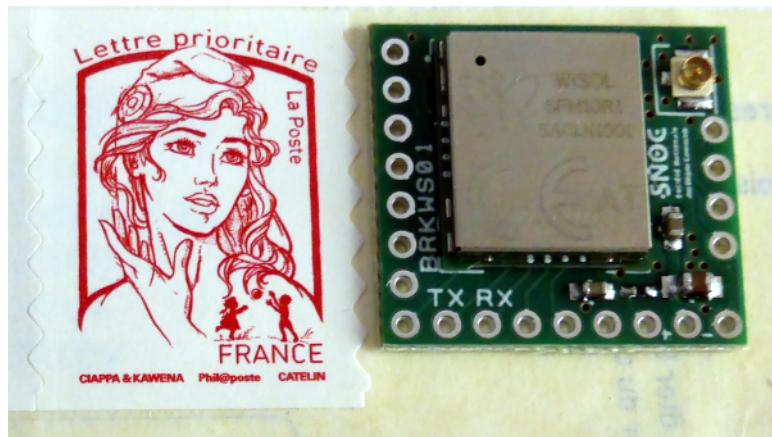




Framboise 314, le Raspberry Pi à la sauce française....

La référence du Raspberry Pi en France – Par l'auteur du livre "Raspberry Pi 3 et Pi Zero" paru aux Edts. ENI
Vous êtes ici : [Accueil](#) » **Carte de prototypage SIGFOX par SNOC**



Publié le 15 février 2017 - par [François MOCQ](#)

Carte de prototypage SIGFOX par SNOC

la [SNOC](#) sort une carte breakout (prototypage) de dimensions réduites. Basée sur un module [Wisol SFM10R1](#) cette carte vous permet de disposer des fonctionnalités de SigFox sur tous vos montages. C'est l'IoT à portée de Raspberry Pi !

Après la carte RPISIGFOX destinée à munir le Raspberry Pi de l'accès à [SigFox](#) que je vous avais présentée fin 2015, voici une nouvelle carte équipée du tout récent module SFM10R1 de Wisol. SNOC m'a proposé de la tester sur le Raspberry Pi, ce que j'ai accepté avec plaisir et j'ai donc reçu à titre gracieux ce module (*merci Pascal*), ce qui vaut à cet article d'être classé comme sponsorisé.

Si vous voulez un rappel sur la technologie SigFox, reportez vous à l'article précédent. Rapidement, on peut dire que la portée d'un émetteur en numérique dépend de sa puissance, mais aussi du débit de données. Avec un faible débit de données, la portée peut être très importante. C'est le principe de SigFox.

Confidentialité

===== Article sponsorisé =====

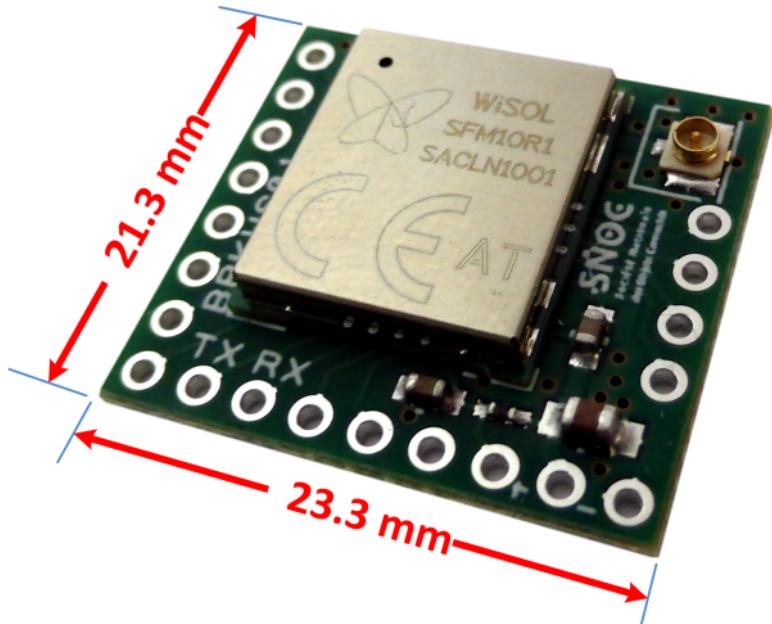
Au sommaire : [[cacher](#)]

- 1 Breakout board – Carte de Prototypage SigFox pour Raspberry Pi et Arduino
 - 1.1 La carte SNOC BRKSW01
 - 1.2 Le matériel reçu
 - 1.3 Le module WISOL WSSFM10R1
 - 1.4 Ses caractéristiques :
 - 1.5 Le schéma du AX-SFEU
 - 1.6 En résumé :
- 2 Montage de la carte SNOC BRKWS01
 - 2.1 Soudure du connecteur
 - 2.2 Connexion de l'antenne
 - 2.3 Connexion du câble sur la carte
 - 2.4 Mise en place de la carte prototype sur le Raspberry Pi
 - 2.5 Connexion et tests
 - 2.6 Configuration de putty
 - 2.7 L'écho, qu'est-ce ?
 - 2.7.1 Premier cas : pas d'écho
 - 2.7.2 Deuxième cas : écho distant
 - 2.7.3 Troisième cas : écho local
 - 2.7.4 Quatrième cas : écho local ET écho distant
 - 2.7.5 Mode écho local sur putty
- 3 Inscription sur le site web SigFox
 - 3.1 Accéder à la page d'inscription
- 4 Envoi de données via SigFox
 - 4.1 Mode manuel
 - 4.2 Programme Python
 - 4.2.1 Essai du programme SigFox en Python
- 5 Envoyer le message vers un site web
 - 5.1 Principe
 - 5.2 Paramétrage du callback
 - 5.3 Le fichier sigfox.php
 - 5.4 Les données enregistrées
- 6 Dans 12 octets qu'est ce que tu veux mettre ?
 - 6.1 Gestion des bits de données
 - 6.1.1 Une station météo isolée
 - 6.1.1.1 Température
 - 6.1.1.2 Pression
- 7 En bonus quelques commandes supplémentaires
- 8 Présentation vidéo de la SNOC
- 9 Conclusion
- 10 Sources

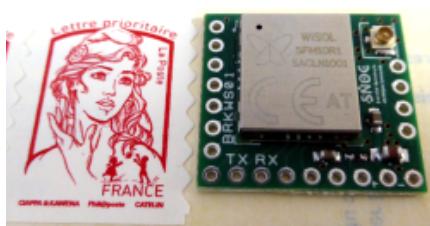
Breakout board – Carte de Prototypage SigFox pour Raspberry Pi et Arduino

 Confidentialité

La carte SNOC BRKSW01



La carte que propose SNOC est vraiment de dimension réduite puisqu'elle occupe moins de 5 cm² !



Si on la compare à un timbre poste on voit que son encombrement est à peine supérieur.
Ceci permet d'envisager de l'embarquer dans des montages de taille réduite qui

nécessiteraient une connexion à SigFox.

IoT nous voici 😊 !

