PROJET DU LABO

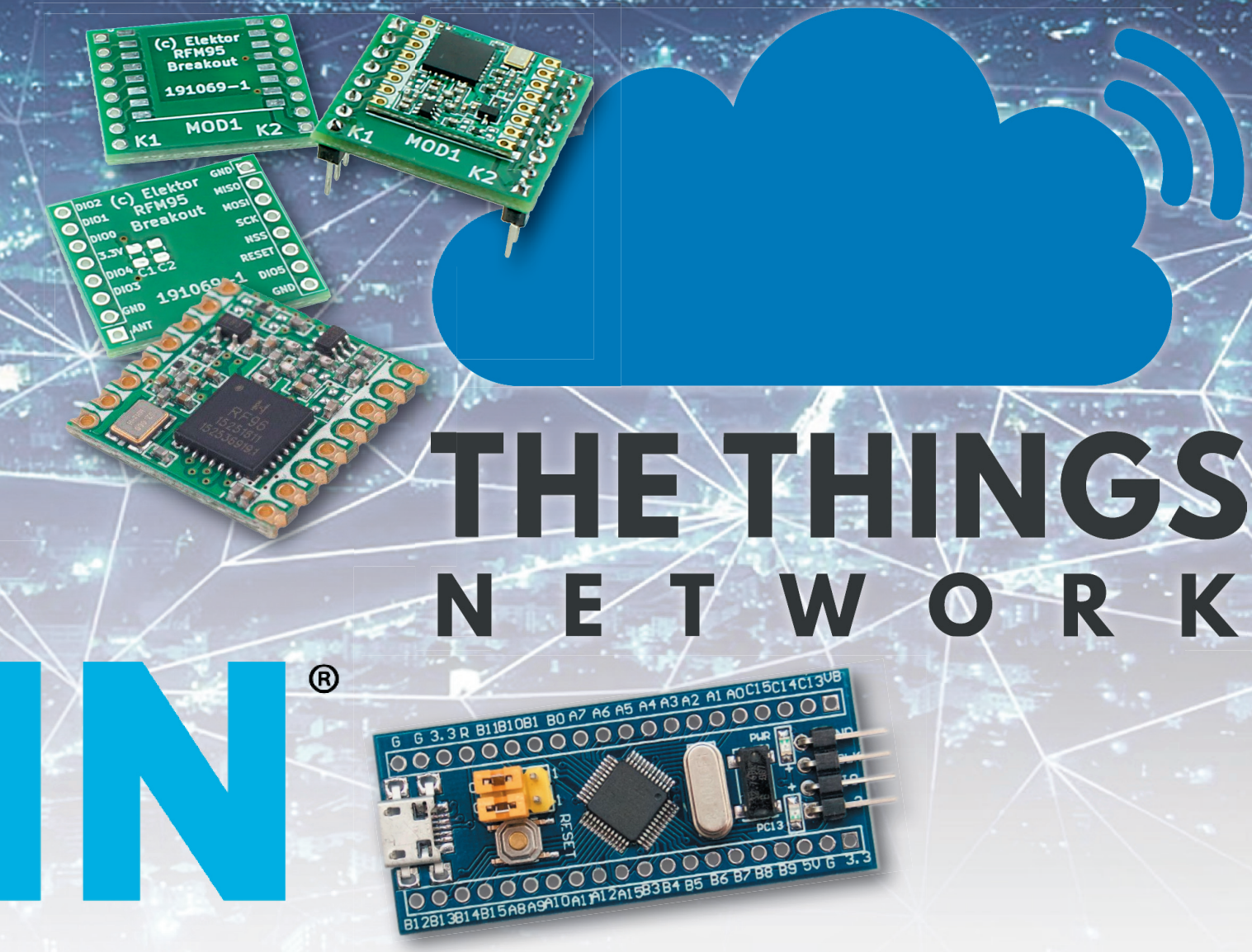
LoRaWAN : décollage facile

Avec Blue Pill, passerelle LoRa et The Things Network

Mathias Claußen (Elektor Labs)

La technologie radio LoRa, proposée par *Semtech*, est un mode de transmission de données à longue portée et à faible consommation. LoRa convient aux capteurs en réseau dont il faut ménager la source d'énergie. *The Things Network* est un réseau populaire et ouvert, qui peut recevoir des données émises par des capteurs et les rendre disponibles dans le monde entier.
Pour vos premières expériences, quelques euros et quelques cartes suffisent.

Comme son nom l'indique, un LoRaWAN utilise la technologie radio LoRa (=*low range*) pour faire circuler des données sur un réseau étendu (WAN = *wide-area network*). Il fournit l'infrastructure à distance pour les nœuds de capteurs en recueillant les données par des *passerelles* (stations de base) via LoRa et en les mettant à disposition sur l'internet. *The Things Network* est un LoRaWAN populaire en pleine expansion. Ce réseau communautaire gratuit jouit d'une bonne couverture. Pour utiliser activement ce réseau, il suffit d'un enregistrement et bien sûr le matériel adéquat. Il n'y a ni frais mensuels ni facturation par message ; il suffit de se conformer à la politique d'utilisation équitable du réseau. Cette *Fair Use Policy* vise à permettre à tous les participants de transmettre leurs données et à éviter la surcharge des passerelles (*gateways*).



