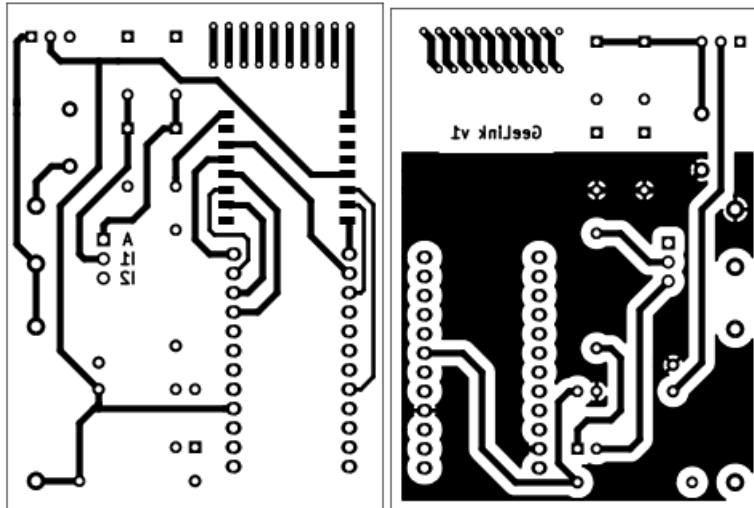


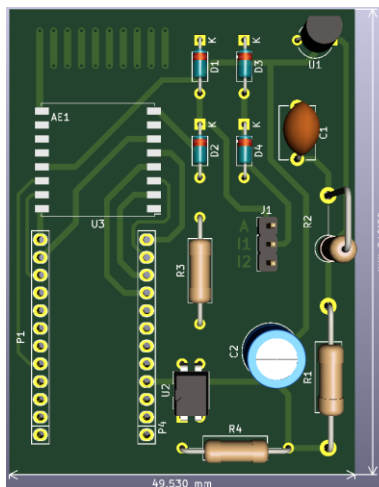
❖ GeeLink Tutorials

Hardware:

Print PCB design by KiCad https://github.com/hovuduybao/GeeLink/tree/master/KiCad/GeeLink_v1/PCB



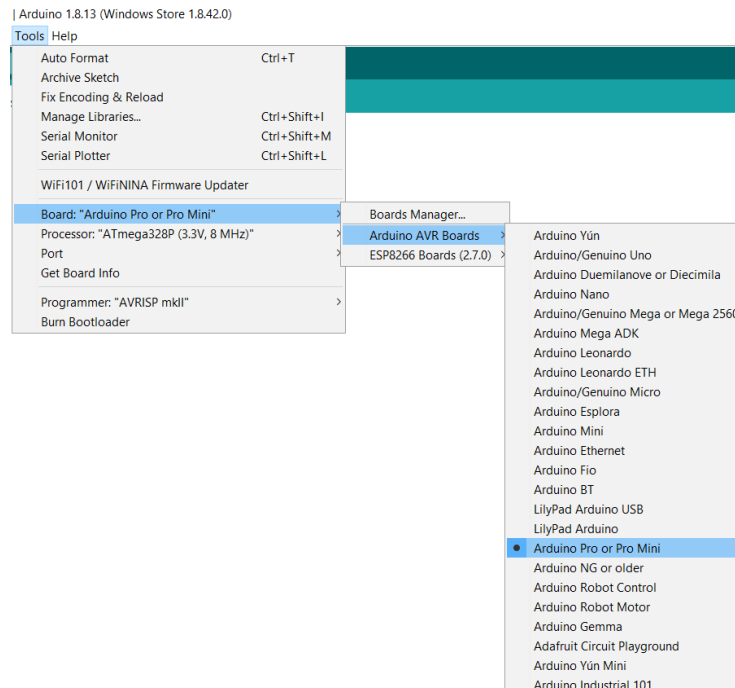
Solder the components following this 3D model