Le matériel reçu



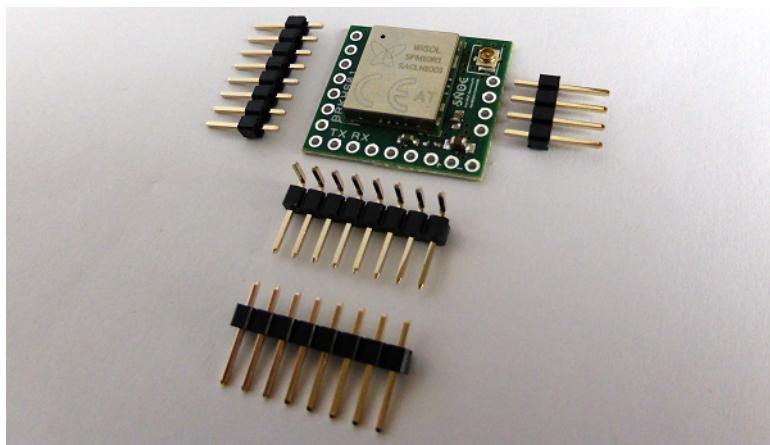
Confidentialité



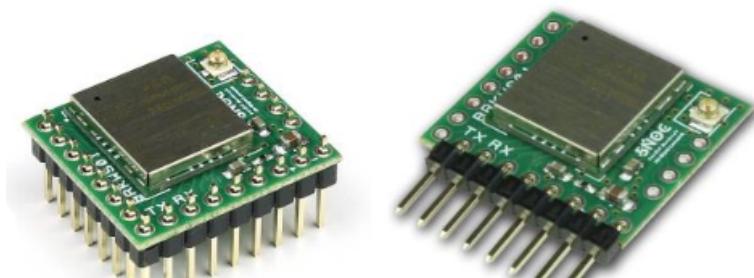
L'ensemble du matériel arrive dans un carton. Tout est bien protégé.
L'antenne et son câble sont dans un sachet plastique zippé.



La carte SigFox est dans une pochette antistatique de bonne qualité.
Vous éviterez de trop la manipuler, de mettre les doigts sur les pastilles
etc. Ça craint l'électricité statique ces petites choses là !

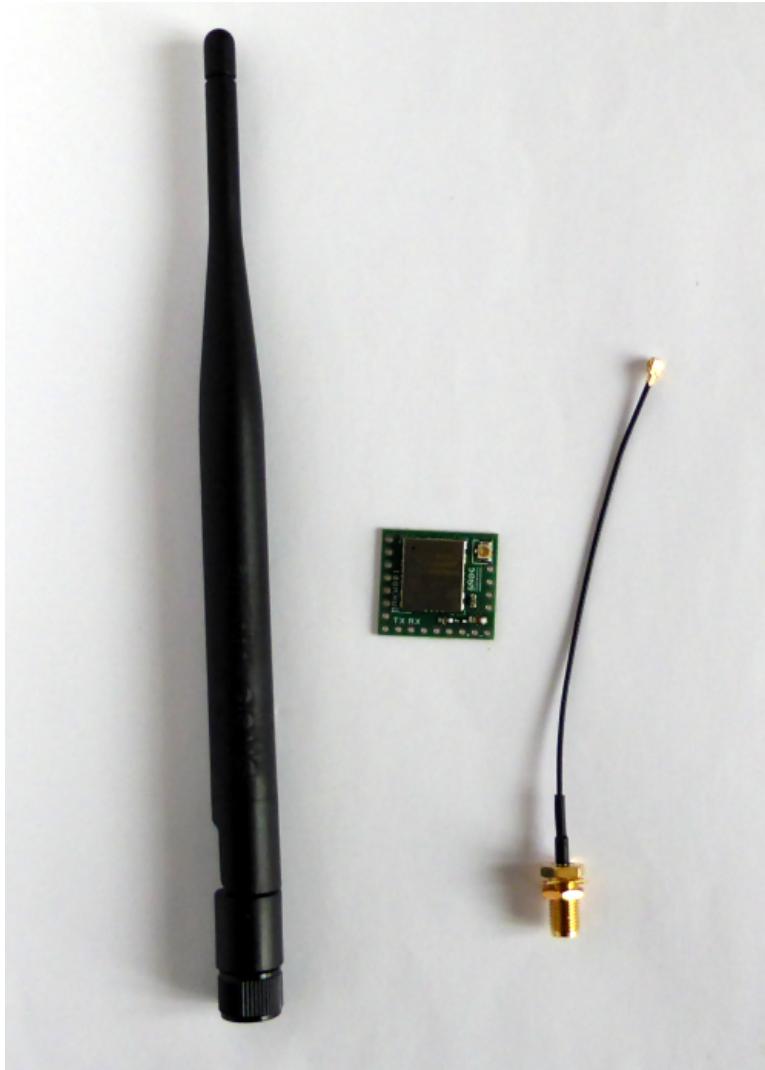


Le matériel inclus dans le paquet reçu intègre le module lui-même, ainsi
que les connecteurs à souder (*eh oui, il faudra sortir votre fer à souder !*)
pour monter le module à l'horizontale ou à la verticale.



Confidentialité

Le montage de gauche permet d'assurer une bonne rigidité en cas de montage sur une breadboard (*plaque d'expérimentation*) ou sur une carte de circuit imprimé. La version de droite, pour montage vertical autorise un montage/démontage facile sur une breadboard. C'est cette configuration que j'ai choisie.



Le matériel comprend également une antenne adaptée à la fréquence de SigFox, ainsi qu'un adaptateur **UFL <=> RP-SMA** de 100 mm de longueur, assurant la liaison entre la prise coaxiale présente sur la carte et l'antenne.

Le module WISOL WSSFM10R1

Le module émetteur/récepteur SigFox qui équipe la carte pèse moins d'un gramme (0,85 g) et ne mesure quant à lui que 13 x 15 x 2,21 mm. Il confère à la carte ses caractéristiques, la contribution de SNOC est de faciliter son utilisation par les makers en proposant une carte facile à mettre en œuvre. Le module est en CMS et pas facile à souder avec des

Confidentialité

moyens amateurs.



Alimenté entre 1,8 et 3,3 volts, il consomme un peu plus de 50 mA en émission et 15 mA en réception (uniquement pendant le temps de ces opérations). En veille, la consommation chute à 2 µA !