THE THINGS NETWORK

IN®

Petits moyens

La solution présentée ici n'est qu'imparfaitement conforme aux spécifications de LoRaWAN et reste donc réservée à l'expérimentation. Les coûts sont de l'ordre de 23 € pour le nœud du capteur, plus éventuellement une plaque d'expérimentation et quelques fils de connexion. La liste des composants pour le capteur LoRa est assez courte :

- Carte contrôleur STM32 *Blue Pill* [**1**]
- Module radio LoRa RFM95 [**2**]
- Convertisseur USB-série [**3**]
- Cavaliers
- Plaque d'expérimentation

Pour commencer, nous utilisons une carte *Blue Pill* arduinoïde avec un puissant contrôleur STM32-ARM Cortex, disponible pour quelques euros, débogueur compris. Elle n'offre pas moins 64 Ko de Flash, 16 Ko de RAM et un port USB. Ce dernier est à utiliser avec circonspection, car cette carte est disons fortement optimisée en termes de coûts (pour le dire avec des fleurs), de sorte que les problèmes avec le port USB sont malheureusement

INFOS SUR LE PROJET

LoRa · LoRaWAN · BoB · The Things Network

débutant
➡ connaisseur
expert

±30 min

PC,
plaque d'essais et fils

±30 €

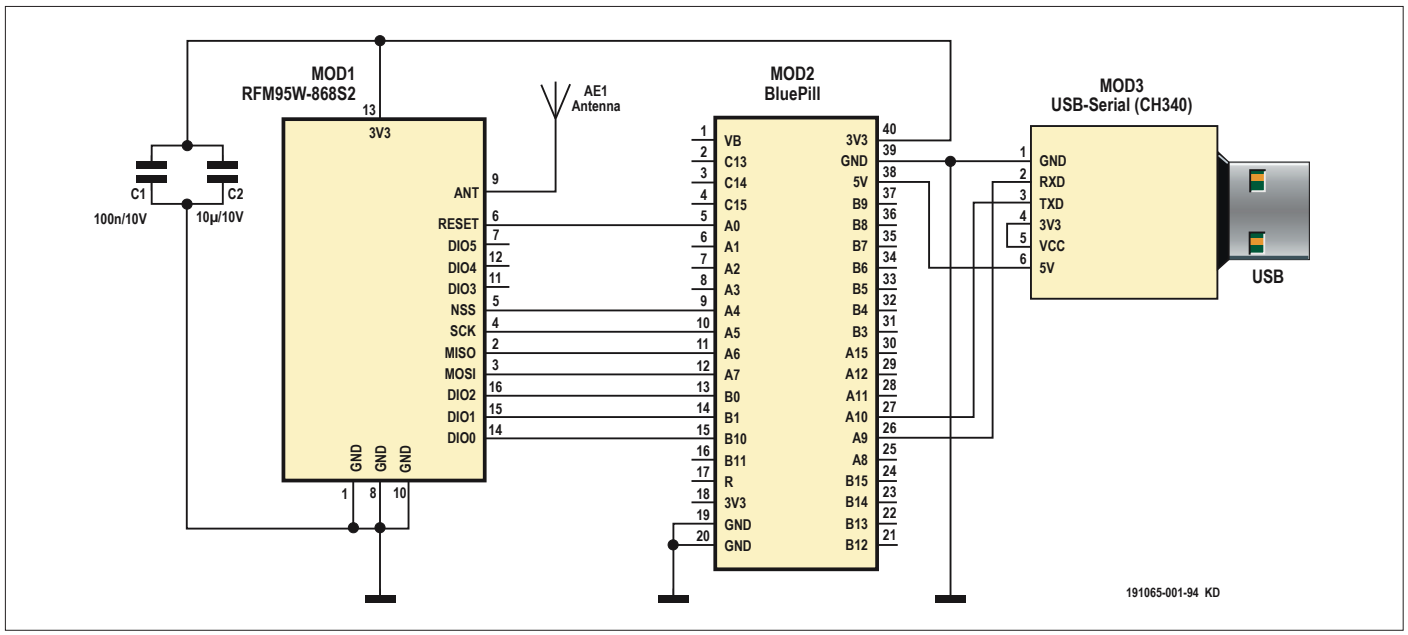


Figure 1 : Schéma du circuit du nœud LoRa à faible coût.

la règle plus que l'exception. Comme prise d'alimentation, pas de problème. Pour programmer la puce, on utilise le *bootloader* intégré. Le nouveau logiciel peut être programmé de manière fiable via l'interface série UART de la carte.

Les quelques composants, le module LoRa, la carte *Blue Pill* et un convertisseur USB-série sont connectés comme indiqué dans le schéma (**fig. 1**). Comme le module LoRa avec son pas de 2,00 mm n'est pas compatible avec les plaques d'expérimentation, Elektor Labs propose une petite carte d'interconnexion (**fig. 2**). Grâce à elle, le module pourra être enfiché facilement sur une plaque d'expérimentation. Il est pratique également de pouvoir monter les deux condensateurs C1 et C2 en même temps.

Ce qui manque encore, c'est une antenne. Un morceau de fil de cuivre de 1 mm de diamètre fait l'affaire. Pour le calcul de la longueur requise d'une antenne λ/4 pour la gamme de 868 MHz dans laquelle fonctionne le module LoRa : $\lambda/4 = (c_0/868\ \text{MHz})/4 = (299792458\ \text{m/s})/(8680000*4/\text{s}) = 0{,}08635\ \text{m} = 8{,}635\ \text{cm}$

Cette formule est valable dans le vide, or la propagation du signal dans le cuivre est plus lente que dans le vide. On applique donc un coefficient de raccourcissement de 0,95, de sorte que la longueur d'antenne λ/4 sera d'environ 82 mm.

On voit sur la **fig. 3** qu'il ne faut pas grand-chose pour construire notre capteur LoRa. Aucun capteur externe ne sera d'ailleurs utilisé lors des premiers tests ; car ce qui nous intéresse, c'est comment un nœud de capteurs (aussi appelé nœud) est configuré et mis en service.

