

# Senzor LoRaWaN

Miron Cezar Andrei

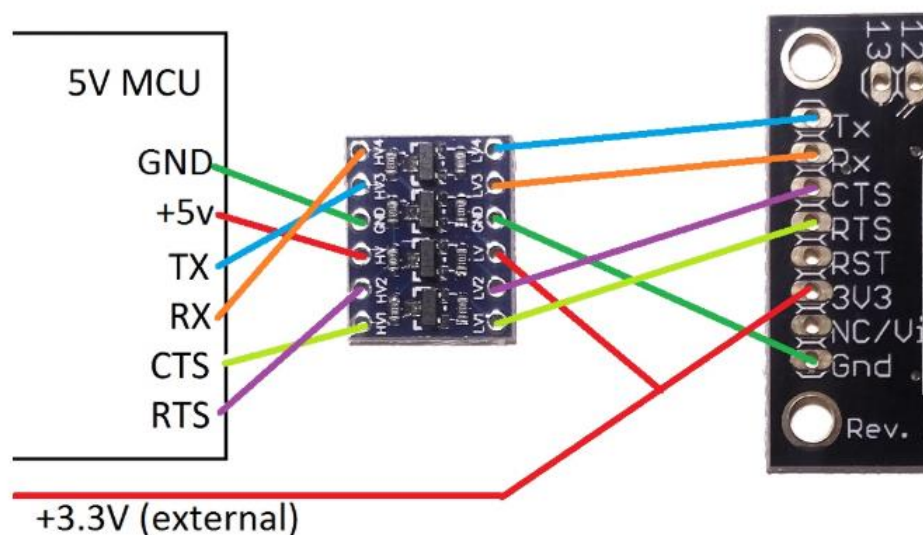
Pentru realizarea proiectului, am folosit următoarele componente :

- Arduino Uno
- Modul RN2483
- Senzor temperatură și umiditate – DHT11
- Breadboard + fire de interconectare

Etape de proiectare :

1. Realizarea comunicației între Arduino și RN2483

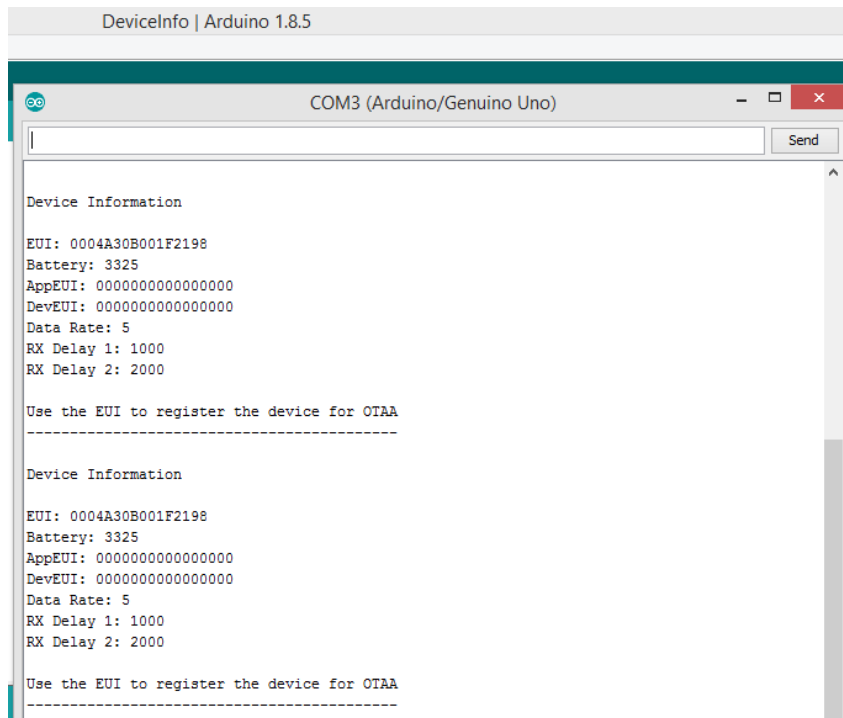
## 5v Microcontroller with external 3.3v (no regulator installed)



Placa de dezvoltare Arduino UNO funcționează în logică de 5V. Și deci, „1” logic în arduino reprezintă 5V în timp ce „0” logic este reprezentat de tensiunea de 0V. Chipul ales pentru a realiza transferul de date folosind protocolul LoraWan este RN2483. Acesta funcționează în logica de 3V3 și de aceea am fost nevoit să folosesc un convertor de nivel.

