

# nanogateway lopy4 add node APB

cmonaton

August 2019

## 1 Introduction

But : Connecter un device à The Things Network par la méthode Activation By Personalisation.

Prérequis : savoir utiliser The Thing Network, cf tuto *nanogateway\_lopy4\_lorawan*

Carte : pycom lopy 4 avec expansion board V3.0



Figure 1: pycom lopy 4

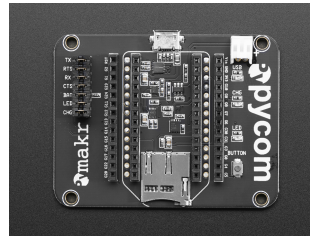


Figure 2: expansion board v3.0



Figure 3: antenne LoRa

## 2 Matériel

Branchez l'antenne LoRa avant d'alimenter la carte sinon la carte grille

## 3 Code pour Activation By Personalisation

Le code se trouve à : [https://github.com/GitClementtest/node\\_apb](https://github.com/GitClementtest/node_apb)

Informations complémentaires : <https://docs.pycom.io/tutorials/lora/lorawan-nano-gateway/>

### 3.1 Activation By Personalisation

*""" ABP Node example compatible with the LoPy Nano Gateway """*

```
from network import LoRa
import socket
import ubinascii
import struct
import time

# Initialise LoRa in LORAWAN mode.
lora = LoRa(mode=LoRa.LORAWAN, region=LoRa.EU868)

# create an ABP authentication params
dev_addr = struct.unpack(">1", ubinascii.unhexlify('2601147D'))[0] # these settings can
#be found from TTN
nwkey_swkey = ubinascii.unhexlify('3C74F4F40CAE2221303BC24284FCF3AF') # these settings can
#be found from TTN
app_swkey = ubinascii.unhexlify('OFFA7072CC6FF69A102A0F39BEB0880F') # these settings can
#be found from TTN

# join a network using ABP (Activation By Personalisation)
lora.join(activation=LoRa.ABP, auth=(dev_addr, nwkey_swkey, app_swkey))

# remove all the non-default channels
for i in range(3, 16):
    lora.remove_channel(i)

# set the 3 default channels to the same frequency
lora.add_channel(0, frequency=868100000, dr_min=0, dr_max=5)
lora.add_channel(1, frequency=868100000, dr_min=0, dr_max=5)
lora.add_channel(2, frequency=868100000, dr_min=0, dr_max=5)

# create a LoRa socket
s = socket.socket(socket.AF_LORA, socket.SOCK_RAW)

# set the LoRaWAN data rate
s.setsockopt(socket.SOL_LORA, socket.SO_DR, 5)

# make the socket non-blocking
s.setblocking(False)
```

```

""" Your own code can be written below! """

for i in range (200):
    s.send(b'PKT #' + bytes([i]))
    time.sleep(4)
    rx = s.recv(256)
    if rx:
        print(rx)
    time.sleep(6)

```

### 3.2 Créer une application sur The Thing Network

- Créez une application Depuis *Console* sélectionner application, puis ajoutez une application :



Donnez un nom pour l'application ID :

**ADD APPLICATION**

**Application ID**  
The unique identifier of your application on the network

test

**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

### 3.3 Ajouter un device dans l'application

Overview **Devices** Payload Formats Integrations Data Settings

**APPLICATION OVERVIEW**

Application ID testtsdlfkdm [documentation](#)

Description

Created 3 minutes ago

Handler ttn-handler-eu (current handler)

**APPLICATION EUIs** [manage euis](#)

Puis *register device*

Choisir un nom pour le device ID et 16 caractères hexadécimaux pour l'EUI.

REGISTER DEVICE
[bulk import devices](#)

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

device\_ex

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

55 55 55 55 55 55 55 55
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 02 10 FF

### 3.3.1 Choisir APB comme méthode d'activation

Dans *Device overview* allez dans *settings*

Overview
Data
**Settings**

**DEVICE OVERVIEW**

**Application ID** node\_nanogateway
**Device ID** test2
**Activation Method** ABP

**Device EUI**
<>
55 55 55 55 55 55 55 55

**Application EUI**
<>
70 B3 D5 7E D0 02 10 CD

Puis entrez la network session key et l'application session key présents dans le code.

Applications > node\_nanogateway > Devices > test2 > Settings

Location

**Description**  
A human-readable description of the device

**Device EUI**  
The serial number of your radio module, similar to a MAC address

55 55 55 55 55 55 55 55 8 bytes

**Application EUI**

70 B3 D5 7E D0 02 10 CD

**Activation Method**

OTAA ABP

**Device Address**

26 01 1A F4 4 bytes

**Network Session Key**

3C 74 F4 F4 0C AE 22 21 30 3B C2 42 84 FC F3 AF 16 bytes

**App Session Key**

0F FA 70 72 CC 6F F6 9A 10 2A 0F 39 BE B0 88 0F 16 bytes

Bouton *Save* puis revenez dans l'onglet *Overview* et ajouter le device address dans le code.

**DEVICE OVERVIEW**

Application ID `node_nanogateway`

Device ID `test2`

Activation Method `ABP`

Device EUI `<> 55 55 55 55 55 55 55 55`

Application EUI `<> 78 B3 D5 7E D8 02 10 CD`

**Device Address** `<> 26 01 1A F4`

Network Session Key `<> . . . . .`

App Session Key `<> . . . . .`

Status ● 5 seconds ago

Frames up 71 [reset frame counters](#)

Frames down 0

Ajoutez Device address, network Session Key et App Session Key dans `main.py`  
Télécharger le code sur la carte.

### 3.4 Lire la payload envoyée par le device

- Une fois le device connecté :

**DEVICES** + register device

< > 1 – 1/1

00000000	11 11 11 11 11 11 11 11	●
----------	-------------------------	---

- Dans l'onglet data :

Overview

Devices

Payload Formats

Integrations

Data

Settings

DEVICES

+

register device

<

>

1 – 1 / 1

XCV

44 44 44 44 44 44 44 44

APPLICATION DATA

Filters

uplink

downlink

activation

ack

error

	time	counter	port	
▲	16:22:34	11	2	dev id: 00000000 payload: 50 4B 54 20 23 0B
▲	16:22:24	10	2	dev id: 00000000 payload: 50 4B 54 20 23 0A
▲	16:22:14	9	2	dev id: 00000000 payload: 50 4B 54 20 23 09
▲	16:22:04	8	2	dev id: 00000000 payload: 50 4B 54 20 23 08
▲	16:21:54	7	2	dev id: 00000000 payload: 50 4B 54 20 23 07