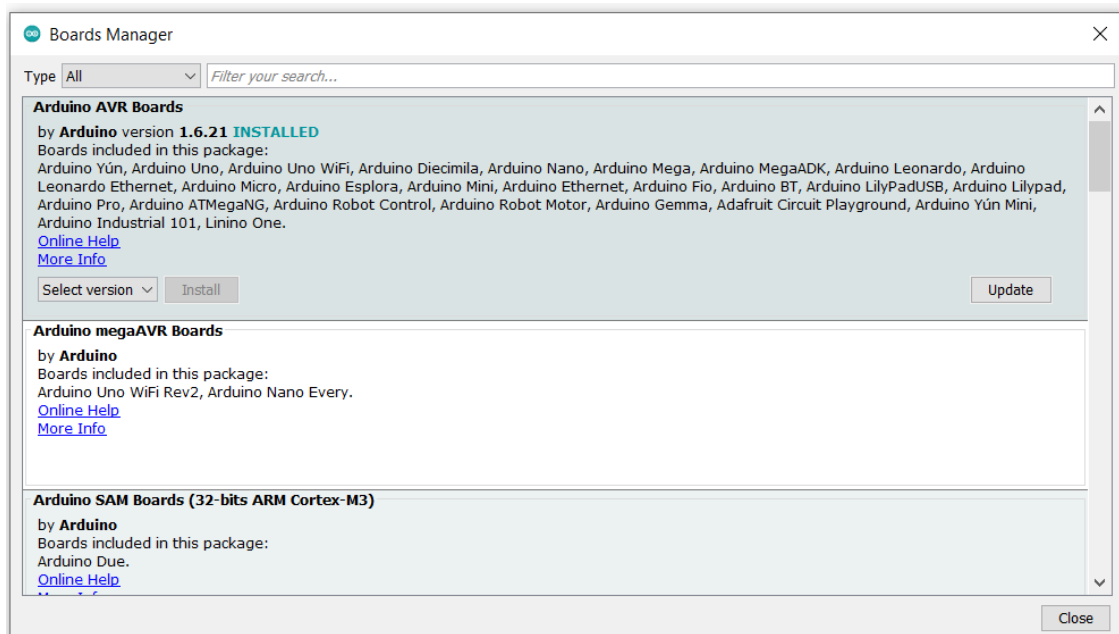
Connect 3 pins (I1, I2 and A) from TIC to the PCB board.

Software:

- Arduino IDE:
 - Setup board information:



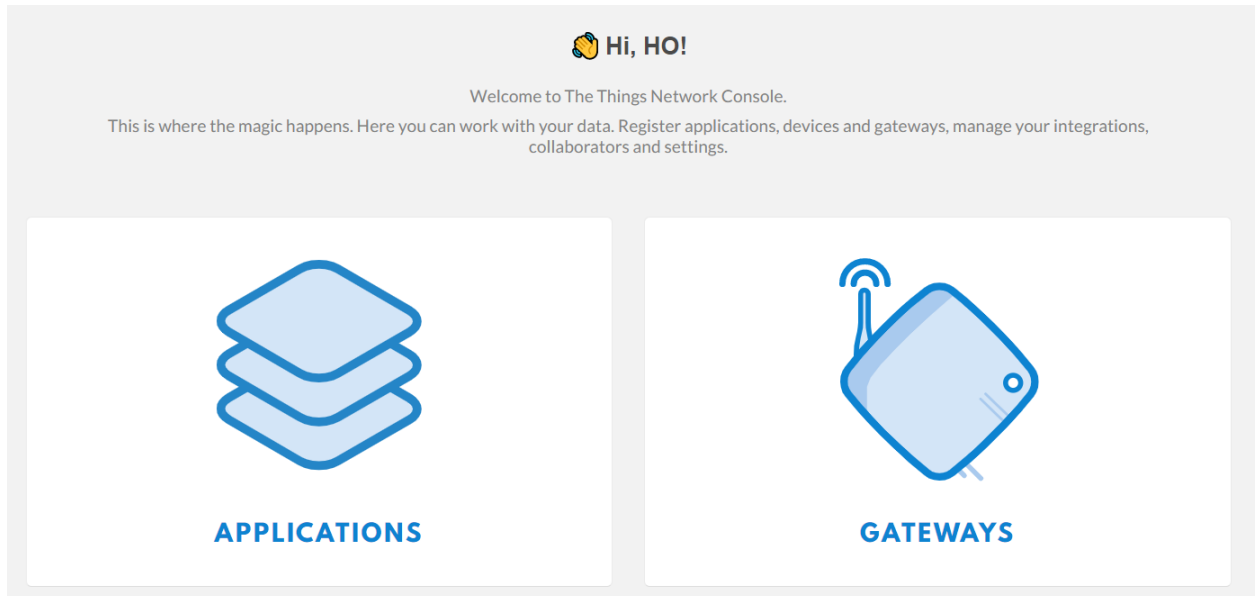
- Install ver **1.6.21** on Boards Manager (the higher versions have the bugs for LMIC library)