Ses caractéristiques :

- Taille : 13.0×15.0×2.21mm
- Chipset : [AX-SFEU-1-01](#) / ON Semiconductor
- Freq. Tx/ Data rate : 868.13MHz/ 100bps
- Freq. Rx/ Data rate : 869.525MHz/ 600bps
- Puissance sortie Tx : +14dBm (max)
- Sensibilité Rx : -127dBm@600bps
- Alimentation : @+3.3V
- Tx : 60mA(max), Rx: 15mA (max)
- Tensions acceptées : +1.8V~+3.6V
- Température d'utilisation : -30°C à +85°C

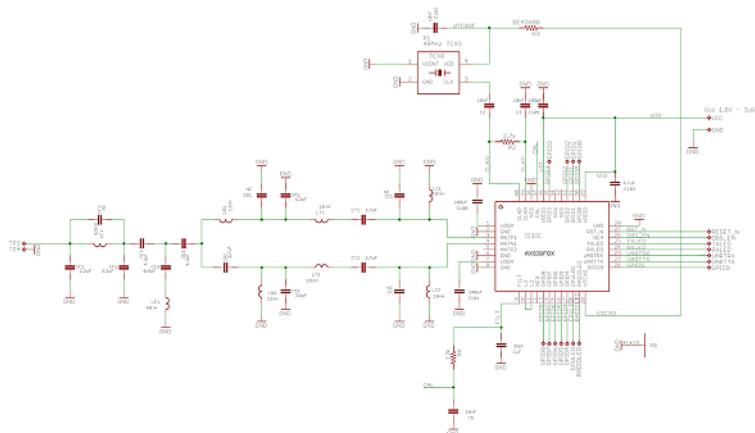
Le module utilise un [TCXO](#) 48 MHz pour rester dans la tolérance des fréquences SigFox durant toute la durée de vie du produit.

Ce module intègre un circuit ON Semiconductor AX-SFEU dont [la documentation est disponible](#) en ligne.

Le schéma du AX-SFEU

Confidentialité

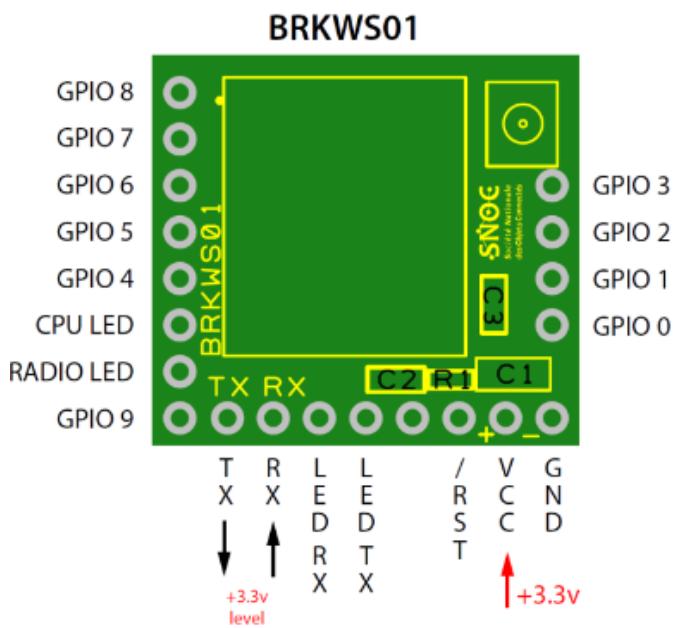
Typical Application Diagrams
Typical AX-SFEU / AX-SFEU-API Application Diagram



Cliquez pour agrandir.

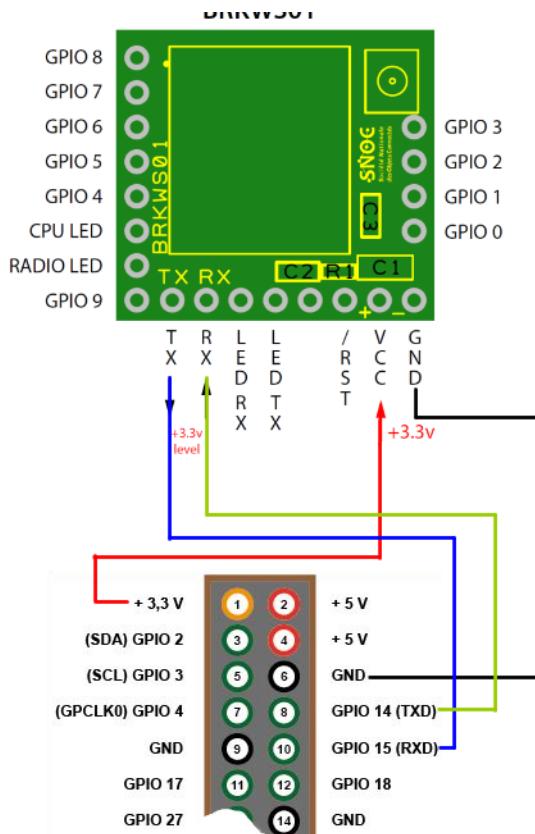
Ce schéma est extrait d'une [note d'application du module AX-SFEU](#) (page 16). S'il peut donner une idée du schéma réel du module Wisol dans lequel le AX-SFEU est utilisé, il n'y a aucune garantie qu'il soit exactement celui qui est mis en œuvre. **Il n'est donné qu'à titre indicatif.**

En résumé :



La carte BRKWS01 requiert au minimum les connexions +3,3v, masse et Tx Rx. Les transmissions de données peuvent être surveillées avec les LED LEDTX et LEDRX. La carte dispose également de deux sorties pour une LED CPU (activité du processeur) et RADIO LED pour l'émission. Il faut réfléchir à l'ajonction de ces LED qui peuvent se révéler utiles pour la mise au point mais seront par le suite consommatrices de courant...

Confidentialité



Voilà le schéma de branchement que j'ai retenu.

La carte SNOc embarque un module [Wisol SFM10R1](#)... qui embarque [un module AX-SFEU](#) 😊 Avec [la doc de ce dernier module](#) on peut donc tirer pas mal d'informations sur le fonctionnement de tout cet ensemble et envisager des trucs rigolos... non ?

Table 10. COMMANDS

| Command | Name | Description |
|------------|-------------|--|
| AT\$I=uint | Information | Display various product information: 0: Software Name & Version Example Response: AX-SFEU 1.0.6-ETSI 1: Contact Details Example Response: support@axsem.com 2: Silicon revision lower byte Example Response: 8F 3: Silicon revision upper byte Example Response: 00 4: Major Firmware Version Example Response: 1 5: Minor Firmware Version Example Response: 0 7: Firmware Variant (Frequency Band etc. (EU/US)) Example Response: ETSI 8: Firmware VCS Version Example Response: v1.0.2-36 9: SIGFOX Library Version Example Response: DL0-1.4 10: Device ID Example Response: 00012345 11: PAC Example Response: 0123456789ABCDEF |

Par exemple, d'après SNOc les commandes AT\$I=10 et AT\$I=11 renvoient l'ID du module et le PAC... Ce qui correspond bien à la documentation du module AX-SFEU.

⚠️ Attention !

