK )-----*/
#define DHTTYPE DHT11 // DHT 11
// #define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321
// #define DHTTYPE DHT21 // DHT 21 (AM2301)
//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 DI00 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
#define DELAYVAL 100 // Time (in milliseconds) to pause between pixels

// 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
/*-----( OBJETUAK )-----*/
DHT dht(DHTPIN, DHTTYPE);
Adafruit_CCS811 ccs;
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);
```



```
String LoRaData;
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
static const PROGMEM u1_t NWKSEKEY[16] = { 0x0E, 0x47, 0xF6, 0x3E, 0xF4, 0x6E, 0x65,
0x07, 0x9B, 0x41, 0x20, 0x16, 0x69, 0x2C, 0x09, 0xBD };

static const u1_t PROGMEM APPSEKEY[16] = { 0x1B, 0x54, 0x8E, 0xAC, 0x66, 0x0F, 0x32,
0x0A, 0xAB, 0x21, 0x09, 0x04, 0x46, 0x1C, 0x08, 0x6A };

static const u4_t DEVADDR = 0x26011E4D ; // <-- Change this address for every node!
/*-----( FUNTZIOAK )-----*/
void CO2sentsoreaSETUP (void);
void CO2sentsoreaLOOP (void);
void DTH11sentsoreaSETUP (void);
void DTH11sentsoreaLOOP (void);
void tntSETUP (void);
void tntLOOP (void);
void OledPantallaSETUP (void);
void OledPantallaLOOP (void);
void NeopixelSETUP (void);
void NeopixelLOOP (void);
//BIDALKETA
void BidalketaLOOP (void);
void os_getArtEui (u1_t* buf) { }
void os_getDevEui (u1_t* buf) { }
void os_getDevKey (u1_t* buf) { }

static uint8_t mydata[] = "Hello, world!";
static osjob_t sendjob;
const unsigned TX_INTERVAL = 18000;

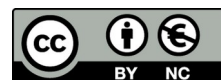
// Pin mapping
const lm323_pinmap lm323_pins = {
    .nss = 18,
    .rxtx = LM323_UNUSED_PIN,
    .rst = 14,
    .dio = {26, 33, 32},
};
```

Neopixel:

```
/* Funtzioa: void NeopixelSETUP (void)
 * Zeregina: Neopixela eta bere argiak zehaztea.
 * Erabilitako liburutegiak:
 * Liburutegia nondik hartuta:
 * Bueltatzen duena: ezer
 * Sartutako parametroak:
 *
 * */

void NeopixelSETUP (void)
{
    // 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)
}
/* Funtzioa: void NeopixelLOOP(void)
 * Zeregina:
```




```

* Erabilitako liburutegiak:
* Liburutegia nondik hartuta: pin zehaztu
* Bueltatzen duena: ezer
* Sartutako parametroak:
*
*
* */
void NeopixelLOOP(void)
{
  pixels.clear(); // Set all pixel colors to 'off'

  // The first NeoPixel in a strand is #0, second is 1, all the way up
  // to the count of pixels minus one.
  //for(int i=0; i<8; i++) { // For each pixel...

    //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(50, 80, 0));
    // pixels.setPixelColor(1, pixels.Color(50, 0, 0));
    // pixels.setPixelColor(2, pixels.Color(0, 50, 0));
    // pixels.setPixelColor(3, pixels.Color(0, 0, 50));
    // pixels.setPixelColor(4, pixels.Color(50, 50, 0));
    // pixels.setPixelColor(5, pixels.Color(50, 0, 50));
    // pixels.setPixelColor(6, pixels.Color(0, 50, 50));
    //pixels.setPixelColor(7, pixels.Color(50, 50, 0));

    if(!ccs.readData()){
      //Serial.println(ccs.calculateTemperature());
      Serial.print("CO2: ");
      Serial.print(ccs.getCO2());
      Serial.println("ppm");
      Serial.print("TVOC: ");
      Serial.println(ccs.getTVOC());
    }
    else{
      Serial.println("ERROR!");
      while(1);
    }
  }