2. Conectarea cu mediul TTN

După ce am verificat conexiunile fizice făcute, am încărcat pe placa de dezvoltare un cod ce identifică informații despre chipul lora, informații ce le voi folosi ulterior pentru a facilita conexiunea cu platforma TTN.



Cod test:

```
#include <TheThingsNetwork.h>
```

```
#include <SoftwareSerial.h>
```

```
//#define loraSerial Serial1
```

```
#define debugSerial Serial
```

```
SoftwareSerial loraSerial(10, 11);
```

```
// Replace REPLACE_ME with TTN_FP_EU868 or TTN_FP_US915
```

```
#define freqPlan TTN_FP_EU868
```

```
TheThingsNetwork ttn(loraSerial, debugSerial, freqPlan);
```

```
void setup()
```

```
{
```

```
  loraSerial.begin(57600);
```

```
  debugSerial.begin(9600);
```

```
}
```

```
void loop()
```

```
{
```

```
  debugSerial.println("Device Information");
```

```
  debugSerial.println();
```

```
  ttn.showStatus();
```

```
  debugSerial.println();
```

```
  debugSerial.println("Use the EUI to register the device for OTAA");
```

```
  debugSerial.println("-----");
```

```
  debugSerial.println();
```

```
  delay(5000);
```

```
}
```

Îndată ce am aflat cele necesare, le voi folosi pentru a-mi înregistra noul meu dispozitiv pe platforma TheThingsNetwork.

**REGISTER DEVICE**[bulk import devices](#)

**Device ID**  
This is the unique identifier for the device in this app. The device ID will be immutable.  

rn2483

**Device EUI**  
The device EUI is the unique identifier for this device on the network. You can change the EUI later.  

<> 00 04 A3 0B 00 1F 21 98 8 bytes

**App Key**  
The App Key will be used to secure the communication between you device and the network.  

this field will be generated

**App EUI**  

70 B3 D5 7E D0 00 F1 54

CancelRegister

După înregistrare vom avea :