Bien préparé...

Quelques logiciels sont nécessaires : l'IDE Arduino, la bibliothèque *MCCI LoRaWAN LMIC* et le paquet de support

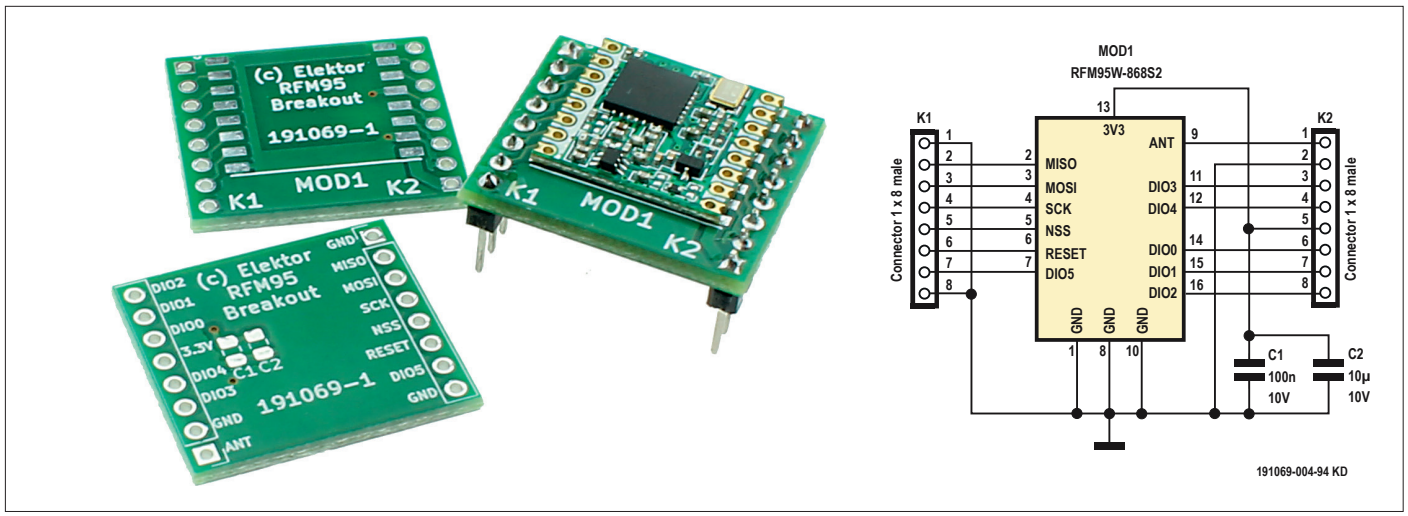
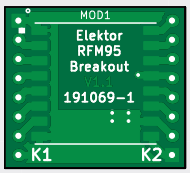


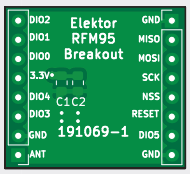
Figure 2 : Carte d'interconnexion pour le module LoRa.

LISTE DES COMPOSANTS

Condensateurs (0805) :
C1 = 100 n (10%, X7R, 10 V)
C2 = 10 µ (20%, X5R, 10 V)

Aussi :
K1,K2 = connecteur mâle 8x1, RM 2,54 mm
MOD1 = module émetteur-récepteur LoRa RFM95W (cf. encadré)
circuit imprimé 191069-1 (cf. encadré)





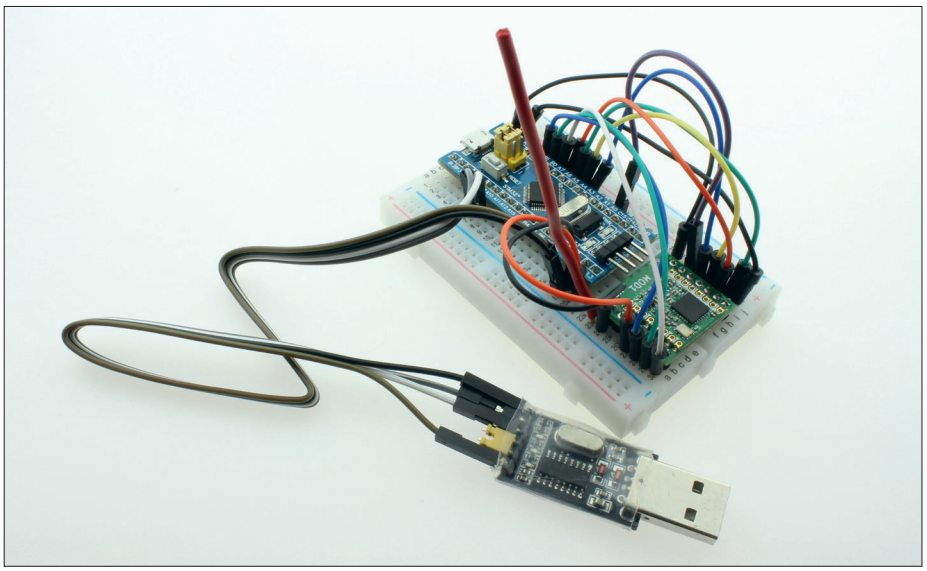


Figure 3 : Le matériel sur la plaque d'essais avec le module LoRa câblé.

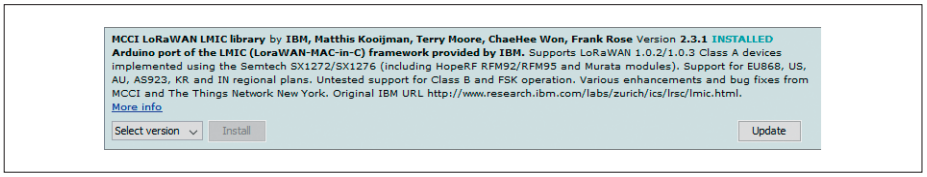


Figure 4 : Installation de la bibliothèque du LMIC.