- Select Processor: "ATmega328P (3.3V, 8MHz)" and Port that is connecting to ProMini
- Install Libraries following this instruction: <https://www.arduino.cc/en/guide/libraries>

The Things Network (TTN)

- Login to TTN console: <https://console.thethingsnetwork.org/> and select Application



- Add a new Application

Applications > Add Application

ADD APPLICATION

Application ID
The unique identifier of your application on the network

geelink_tutorial ✓

Description
A human readable description of your new app

Eg. My sensor network application ✓

Application EUI
An application EUI will be issued for The Things Network block for convenience, you can add your own in the application settings page.


EUI issued by The Things Network

Handler registration
Select the handler you want to register this application to

ttn-handler-eu ✓

Cancel Add application


- Register a device

Applications >  geelink_tutorial > Devices


Overview **Devices** Payload Formats Integrations Data Settings

REGISTER DEVICE [bulk import devices](#)


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

geelink_1 

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

 this field will be generated

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 03 26 FA

Cancel **Register**

- After registering a device, adjust the keys into this format

Applications >  geelink_tutorial > Devices >  geelink_1





Overview **Data** Settings





DEVICE OVERVIEW






Application ID geelink_tutorial

Device ID geelink_1

Activation Method OTAA

Device EUI    { 0x7A, 0x0C, 0x11, 0x50, 0x3F, 0xC6, 0x60, 0x00 } 

Application EUI    { 0xFA, 0x26, 0x03, 0xD0, 0x7E, 0xD5, 0xB3, 0x70 } 

App Key     

Status never seen

Frames up 0 [reset frame counters](#)

Frames down 0

Payload decoder on TTN

Applications > geelink_tutorial > Payload Formats

Overview Devices **Payload Formats** Integrations Data Settings

PAYLOAD FORMATS

Payload Format
The payload format sent by your devices

Custom

decoder converter validator encoder [remove decoder](#)

```

1 function Decoder(bytes, port) {
2   // Decode an uplink message from a buffer
3   // (array) of bytes to an object of fields.
4   var decoded = {};
5
6   // if (port === 1) decoded.Led = bytes[0];
7
8   return decoded;
9 }

```

decoder has no changes

- Change the decoder by the GeeLink_decoder following this link and **remember to Save**
https://github.com/hovuduybao/GeeLink/blob/master/GeeLink_Decoder_TTN/GeeLink_Decoder_TTN.js
- Useful Links:
 - <https://github.com/myDevicesIoT/cayenne-docs/blob/master/docs/LORA.md>
 - <https://doc.rakwireless.com/rak7200-lora-tracker/analyzing-the-data-from-rak7200>


Integrations

- Add this integration:

Applications > geelink_tutorial > Integrations

Overview Devices Payload Formats **Integrations** Data Settings

ADD INTEGRATION



Data Storage (v2.0.1)
The Things Industries B.V.

Stores data and makes it available through an API. Your data is stored for seven days.