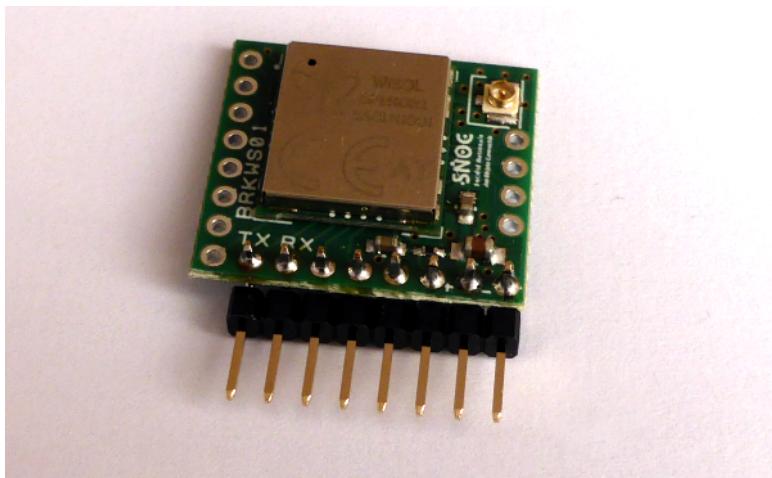
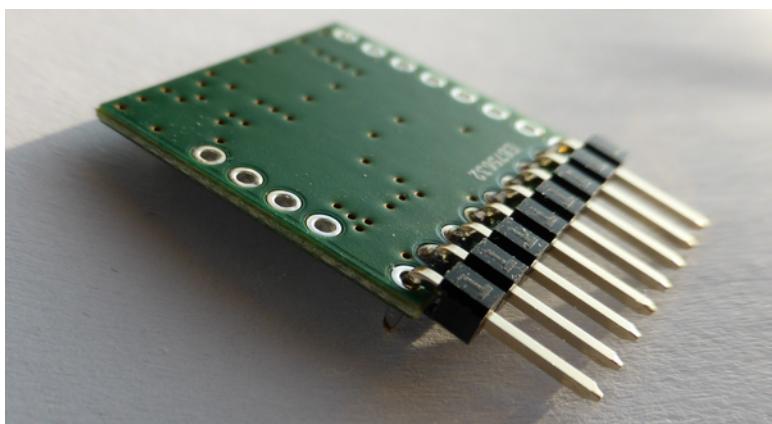
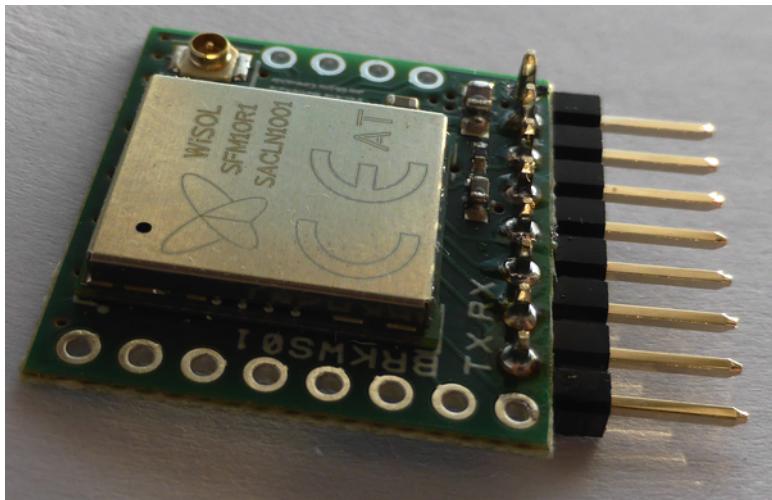
Certaines modifications peuvent mettre en danger le module Wisol embarqué sur la carte SNOc. Vous risquez de compromettre le bon fonctionnement de la carte en modifiant certaines

Confidentialité

valeurs. Dans ce cas vous perdez bien entendu la garantie SNOC.
Si vous intervenez dans les registres de la carte, vous le faites à vos risques et périls, ni SNOC ni framboise314 ne pourront être tenus pour responsables en cas de dysfonctionnement.

Montage de la carte SNOC BRKWS01

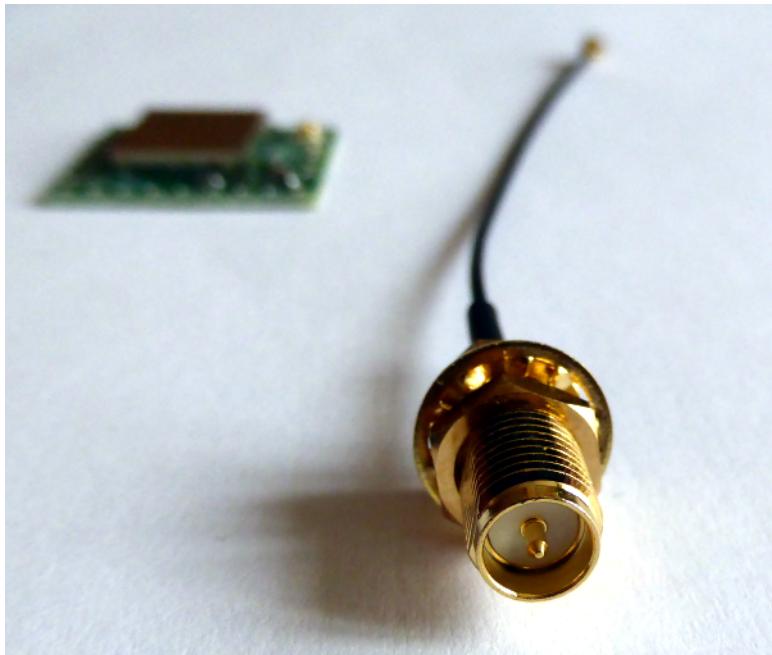
Soudure du connecteur



Confidentialité

J'ai sorti mon plus beau fer à souder pour souder le connecteur coudé. Utilisez un fer bien propre et évitez de chauffer trop fortement les pastilles. Éventuellement laissez un peu de temps entre deux soudures pour laisser redescendre la température.

Connexion de l'antenne

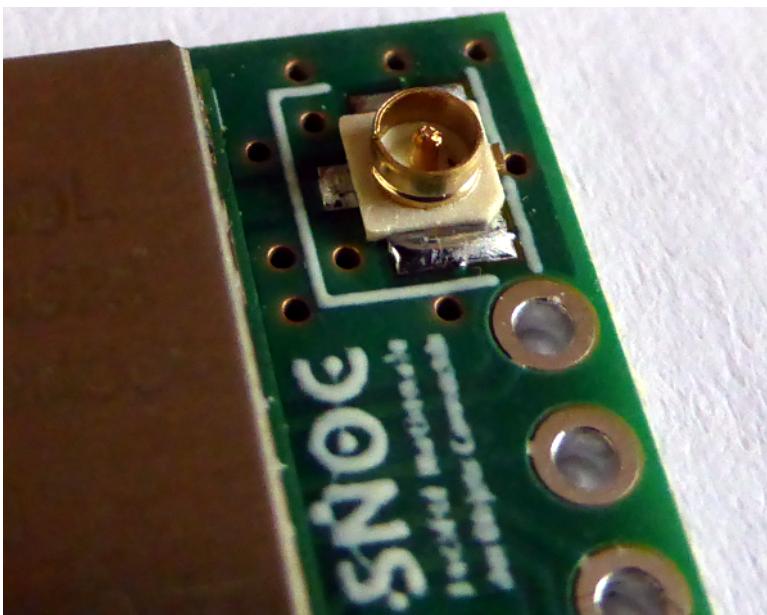


L'antenne se connecte sur cette prise, destinée à être montée sur un châssis ou un boîtier. Si vous faites un montage "en l'air" vissez d'abord la prise SMA sur l'antenne pour éviter de soumettre le câble à des torsions trop importantes.