Arduino-Core-STM32 Board.
La bibliothèque du LMIC peut être installée comme d'habitude à partir des bibliothèques (**fig. 4**). Dans le fichier de configuration du dossier de la bibliothèque du LMIC (sous Windows généralement parmi les documents de l'utilisateur) il faut indiquer dans quelle partie du monde vous vous trouvez et si c'est un module LoRa SX1276 (ce qui est le cas du RFM95). Pour l'Europe, dans le fichier \project_config\lmic_project_config.*h* on aura :

```
// project-specific definitions
#define CFG_eu868 1
//#define CFG_us915 1
//#define CFG_au921 1
//#define CFG_as923 1
//#define LMIC_COUNTRY_CODE LMIC_
   COUNTRY_CODE_JP /* for as923-
   JP */
//#define CFG_in866 1
#define CFG_sx1276_radio 1
//#define LMIC_USE_INTERRUPTS
```

Pour que l'EDI Arduino supporte la carte *Blue Pill*, une URL [**4**] doit être ajoutée dans les *Préférences Fichiers,* sous *URL supplémentaires des administrateurs de carte.* Ensuite, nous pouvons rechercher «STM32» dans le gestionnaire de cartes et installer des cœurs *STM32.* À l'issue de quoi toutes les bibliothèques requises sont incluses. La dernière étape consiste à sélectionner la carte (**fig. 5**).
Avant d'entrer dans le premier code, la différence entre OTAA (*over the air activation = par voie hertzienne*) et APB (*activation by personalization = par personnalisation*) doit être mentionnée brièvement. Avec OTAA, le nœud LoRa rejoint activement le réseau. Il reçoit une adresse de périphérique du réseau et échange des clés avec le réseau.
Pour *The Things Network*, c'est la méthode préférée pour participer au trafic sur le réseau.
En mode APB, l'adresse de l'appareil et les touches de l'appareil sont inscrites de manière permanente dans le code. Ça facilite la mise en route du nœud, ça en affaiblit aussi la sécurité. Pour plus de détails, consultez la documentation [**5**] de *The Things Network*.

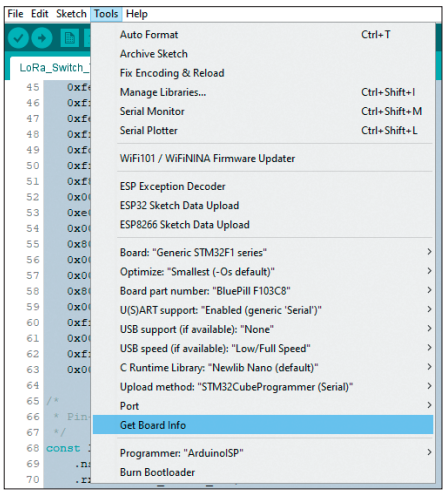


Figure 5 : L'information sur la carte.

Le programme d'exemple *ttn-abp* de la bibliothèque LMIC est utilisé comme point de départ. Le code envoie un *Hello, world!* toutes les 60 secondes et nécessite quelques ajustements concernant le matériel. À partir de la ligne 86 du croquis Arduino, l'attribution des broches est modifiée ainsi :

```
/*
 * Pin-Mapping for the RFM95 LoRa Module
 */
const lmic_pinmap lmic_pins = {
    .nss = PA4,
    .rxtx = LMIC_UNUSED_PIN,
    .rst = PA0,
    .dio = ,
};
```

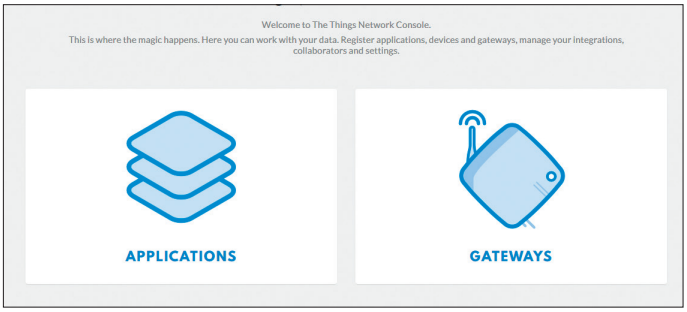


Figure 6 : Bienvenue dans le terminal de *The Things Network*.

Figure 7 : *Add Application!*

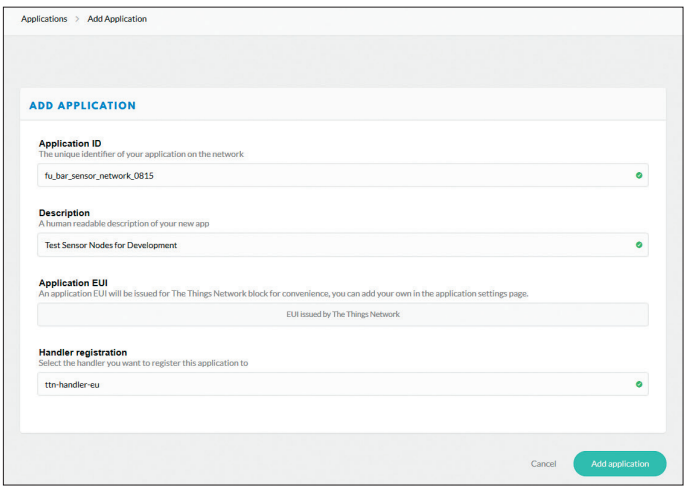


Figure 8 : Enregistrement d'une application.