Cancel [Add integration](#)

Application Server

- Install InfluxDB and Grafana following this instruction:

<https://github.com/ITU-PITLab/public/blob/master/TheThingsNetwork%2Bnode-red%2Binfluxdb%2Bgrafana.md>

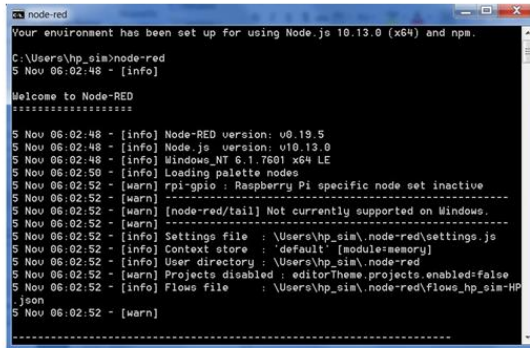
- Install Node-RED

If Git is not installed on your PC: <https://git-scm.com/downloads> After that, run this command on Gitbash

npm install -g --unsafe-perm node-red

Connecting to TTN

- Start NODE.js command prompt
- Run: node-red
- Open your web browser and go to <http://127.0.0.1:1880>



```

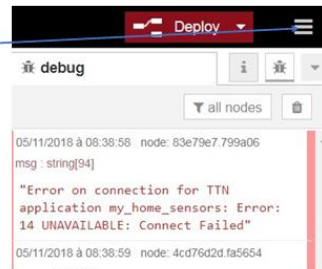
node-red
Your environment has been set up for using Node.js 10.13.0 (x64) and npm.

C:\Users\hpi\sim\node-red
$ Nov 06 02:46 - [info]
Welcome to Node-RED
-----
$ Nov 06 02:46 - [info] Node-RED version: v0.19.5
$ Nov 06 02:46 - [info] Node.js version: v10.13.0
$ Nov 06 02:46 - [info] Windows_NT 6.1.7601 x64 LE
$ Nov 06 02:50 - [info] Loading palette nodes
$ Nov 06 02:52 - [warn] rpi-gpio : Raspberry Pi specific node set inactive
$ Nov 06 02:52 - [warn] -----
$ Nov 06 02:52 - [warn] [node-red/tail] Not currently supported on Windows.
$ Nov 06 02:52 - [warn] -----
$ Nov 06 02:52 - [info] Settings file : \Users\hpi\sim\node-red\settings.js
$ Nov 06 02:52 - [info] Context store : default (module=memory)
$ Nov 06 02:52 - [info] User directory : \Users\hpi\sim\node-red
$ Nov 06 02:52 - [warn] Projects disabled : editorTheme.projects.enabled=false
$ Nov 06 02:52 - [info] Flows file : \Users\hpi\sim\node-red\flows_hpi\sim-HP
.json
$ Nov 06 02:52 - [warn]

```

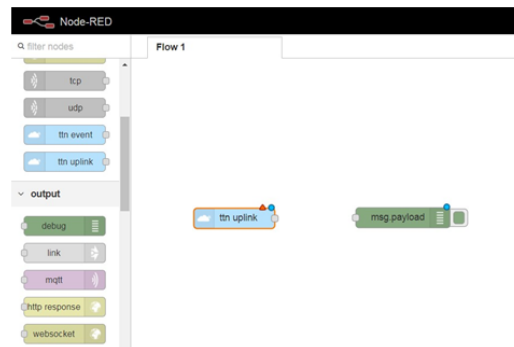
- On the editor, click here
- and go to palette editor
- Install:

- node-red-contrib-ttn
- node-red-contrib-influxdb



Connecting to TTN

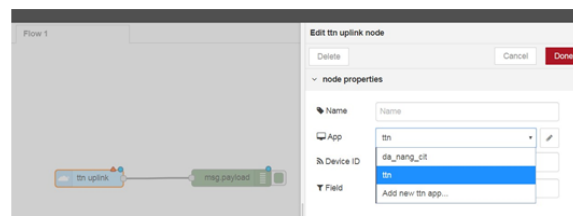
- You have the graphical Node-red editor
- Add ttn uplink and a debug output
- Edit TTN uplink
- Choose « Add new ttn app ... » in App and click on edit



App ID:

Access Key:

Discovery address:



Connecting to TTN

- You need :