  if (ccs.getCO2()>=400 & ccs.getCO2()<750){
    pixels.setPixelColor(0, pixels.Color(0, 50, 0));
    pixels.setPixelColor(1, pixels.Color(0, 50, 0));
    pixels.setPixelColor(2, pixels.Color(0, 50, 0));
  }
  else {
    if (ccs.getCO2()>=750 & ccs.getCO2()<1100){
      pixels.setPixelColor(0, pixels.Color(0, 50, 0));
      pixels.setPixelColor(1, pixels.Color(0, 50, 0));
      pixels.setPixelColor(2, pixels.Color(0, 50, 0));
      pixels.setPixelColor(3, pixels.Color(50, 50, 0));
      pixels.setPixelColor(4, pixels.Color(50, 50, 0));
      pixels.setPixelColor(5, pixels.Color(50, 50, 0));
    }
    else {
      pixels.setPixelColor(0, pixels.Color(0, 50, 0));
      pixels.setPixelColor(1, pixels.Color(0, 50, 0));
      pixels.setPixelColor(2, pixels.Color(0, 50, 0));
      pixels.setPixelColor(3, pixels.Color(50, 50, 0));
      pixels.setPixelColor(4, pixels.Color(50, 50, 0));
      pixels.setPixelColor(5, pixels.Color(50, 50, 0));
      pixels.setPixelColor(6, pixels.Color(50, 0, 0));
      pixels.setPixelColor(7, pixels.Color(50, 0, 0));
    }
  }
}

```



```

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

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

```

Oled pantaila:

```

/* Funtzioa: void OledPantallaSETUP (void){
 * Zeregina: oled pantalla datuak agertzea
 * Erabilitako liburutegiak:
 * Liburutegia nondik hartuta:
 * Bueltatzen duena: ezer
 * Sartutako parametroak:
 *
 *
 * */
void OledPantallaSETUP (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("Kargatzen...");
    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();
}

/* Funtzioa: void OledPantallaLOOP (void) {
 * Zeregina:

```



```

* Erabilitako liburutegiak:
* Liburutegia nondik hartuta:
* Bueltatzen duena: ezer
* Sartutako parametroak:
*
* */
void OledPantallaLOOP (void) {
    float h = dht.readHumidity();
    // Read temperature as Celsius (the default)
    float t = dht.readTemperature();

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

    // Dsisplay information
    display.clearDisplay();
    display.setCursor(0,0);
    display.print("MIKEL_JON");
    display.setCursor(0,20);
    display.print("TenperaturaHezetasuna");
    display.setCursor(0,30);
    display.print(t);
    display.setCursor(70,30);
    display.print(h);
    display.setCursor(0,40);
    display.print("CO2_TVOC");
    display.setCursor(0,50);
    display.print(ccs.geteCO2());
    display.setCursor(40,50);
    display.print(ccs.getTVOC());
    display.display();
}

```

Ttn:

```

/* Funtzioa: onEvent (ev_t ev)
* Zeregina:
* Erabilitako liburutegiak: https://github.com/adafruit/Adafruit\_Sensor
* https://github.com/adafruit/DHT-sensor-library
* Liburutegia nondik hartuta: Internetetik
* Bueltatzen duena: ezer
* Sartutako parametroak:
* void
*
* */

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;
}
}

```



```

/* Funtzioa: void CO2sentsorea (void)
 * Zeregina:CO2 sentsorean naurtzea du.
 * Erabilitako liburutegiak: DTH11DHT Sensor Library eta Adafruit
 * Liburutegia nondik hartuta://https://github.com/adafruit/Adafruit\_CCS811
 * Bueltatzen duena: ezer
 * Sartutako parametroak:
 *     int
 *
 * */
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 tntSETUP (void){

    Serial.println(F("Starting"));

#ifdef VCC_ENABLE
    // For Pinoccio Scout boards
    pinMode(VCC_ENABLE, OUTPUT);
    digitalWrite(VCC_ENABLE, HIGH);
    delay(1000);
#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

#ifdef 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_SF9;

    // 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 tntLOOP (void){
    os_runloop_once();
}

```



12 Egindako sistemaren argazkia