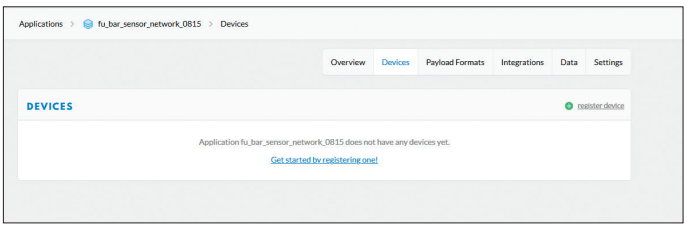


Figure 9 : Demande d'enregistrement de nœuds dans l'application.

Inscription dans *The Things Network*

En plus des broches, il faut configurer les données de réseau pour le nœud du capteur. Pour les obtenir, un nouveau nœud est d'abord créé sur *The Things Network*. Dans un premier temps, vous créez un compte d'utilisateur [**6**], dans lequel les nœuds sont créés et gérés. Pour qu'un utilisateur puisse garder la trace de ses nœuds, ceux-ci peuvent être regroupés sous *Applications* selon leur fonction. Après la connexion, les *applications* et les *passerelles* peuvent être gérées via le terminal (**fig. 6**).

S'il n'y a pas à proximité de passerelle utilisable, vous pouvez exploiter votre propre passerelle et la mettre à la disposition d'autres utilisateurs à proximité. La procédure sera expliquée plus tard.

Pour produire des données adéquates au nœud, une application doit d'abord être créée. Cliquez sur l'icône *APPLICATIONS* puis

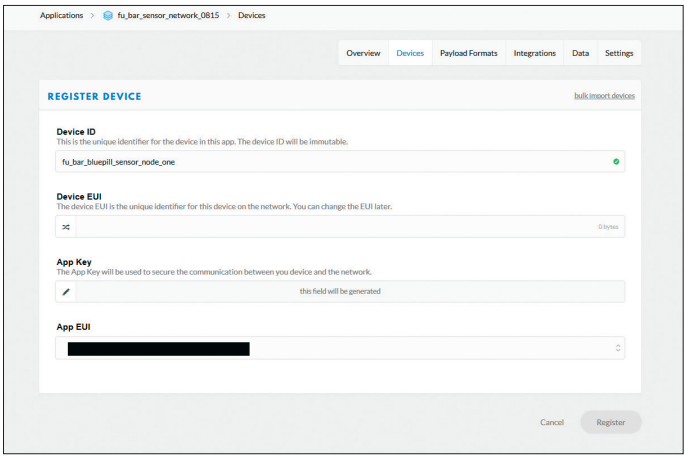


Figure 10 : Ici les données du nouveau nœud sont spécifiées.

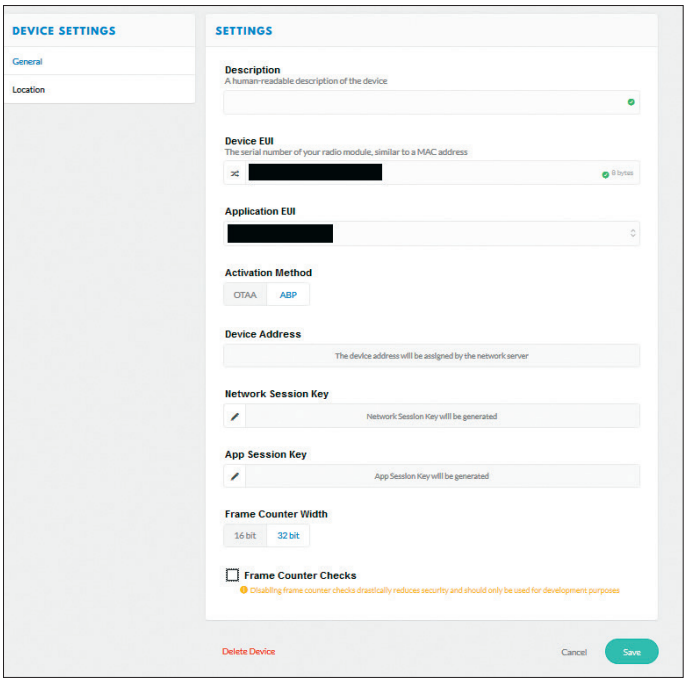


Figure 11 : Changer la méthode d'activation et désactiver (temporairement) les contrôles du compteur de trames.

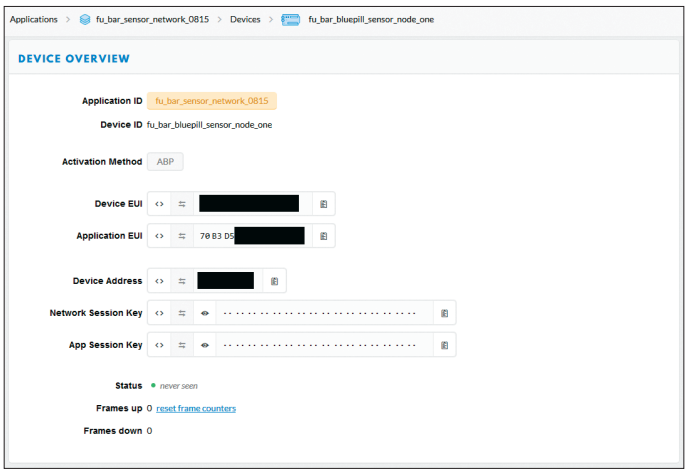


Figure 12 : Dans l'aperçu des appareils, il reste à régler correctement le format de données (LSB/MSB).