**DEVICE OVERVIEW**

**Application ID** proiectiulian

**Device ID** rn2483

**Activation Method** OTAA

**Device EUI** <> 00 04 A3 0B 00 1F 21 98

**Application EUI** <> 70 B3 D5 7E D0 00 F1 54

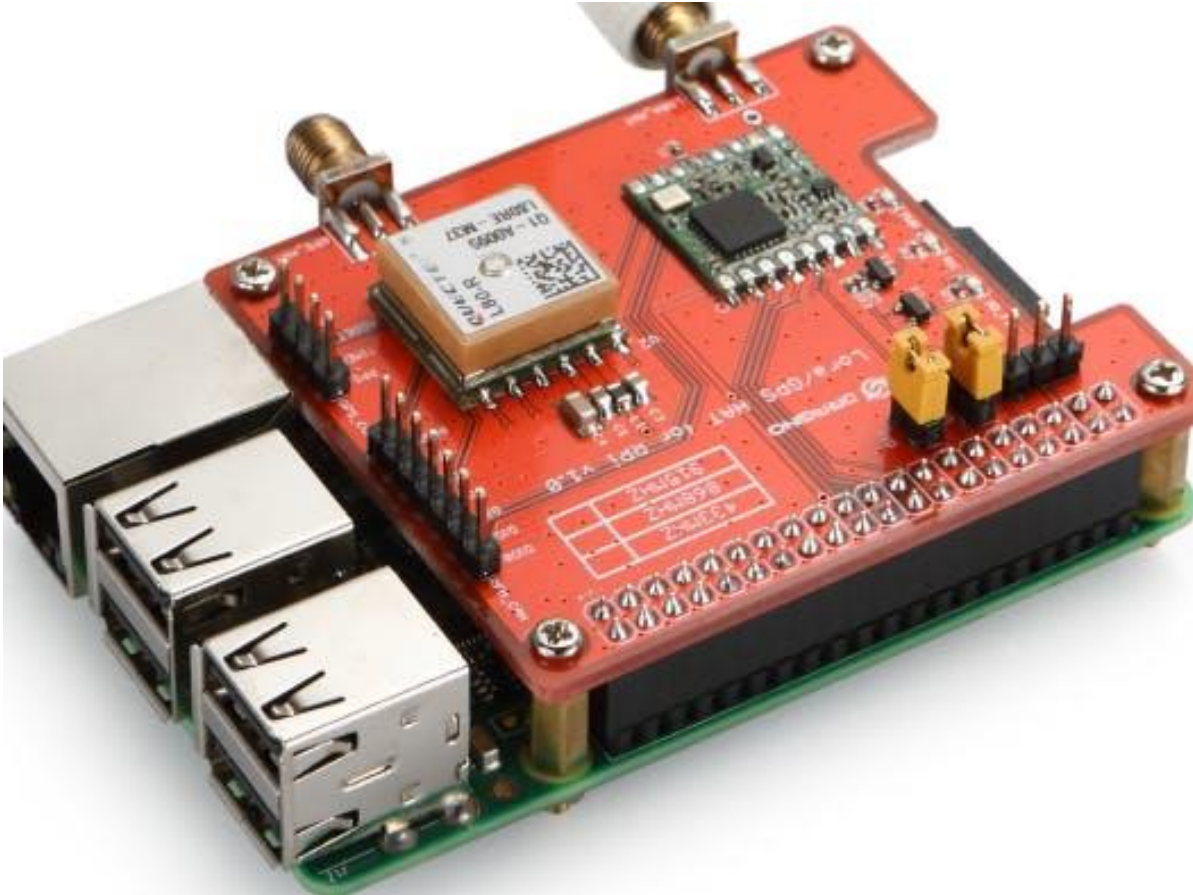
**App Key** <> .....

**Status** never seen

**Frames up** 0 [reset frame counters](#)

**Frames down** 0

Îndată ce device-ul a fost înregistrat, testăm conexiunea acestuia cu unul din gateway-urile LoRaWAN disponibile. Pentru a testa acest lucru, m-am folosit de un gateway creat cu ajutorul plăcii de dezvoltare Raspberry Pi3 și un modul Lora Hat.