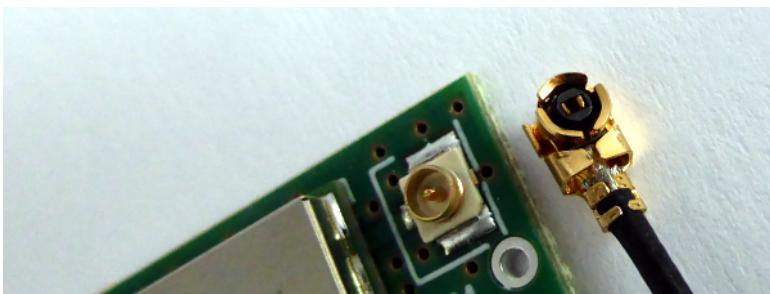


Vissez bien la prise SMA sur l'antenne.

Connexion du câble sur la carte



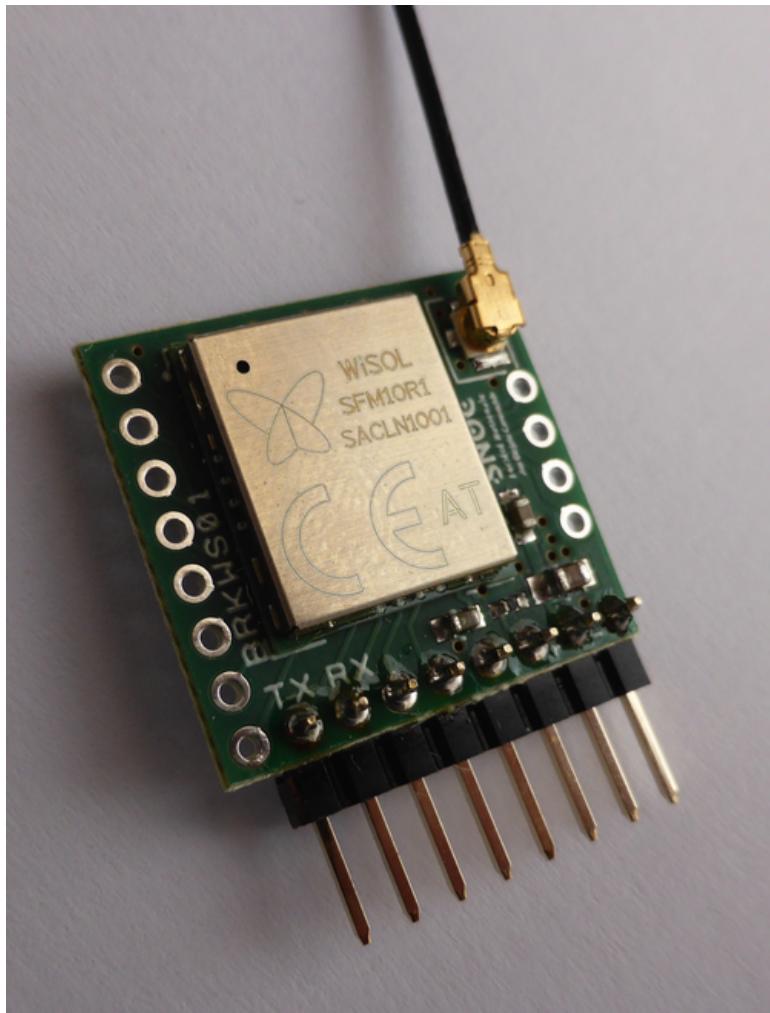
La carte comporte une prise coaxiale miniature.



Confidentialité

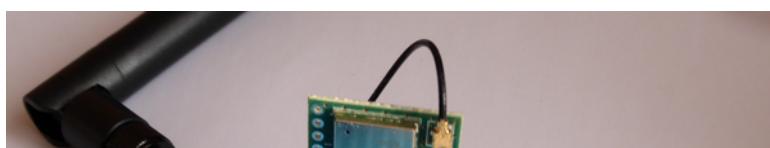


La prise présente à l'extrémité du câble d'antenne va venir se positionner sur la prise de la carte. Il faudra bien l'enfoncer pour assurer le contact et une solidité suffisante de la liaison.

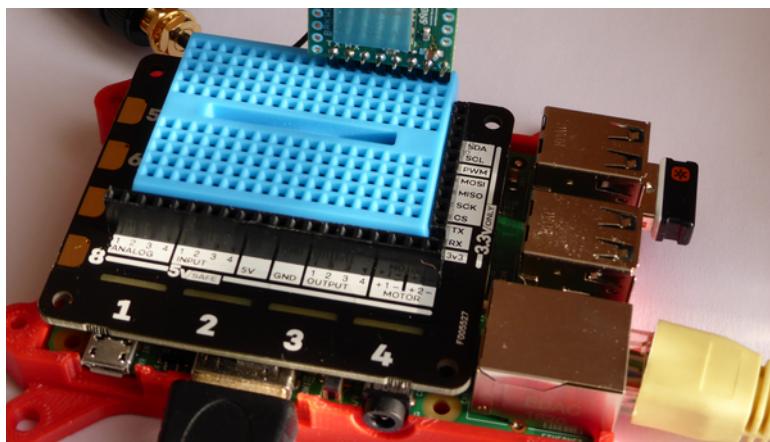


Voilà, la prise est connectée à la carte. On est pratiquement prêt à passer aux essais. S'il se fait tard, reportez les essais à demain matin. Lorsqu'on commence à être fatigué il est plus facile de faire des ~~connexions~~ erreurs et vous risquez de mettre la vie de la carte SigFox en danger.

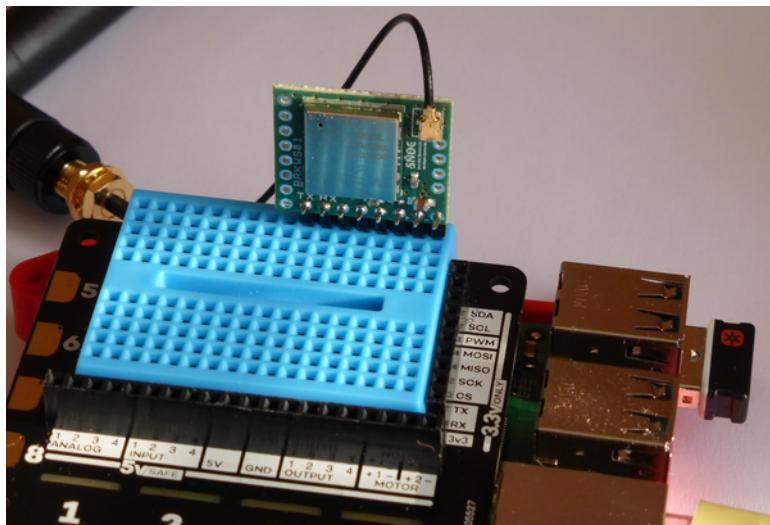
Mise en place de la carte prototype sur le Raspberry Pi



Confidentialité



Bon, j'avoue ce n'était pas simple de connecter la carte SNOC BRKWS01 sur le GPIO du Raspberry Pi. Je ne voulais pas faire un montage volant avec des fils de liaison femelle/femelle. C'est un coup à laisser traîner (volontairement ou involontairement) la carte à un endroit dangereux. J'ai donc choisi d'utiliser une [carte Explorer HAT Pro](#) récemment arrivée. Elle offre la possibilité d'accéder facilement à des ports GPIO intéressants dans ce cas : le 3,3 v et la masse, ainsi que les E/S de l'UART, TxD et RxD sur des connecteurs type Arduino.