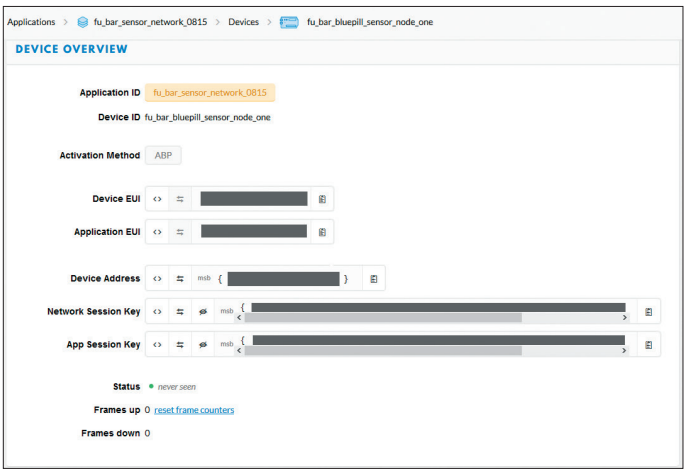


Figure 13 : L'appareil est configuré.

sur *Add application* (**fig. 7**). Remplir la fenêtre (**fig. 8**), puis le processus se termine par *Add application.* Dans l'application que l'on vient de créer, aucun nœud n'est encore enregistré. Cliquez sur *register devices* dans la fenêtre qui s'ouvre (**fig. 9**). Puis, dans la boîte de dialogue (**fig. 10**), saisissez le nom du nœud et cliquez sur le symbole de flèche sous *Device EUI à* gauche pour qu' apparaisse le texte *this field will be generated*. En appuyant sur *Register,* vous envoyez ces spécifications à *The Things Network*, qui enregistre alors le nouveau nœud et fournit les données d'accès. Sous *Settings* (**fig. 11**), la méthode d'activation passe de *OTAA* à *ABP* et pour les premières expériences, la case *Frame Counter Checks* est décochée. Dans la vraie vie, ce réglage est risqué ! Une fois les premiers pas réussis, il faut donc, à la première occasion, remettre cette coche d'urgence ! Après l'enregistrement des réglages, de retour sur la *carte de l'appareil,* un clic sur l'*onglet Overview* permet de visualiser d'un coup d'œil toutes les données nécessaires pour le nœud (**fig. 12**). Le *Device EUI, qui* correspond à peu près à une adresse MAC [**5**], ne nous intéresse pas ici, pas plus que l'*Application EUI* attribuée par TTN, nécessaire pour la procédure OTAA. Seul le contenu des trois champs suivants est pertinent : *Device Address*, *Network Session Key* et *App Session Key*. Les formats de données sont une pierre d'achoppement : il faut indiquer correctement si les zones LSB ou MSB doivent être transférées. Le croquis nécessite, en tant que LSB, l'adresse du périphérique et la clé de session réseau et, en tant que tableau MSB, la clé de session de l'application. Les réglages peuvent être basculés en cliquant sur «< >» ; les réglages apparaissent à gauche des fenêtres (**fig. 13**).
Dans le sketch, la clé de session réseau est ensuite saisie dans NWKSKEY, la clé de session applicative dans APPSKEY et enfin l'adresse du périphérique dans DEVADDR.

```
// LoRaWAN NwkSKey, network session key
static const PROGMEM u1_t NWKSKEY[16] = { Network
   Session Key };

// LoRaWAN AppSKey, application session key
static const u1_t PROGMEM APPSKEY[16] = { App Session
   Key };
```

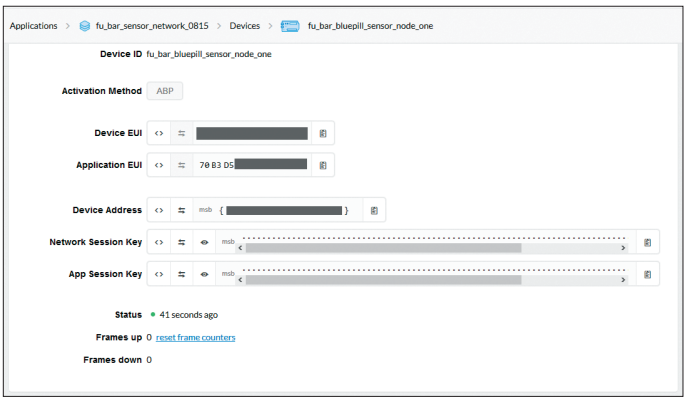


Figure 14 : Le noeud transmet !

```
// LoRaWAN end-device address (DevAddr)
// See http://thethingsnetwork.org/wiki/AddressSpace
// The library converts the address to network byte
   order as needed.
static const u4_t DEVADDR = 0xDEVICE_ADDRESS;
// Change this address for every node!
```

Si une passerelle LoRa se trouve à proximité, le sketch peut être compilé et téléchargé tel quel. Dans le terminal TTN, vous voyez immédiatement que le nœud vient d'envoyer des données (**fig. 14**). Un clic sur *Data* affiche les données de l'utilisateur du nœud.

Mise en place de votre propre passerelle

Si aucune passerelle n'est à portée, vous pouvez en créer une. Il existe différentes possibilités, allant de la solution commerciale toute faite à la passerelle monocanal (non conforme à LoRaWAN) composée d'un RPi avec LoRaWAN-HAT. Une passerelle entièrement compatible est la passerelle LoRaWAN Indoor LPS8 de Dragino [**7**], disponible dans la boutique en ligne d'Elektor pour environ 120 € (voir encadré). Elle peut recevoir simultanément sur toutes les bandes de fréquences dans la gamme de 868 MHz. La variante Dragino LG02 avec deux modules LoRa est un peu limitée en réception, mais

Figure 15 : Un Raspberry Pi avec Dragino-LoRa-HAT comme passerelle.

aussi meilleur marché (80 €). Les passerelles sont basées sur la distribution Linux OpenWRT, spécialement conçue pour les routeurs, avec quelques ajustements pour fonctionner comme une passerelle LoRa.