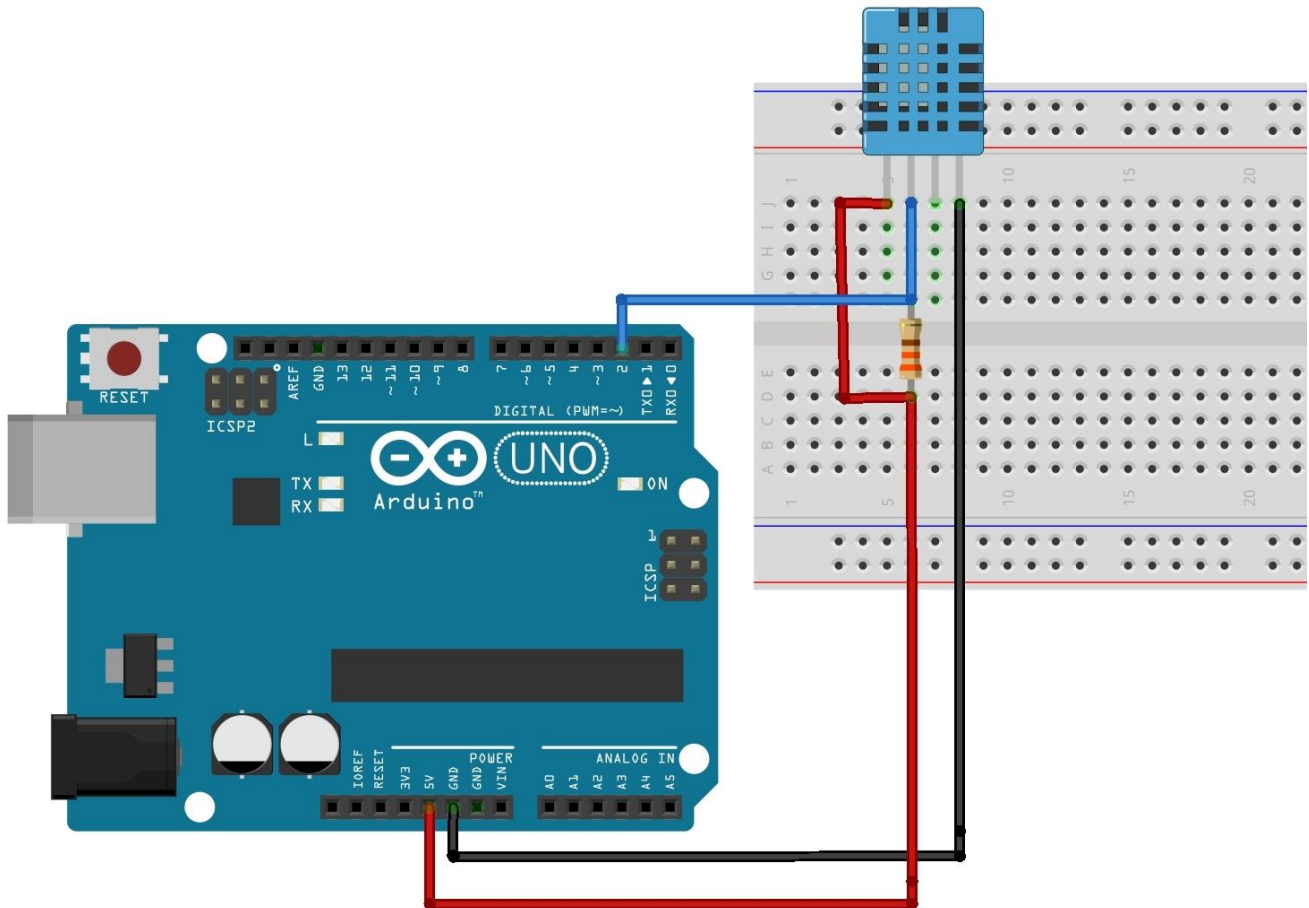
Metoda de activare a dispozitivului este ABP și se bazează pe Device Address, Network Session Key și App Session Key.

```
1 const char *devAddr = "2601172D";  
2 const char *nwksKey = "E2D9DD32BF0D1751168F34DE[REDACTED]";  
3 const char *appSKey = "CCF9364C493AFA4707A47A[REDACTED]";
```

Folosindu-mă de aceste adrese am reușit să stabilesc legătura între Arduino + RN2483 și platforma TheThingsNetwork.

### 3. Conexiunea senzorului DHT11 cu Arduino

- VCC – 5V
- GND – GND
- SIG – pin 7



Pentru citirea datelor oferite de către senzor, m-am folosit de biblioteca SimpleDht.

### 4. Implementarea codului și transmiterea datelor

În această etapă a proiectului, am integrat toate componentele sistemului și am transmis prin protocolul LoraWan umiditatea și temperatura relativă.

Acestea sunt trimise în hexazecimal, conversia în decimal făcându-se pe platforma TTN.

```
// Send it off  
ttn.sendBytes(payload, sizeof(payload));
```

```
2800 *C, 4500 H
-- LOOP
Sending: mac tx uncnf 1 0AF01194
Successful transmission
Sample OK: 28 *C, 45 H

2800 *C, 4500 H
-- LOOP
Sending: mac tx uncnf 1 0AF01194
Successful transmission
Sample OK: 28 *C, 44 H

2800 *C, 4400 H
-- LOOP
Sending: mac tx uncnf 1 0AF01130
Successful transmission
Sample OK: 28 *C, 44 H

2800 *C, 4400 H
-- LOOP
Sending: mac tx uncnf 1 0AF01130
Successful transmission
Sample OK: 28 *C, 44 H

2800 *C, 4400 H
-- LOOP
Sending: mac tx uncnf 1 0AF01130
Successful transmission
```