L'Explorer HAT Pro est également équipée d'une mini breadboard bien pratique pour des essais rapides de petits montages électroniques, et bien adaptée dans le cas qui nous intéresse. Dans un premier temps j'avais positionné la carte "dans le fond" mais à l'usage il s'est avéré plus judicieux de la monter plus en avant et de connecter les fils derrière. Après... vous ferez bien comme vous voulez ☺

Avant de relier les fils, si vous utilisez ce type de carte, pensez à **faire un test de bon fonctionnement du Raspberry Pi**. Un démarrage du système au minimum pour vérifier qu'il n'y a pas de souci avec la carte seule. Ensuite vous pourrez connecter les fils...

Pensez aussi à **tester le bon fonctionnement du port série** avant de

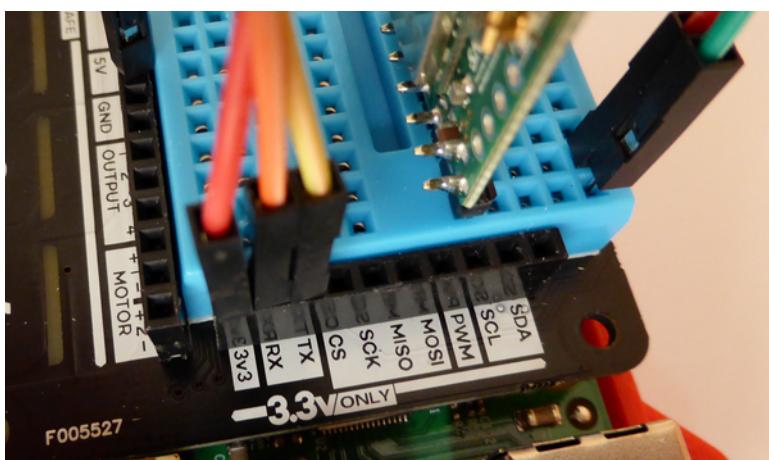
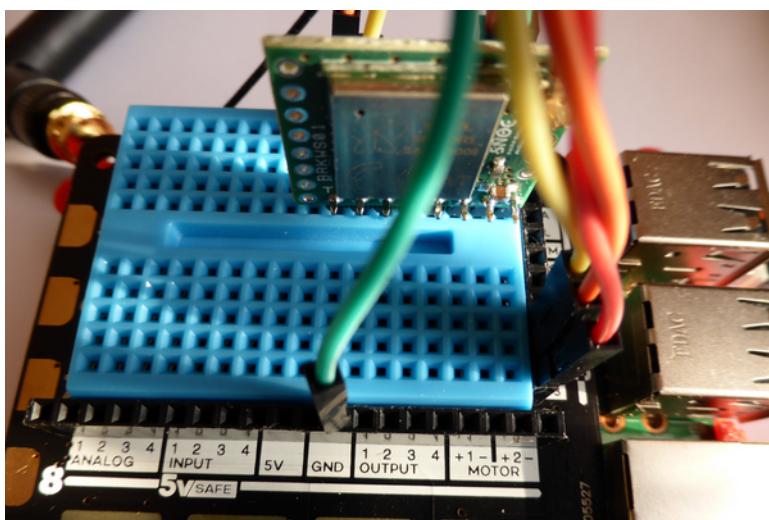
Confidentialité

vous lancer dans les tests ! Pas la peine d'aller plus loin si le port série ne répond pas 😞

Pour ma part j'ai bouclé Tx et Rx et utilisé **putty**.

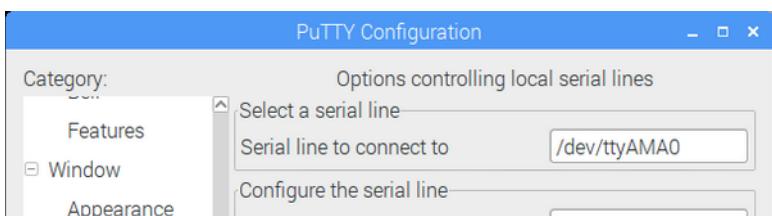
Comme d'habitude, si vous montez, connectez, reliez tout le bazar et que ça ne démarre pas ou que ça ne fonctionne pas... vous ne saurez pas où chercher 😊 Alors allez-y doucement, pas à pas, étape par étape... Vous augmenterez vos chances de réussite !

Connexion et tests

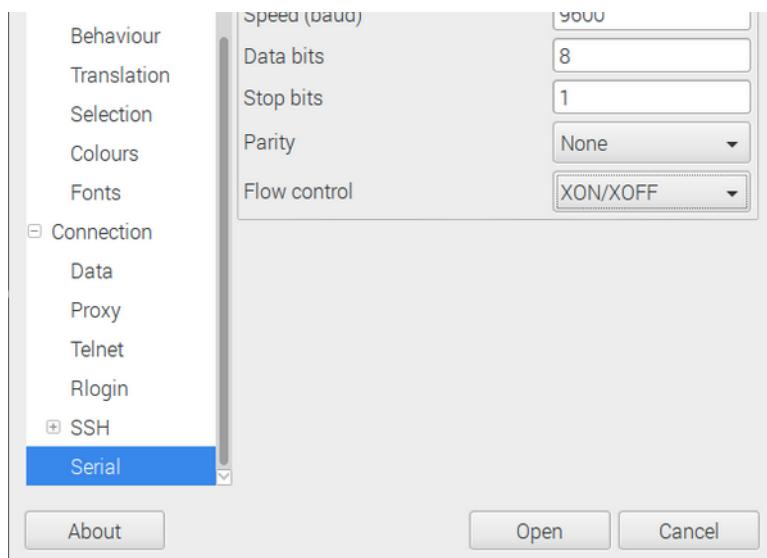


Bon, voilà... tout est connecté on peut passer aux essais. Pour tester la carte SigFox, j'ai choisi d'utiliser **putty** sur le Raspberry Pi (il n'est pas d'origine dans Raspbian, il faudra l'installer).