- App ID :
- Access Key :
- Discovery address :

discovery.thethingsnetwork.org:1900

- Go to your application in TTN
- Copy past the Application ID and Access Key

Edit ttn uplink node > Edit ttn app node

Delete Cancel Update

App ID 70B3D57ED0013EAD

Access Key

Discovery address discovery.thethingsnetwork.org:1900

Application Overview

Application ID my_home_sensors

Description my home sensors

Created 7 months ago

Register on thethingsnetwork.org

Application EUIs

DEVICES

COLLABORATORS

ACCESS KEYS

Connecting to TTN

- Click on Deploy
- Your uplink TTN should be connected
- Click on debug window
- You will receive the packet of the application
- If you want to filter only your device, add your device ID
- Click here:

Deploy

debug

05/11/2019 à 06:17:40 node: ba3a9ed9 e50308

msg.payload: Object

{ analog_in_3: 5.14, digital_out_4: 0, relative_humidity_2: 79, temperature_1: 21.2 }

DEVICE OVERVIEW

Application ID my_home_sensors

Device ID 50ff1a0000010004

Description Light test

Activation Method OTAA

Edit ttn uplink node

Delete Cancel Done

node properties

Name my_home_sensors

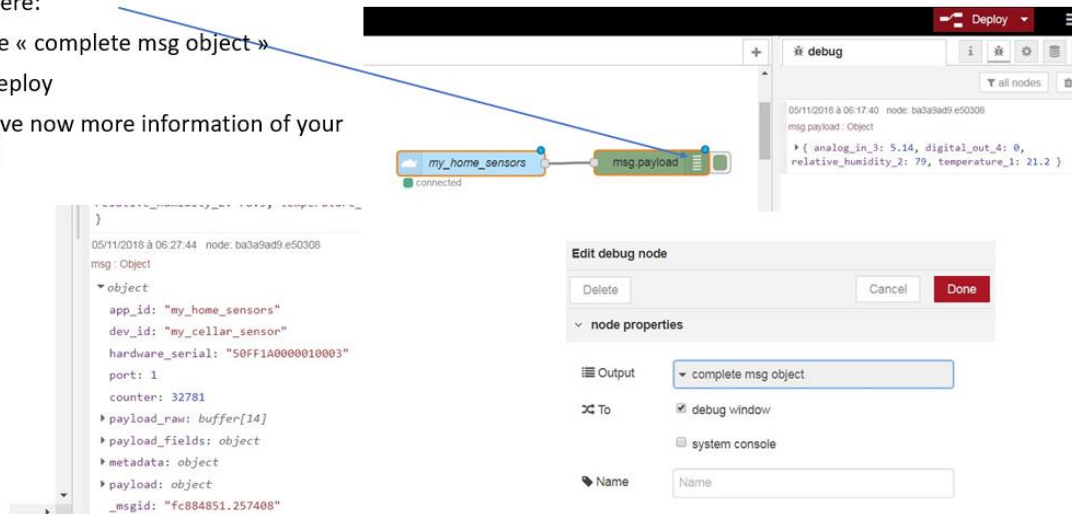
App my_home_sensors

Device ID 50ff1a0000010004

Field

Connecting to TTN

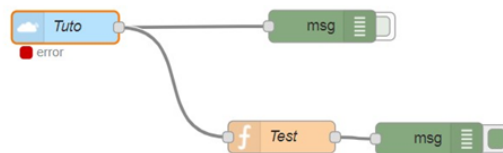
- Click here:
- Choose « complete msg object »
- And Deploy
- You have now more information of your uplink



The screenshot shows the TTN console interface. On the left, a node configuration for 'my_home_sensors' is displayed, including fields like 'app_id', 'dev_id', 'hardware_serial', 'port', 'counter', 'payload_raw', 'payload_fields', 'metadata', 'payload', and '_msgid'. On the right, a 'debug' window shows a log entry for a message payload. Below the debug window, an 'Edit debug node' dialog is open, showing 'node properties' with 'Output' set to 'complete msg object', 'To' set to 'debug window', and 'Name' set to 'Name'.