Si vous possédez un Raspberry Pi, vous pouvez l'équiper d'un module LoRa et l'utiliser comme passerelle avec le logiciel [**8**] qui supporte le Dragino-LoRa-HAT [**9**]. Il est encore plus économique de connecter le module RFM95 directement au RPi avec quelques fils. Avec la carte d'interconnexion, c'est très facile. Pour le câblage et l'installation, vous suivez les indications [**8**] et la définition des broches pour le Dragino-LoRa-HAT.

Avec cette passerelle RPi, un nœud ne peut cependant être exploité qu'en mode ABP ; il n'est pas possible de transporter des messages du LoRaWAN vers le nœud. Le canal et le facteur d'étalement doivent également être réglés de façon permanente dans le nœud. Si vous voulez démarrer plus rapidement et avec moins d'embûches, vous devriez investir les quelques euros et utiliser la passerelle LG02 de Dragino, qui permet également le mode OTAA et l'envoi de données aux nœuds depuis le LoRaWAN. Pour plus de commodité (et moins de mauvaises expériences avec le RFM95 et les cavaliers), nous avons fait une razzia dans le stock d'Elektor et utilisé un Dragino-LoRa-HAT sur un Raspberry Pi 3B+ avec une image Raspbian à jour, fraîchement installée pour le test (**fig. 15**). Tout cela est rapidement mis en place, la carte SD est insérée, on branche moniteur, souris, clavier et câble Ethernet pour l'accès à l'internet, et c'est parti. Le système démarre directement sur le bureau, mais il faut d'abord ouvrir le terminal et régler certains paramètres matériels.

Avec `sudo raspi-config` vous accédez au menu de configuration du RPi. Sous *Interface Options,* sélectionnez *P4 SPI* et activez cette interface avec un *Yes*. Après le redémarrage, l'interface SPI est active, ce qui permet de poursuivre la configuration.

Encore une fois, un terminal est ouvert pour télécharger certains paquets et le code source du *Packet-Forwarder*. Le code source est fourni par GitHub, donc la commande `git clone https://github.com/hallard/single_chan_pkt_fwd.git` téléchargera la dernière version. Des bibliothèques supplémentaires doivent être installées. La commande `sudo apt-get install wiringpi` installe la bibliothèque *WiringPi* dont nous avons besoin pour la compilation. Avec `cd single_chan_pkt_fwd` nous passons dans le répertoire du code source téléchargé, compilons le code source avec `make` d'abord dans un programme et l'installons avec `make install` ensuite.

La passerelle est maintenant configurée, mais la configuration des broches pour la LoRa-HAT dans le fichier *global_conf.json* reste à faire. La configuration suivante peut être utilisée pour le HAT de Dragino :

```
{
«SX127x_conf»:
  {
    «freq»: 868100000,
    «spread_factor»: 7,
    «pin_nss»: 6,
    «pin_dio0»: 7,
    «pin_rst»: 0
  },
  «gateway_conf»:
  {
    «ref_latitude»: 0.0,
    «ref_longitude»: 0.0,
    «ref_altitude»: 10,

    «name»: «Enter your Gatewayname here»,
    «email»: «contact@whatever.com»,
    «desc»: «Dragino Single Channel Gateway on RPI»,

    «servers»:
    [
     {
       «address»: «router.eu.staging.thethings.
   network»,
       «port»: 1700,
       «enabled»: true
     },
     {
       «address»: «router.eu.thethings.network»,
       «port»: 1700,
       «enabled»: false
}]}}
```

Avec ces réglages, la passerelle écoute sur 868,1 MHz avec un facteur d'étalement SF7. Pour pouvoir envoyer les données au TTN, il reste quelques réglages mineurs à effectuer. Il faut d'abord redémarrer la passerelle pour activer les modifications qui viennent d'être effectuées. Pour ce faire, saisissez `systemctl stop single_chan_pkt_fwd` dans le terminal, puis `systemctl start single_chan_pkt_fwd`. La passerelle est maintenant configurée comme un service ou peut être exécutée directement sous l'utilisateur root. Avec le `statut systemctl single_chan_pkt_fwd` vous interrogez le statut du service (**fig. 16**).

La passerelle est maintenant prête à fonctionner. Il suffit de la configurer dans TTN pour que les paquets puissent également être attribués et que la passerelle puisse être administrée. Cliquez sur l'icône *GATEWAYS* (**fig. 17**) puis sur *register gateway* (**fig. 18)** dans le terminal de *The Things Network*. Dans l'onglet *Passerelle de registre* (**fig. 19**), la coche doit être réglée sur «*J'utilise le réacheminement de paquets Semtech*».