- Consola Arduino – metoda de transimie a datelor

APPLICATION DATA

pause

clear

Filters

uplink

downlink

activation

ack

error

	time	counter	port			
▲	00:00:46	54	1	payload: 0A F0 11 30	temperatura: 28	umiditatea: 44
▲	00:00:05	46	1	payload: 0A F0 11 94	temperatura: 28	umiditatea: 45
▲	23:59:26	39	1	payload: 0A F0 12 5C	temperatura: 28	umiditatea: 47
▲	23:58:36	30	1	payload: 0B 54 15 18	temperatura: 29	umiditatea: 54
▲	23:57:54	22	1	payload: 0B 54 17 D4	temperatura: 29	umiditatea: 61
▲	23:57:13	14	1	payload: 0B 54 15 E0	temperatura: 29	umiditatea: 56

- \* Consola TTN – metoda de primire a datelor

## Payload TTN :

**PAYLOAD FORMATS**

**Payload Format**  
The payload format sent by your devices

Custom

decoder

converter

validator

encoder

remove decoder

```
1 function Decoder(bytes, port) {
2   var temperature = (bytes[0] << 8) | bytes[1];
3   var humidity = (bytes[2] << 8) | bytes[3];
4
5   return {
6     temperatura: temperature/100,
7     umiditatea: humidity/100
8   }
9 }
10
```

decoder has no changes

## Codul aplicației :

```
#include <TheThingsNetwork.h>
```

```
#include <SoftwareSerial.h>
```

```
#include <SimpleDHT.h>
```

```
// Set your DevAddr, NwkSKey, AppSKey and the frequency plan
```

```
const char *devAddr = "2601172D";
```

```
const char *nwkSKey =
"E2D9DD32BF0D1751168F34DB74C3CC4B";
```

```
const char *appSKey =
"CCF9364C493AFA4707A47A4C21DC4307";
```

```
//#define loraSerial Serial1
```

```
SoftwareSerial loraSerial(10, 11);
```

```
#define debugSerial Serial
```

```
// Replace REPLACE_ME with TTN_FP_EU868 or
TTN_FP_US915
```

```
#define freqPlan TTN_FP_EU868
```

```
TheThingsNetwork ttn(loraSerial, debugSerial, freqPlan);
```

```
int pinDHT11 = 7;
```

```
SimpleDHT11 dht11;
```

```
void setup()
```

```
{
```

```
  loraSerial.begin(57600);
```

```
  debugSerial.begin(9600);
```

```
// Wait a maximum of 10s for Serial Monitor
```

```
while (!debugSerial && millis() < 10000)
```

```
;
```

```
debugSerial.println("-- PERSONALIZE");
```

```
ttn.personalize(devAddr, nwkSKey, appSKey);
```



```

debugSerial.println("-- STATUS");

ttn.showStatus();                                     // Send it off
}                                                       ttn.sendBytes(payload, sizeof(payload));

void loop()                                           delay(3000);
{                                                       }

byte temperature = 0;

byte humidity = 0;

int err = SimpleDHTErrSuccess;

if ((err = dht11.read(pinDHT11, &temperature, &humidity,
NULL)) != SimpleDHTErrSuccess) {

    Serial.print("Read DHT11 failed, err=");
    Serial.println(err);delay(1000);

    return;
}

Serial.print("Sample OK: ");

Serial.print((int)temperature); Serial.print(" *C, ");

Serial.print((int)humidity); Serial.println(" H");

uint32_t temp = (int)temperature * 100;

uint32_t hum = (int)humidity * 100;

Serial.println();

Serial.print(temp); Serial.print(" *C, ");

Serial.print(hum); Serial.println(" H");

debugSerial.println("-- LOOP");

// Prepare payload of 1 byte to indicate LED status

byte payload[4];

payload[0] = highByte(temp);

payload[1] = lowByte(temp);

payload[2] = highByte(hum);

payload[3] = lowByte(hum);

```