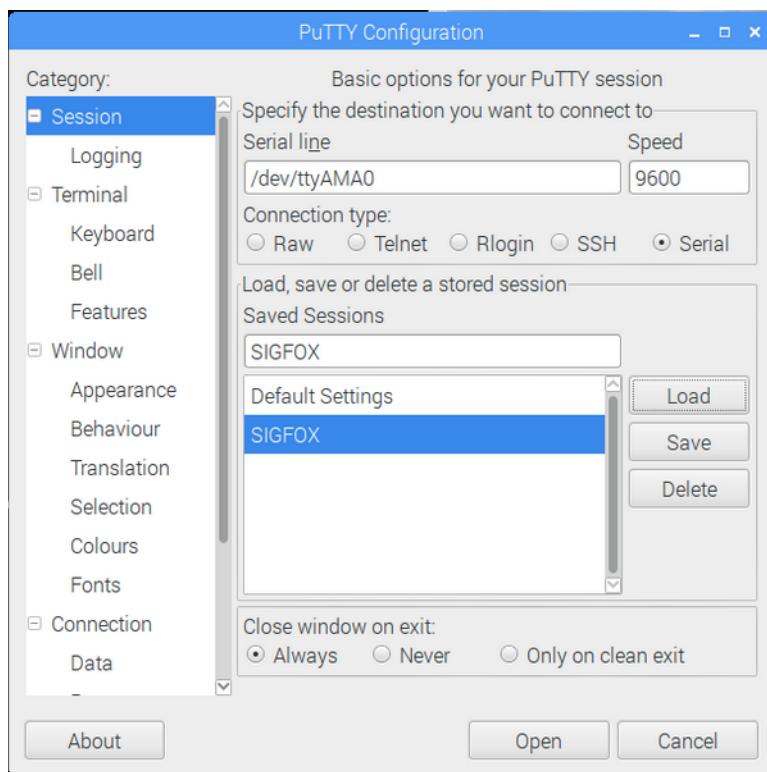
Configuration de putty



Confidentialité



Configurez l'interface série comme ci-dessus.



Puis connectez vous sur le port série (cliquez sur le bouton **Open**). Avant de continuer on va faire une parenthèse, je vous explique ce qu'est l'écho. Si vous connaissez déjà... passez au chapitre suivant 😊

L'écho, qu'est-ce ?

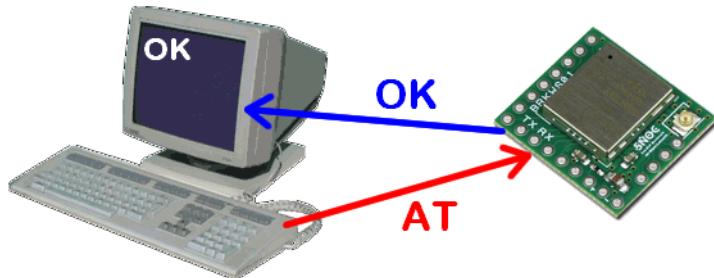
En informatique ça désigne le traitement qui est réservé à **un caractère** envoyé par un terminal (putty dans notre cas) à une unité centrale (la carte SigFox ici). Si une commande est composée de plusieurs caractères, chaque caractère est traité indépendamment.

Confidentialité



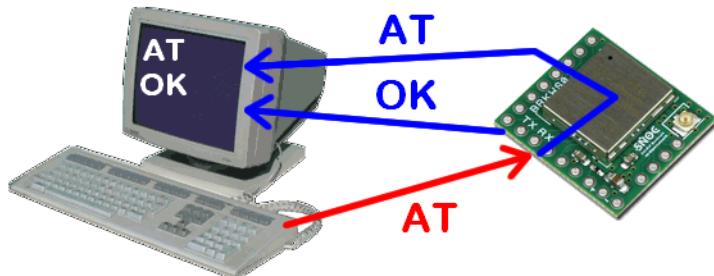
Les plus anciens se souviennent des commandes AT ou Hayes qui permettaient de commander les modems : numérotter (ATDT, ATDP), régler le volume, manipuler les registres internes du modem... Ce sont ces mêmes commandes qui sont utilisées sur les modules SigFox 😊 Ça ne nous rajeunit pas !!

Premier cas : pas d'écho



Le terminal envoie la commande **AT** (aussi appelée **commande Hayes**) à la carte SigFox. En l'absence d'écho, la carte SigFox reçoit la commande et répond **OK** si elle fonctionne... Le problème c'est que si rien ne s'affiche on ne sait pas si c'est parce que la carte ne fonctionne pas ou si la commande AT est réellement partie... (configuration, problème hard...)

Deuxième cas : écho distant

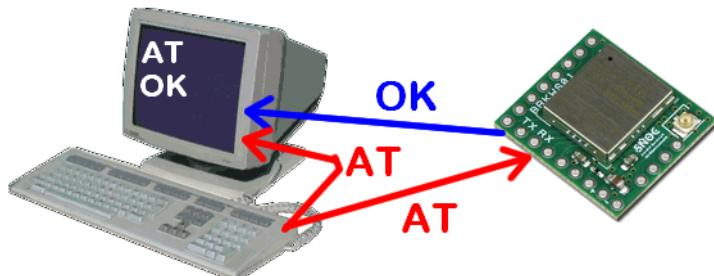


Lorsque l'écho distant est activé (dans ce cas ça se passe au niveau de l'UC distante, donc de la carte SigFox dans notre cas) le terminal envoie la commande **AT** à l'UC distante, qui la renvoie vers le terminal. Dans ce cas on est certain(e) que tout fonctionne, la liaison aller-retour ET la carte SigFox. Ensuite si l'UC distante (la carte SigFox) est prête, elle renvoie **OK**.

Le problème c'est que je n'ai pas trouvé dans les commandes de l'AX-SFEU de commande AT permettant d'activer l'écho 😞 On ne pourra donc pas utiliser l'écho distant. Par défaut on sera donc en mode sans écho comme ci-dessus dans le premier cas.

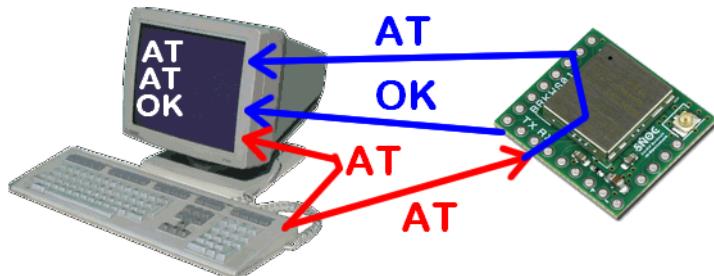
Confidentialité

Troisième cas : écho local



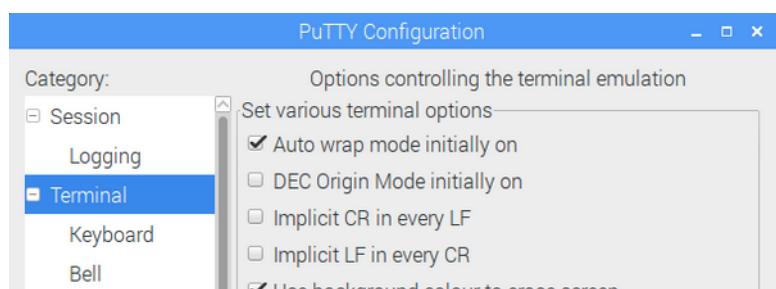
Pour pallier à l'absence de l'écho distant il existe sur les terminaux un **mode écho local** qui envoie sur l'écran les commandes saisies, sans les faire transiter par l'UC distante. On voit donc s'afficher la commande et la réponse de l'UC distante. Bien évidemment, en cas de problème de liaison ou de non fonctionnement de la carte SigFox, on se retrouve avec juste **AT** affiché sur l'écran, sans savoir d'où vient le souci. C'est ce mode que nous utiliserons pour dialoguer avec la carte SigFox. Nous verrons comment activer le mode **écho local** sur putty.