Pour le *Gateway EUI,* l'adresse MAC du Raspberry Pi (déterminée avec `cat /sys/class/net/eth0/address`) est saisie et remplie avec *FF FF au* milieu. Par exemple, si `b8:27:eb:12:34:56` est l'adresse MAC, entrez *B8 27 EB* **FF FF** *12 34 56* dans le champ *Gateway EUI* du terminal de *The Things Network*. Cliquez sur l'*onglet Gateway,* la passerelle est maintenant prête à l'emploi. Pour que le nœud LoRa et la passerelle fonctionnent ensemble, il faut définir une fréquence de transmission dans le code source et spécifier le facteur d'étalement. De plus, le nœud est limité à la fréquence définie dans la passerelle. L'extrait du code source montre les ajustements pour le nœud LoRa. Un exemple complet peut être téléchargé à partir de la page Elektor Labs [**10**].

```
…
// 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);

//If we have a single channel one module gateway we
   need to add these lines
// Define the single channel and data rate (SF) to
   use
int channel = 0;
int dr = DR_SF7;

// Disable all channels, except for the one defined
   above.
// FOR TESTING ONLY!
for(int i=0; i<9; i++) { // For EU; for US use i<71
  if(i != channel) {
    LMIC_disableChannel(i);
  }
}
```

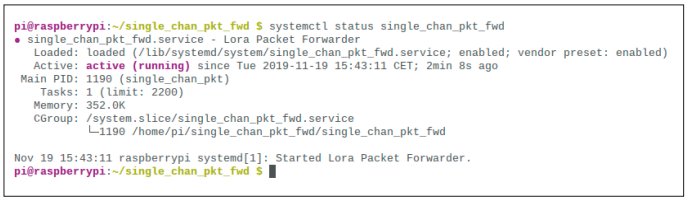


Figure 16. Copie d'écran du terminal : Service OK !

Figure 17 : Dans le TTN, on fait les réglages de la passerelle.

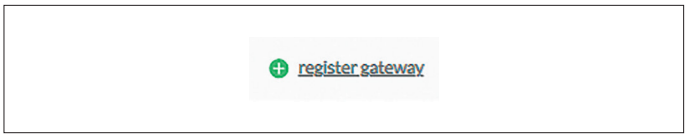


Figure 18 : Enregistrement de la passerelle dans le TTN.

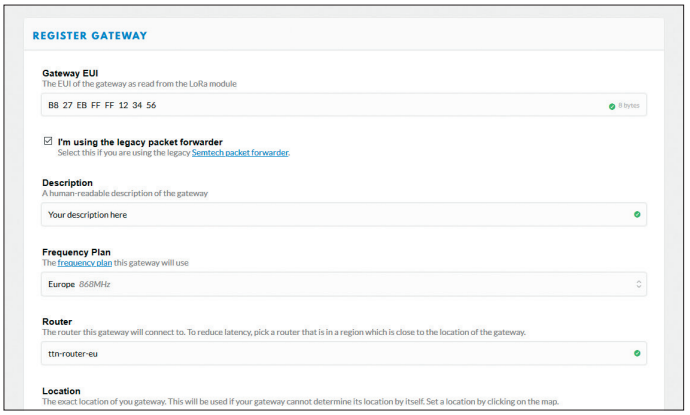


Figure 19 : Les données de la passerelle sont saisies ici.

```
// Set data rate (SF) and transmit power for uplink
LMIC_setDrTxpow(dr, 14);

// Start job
do_send(&sendjob);
…
```

Dans cet exemple, le nœud est réglé sur 868,1 MHz et SF 7. Nous utilisons également ces valeurs dans notre passerelle. La passerelle et le nœud ont maintenant leur logiciel pour que le test puisse démarrer. Après un court laps de temps, de nouveaux messages de notre nœud devraient arriver dans le terminal de TTN (**fig. 20**).

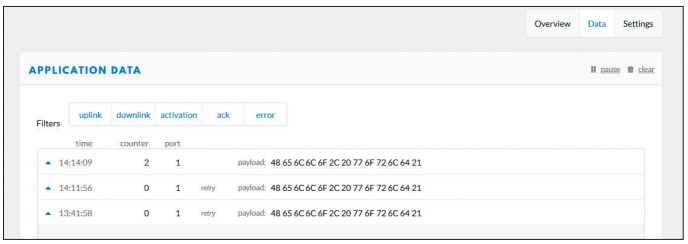


Figure 20 : Youpie, les données du nœud arrivent !

Conclusion et perspectives

Ainsi, pour faire ses premiers pas avec LoRaWAN, un nœud pour ±25 € et une passerelle basée sur un Raspberry Pi et un module RFFM95 suffisent. Cette solution reste très limitée et ne convient pas à une exploitation productive. Si après l'avoir essayé vous souhaitez faire plus avec LoRaWAN, vous devriez échanger la passerelle contre un appareil commercial entièrement compatible LoRaWAN, disponible pour moins de 120 € [**7**]. Avec une telle passerelle, il est possible d'utiliser tout le potentiel du LoRaWAN, y compris les messages de la passerelle vers le nœud.

Un point que nous n'avons pas encore examiné est la collecte et le traitement des données de *The Things Network*. Ce sera pour un autre article dans un prochain numéro.

191065-02

Liens

[1] Carte Blue Pill : www.amazon.de/s?k=bluepill&__mk_de_DE=%C3%85M%C3%85%C5%BD%C3%95%C3%91&ref=nb_sb_noss_1

[2] Module LoRa : www.elektor.de/rfm95-ultra-lora-transceiver-module-868-915-mhz

[3] Convertisseur USB-série CH340 (3,3 V) : www.elektor.de/19151

[4] Support de la carte : https://github.com/stm32duino/BoardManagerFiles/raw/master/STM32/package_stm_index.json

[5] LoRaWAN : adressage et activation : www.thethingsnetwork.org/docs/lorawan/addressing.html

[6] Créer un compte TTN : https://account.thethingsnetwork.org/register

[7] Passerelle LPS8 : www.dragino.com/products/lora-lorawan-gateway/item/148-lps8.html

[8] Logiciel HAT Gateway : https://github.com/hallard/single_chan_pkt_fwd

[9] HAT Gateway : www.elektor.de/dragino-lora-gps-hat-for-raspberry-pi-868-mhz

[10] La page de ce projet : www.elektormagazine.de/191065-01