Connecting to TTN

- If you want to extract only 1 data,
- as an example the RSSI (received signal Strength indicator)
- Use a function to extract the wanted data



```
return {
  // Some fields from the metadata freq:
  msg.metadata.frequency,
  cr: msg.metadata.cr,
  dr: msg.metadata.dr,

  // Combine RSSI and SNR of all gateways into two arrays:
  rssi: gateways.map(gw => gw.rssi),
  snr: gateways.map(gw => gw.snr),

};
```

InfluxDB

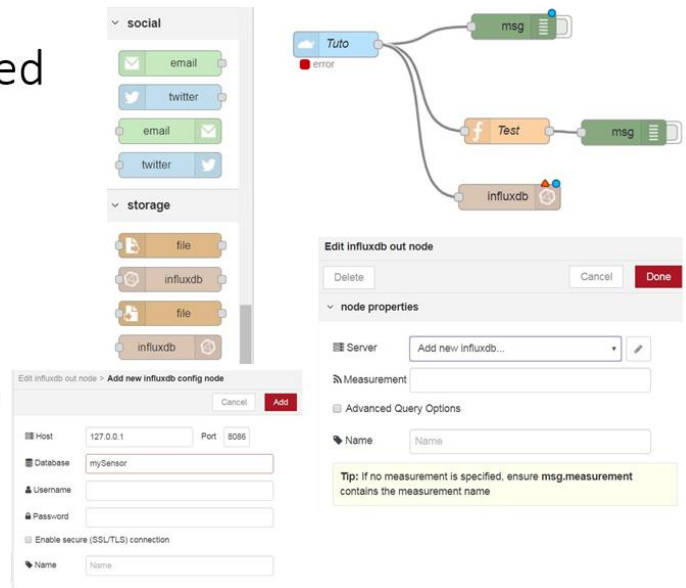
- Run « influxd.exe », it will start the database
- Run « influx.exe », it will open a shell
- Write: « CREATE DATABASE mySensor »
- Then write: « SHOW DATABASES »

```
> CREATE DATABASE mySensor
> SHOW DATABASES
name: databases
name
----
_internal
tuto
mySensor
>
```

Your database is created

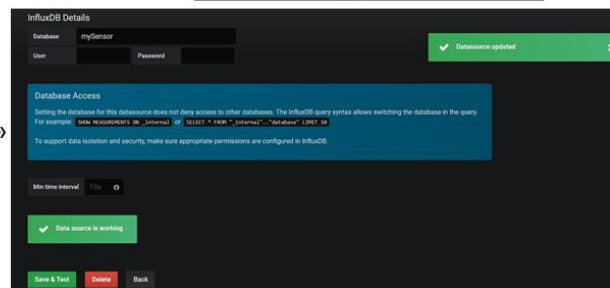
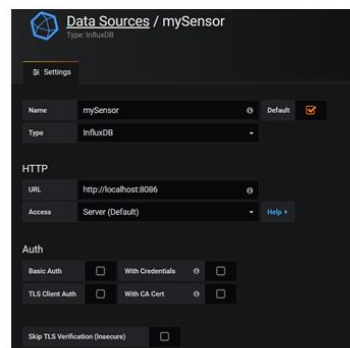
InfluxDB – Node Red

- How to store data in your database?
- Add an influxdb storage and connect it to your uplink
- Define a server, just add the Database name: mySensor
- Add
- In measurement field, add a name for your device: device1
- Go to InfluxDB shell
- Run : SHOW SERIES ON mySensor



Grafana

- Go to yours unzip Grafana directory/bin
- Start grafana-server.exe
- Go to : http://127.0.0.1:3000
- User name and password is: admin
- Provide a new password
- Click « Add data source »
- Add a name
- Choose InfluxDB type
- Define Database name « mySensor »
- Click on Save and Test



Grafana

- Create a new dashboard
- Click on Graph
- Panel Title / Edit
- Select your data source and measurement, field temperature, time 1s, fill linear
- Change to the last 5mn
- Put your finger on the sensor
- Look at your curve