Quatrième cas : écho local ET écho distant

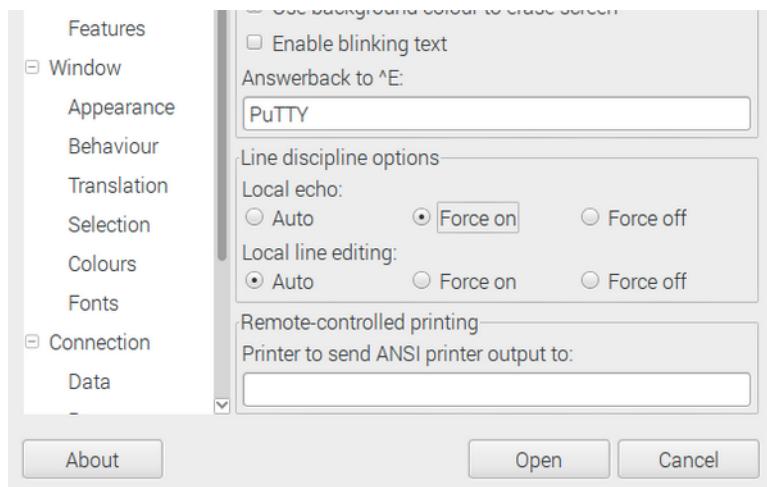


Là c'est la totale. Lorsque vous tapez AT sur le clavier le terminal affiche AT sur l'écran (écho local) et envoie la commande simultanément via le port série jusqu'à l'UC distante. L'UC distante renvoie la commande AT vers l'écran. On se retrouve donc avec deux affichages successifs de chaque commande. Puis la carte SigFox répond OK. Sa réponse s'affiche sous les deux commandes AT.

Mode écho local sur putty

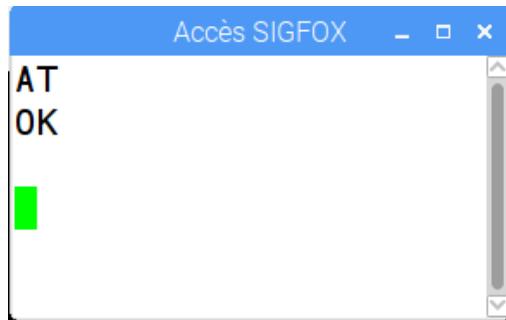


Confidentialité



Dans putty ouvrez la catégorie Terminal et cochez **Force on** dans la rubrique **Local echo**. C'est bon nous sommes prêt(e)s pour tester la carte :

On commence par taper **AT** au clavier, puis **Entrée**



Yesssss la commande AT s'affiche et... ouf la carte répond OK ! On va pouvoir continuer à explorer les possibilités de la carte.

Inscription sur le site web SigFox

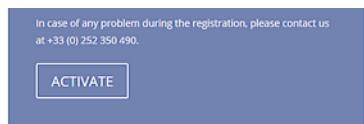
Avant de pouvoir accéder au réseau SigFox, il faut créer un compte sur le site de la société toulousaine. Cette inscription enregistre le numéro de série de votre carte SNOC et autorise le réseau à faire transiter vos données. Avec la carte SNOC, vous avez un an d'abonnement inclus, comprenant 140 messages (jusqu'à 12 octets) en UP par jour, et 4 messages en DOWN par jour.

Accéder à la page d'inscription

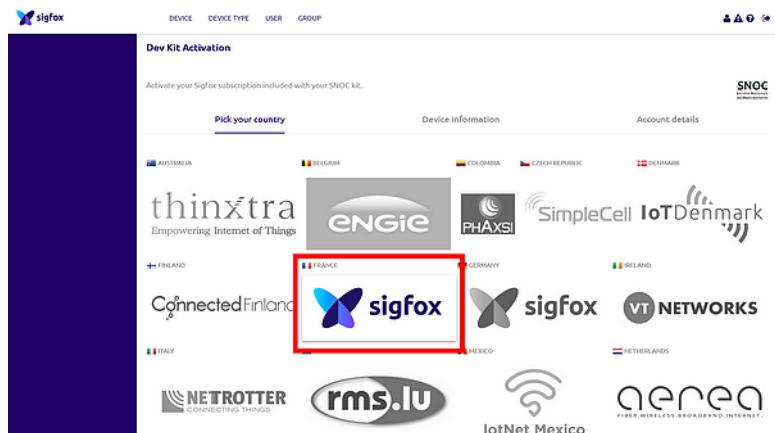
Activation d'abonnement SigFox
Vous venez d'acquérir un produit SNOC disposant d'un abonnement au Réseau SigFox. Vous allez pouvoir l'activer en cliquant sur le bouton. Vous serez redirigé vers le site SigFox où vous allez pouvoir vous enregistrer. Munissez vous des informations fournies avec le produit (ID et PAC).

SigFox subscription activation
You have purchased a SNOC product with a subscription to SigFox Network. You will be able to activate it by clicking the button. You will be redirected to the SigFox website where you can register. Be sure to have the product information to start (ID and PAC).

Confidentialité



Rendez vous sur la page snoc.fr/sigfoxactivate. Cliquez sur le bouton **ACTIVEZ**.



Vous aboutissez sur le site SigFox. Sélectionnez le prestataire de votre pays. Pour la France... c'est SigFox 😊

Device information

DEVICE ID (HEX)

21

PAC

E6E[REDACTED]642

Saisissez les informations relatives à votre carte SNOC BRKWS01 (elles figurent sur les étiquettes du matériel que vous avez reçu).

Account creation

[Already have an account ? Sign in](#)

FIRST NAME *

Francois

LAST NAME *

MOCQ

EMAIL ADDRESS *

[Confidentialité](#)

francois. 

TIMEZONE

Europe Paris

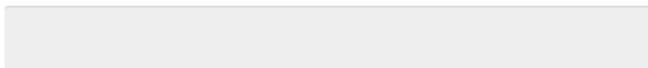
POSITION

Le Creusot

COMPANY NAME *

framboise314

COMPANY ADDRESS



Saisissez les informations nécessaires à la création de votre compte.



Dev Kit Activation finished

Thank you for subscribing your kit ! An email has been sent
to confirm your address and set your password.

Votre inscription est terminée. Il reste à la valider en suivant le lien que vous avez reçu dans un mail à l'adresse que vous avez fournie.

SIGFOX <backend-noreply@sigfox.com>

A moi

[anglais](#) > [français](#) [Traduire le message](#)



Hi ,

To set your password, click on the following link :

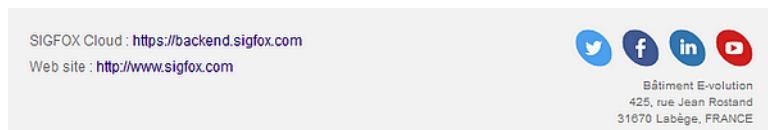
<https://backend.sigfox.com/auth/change-evk-password?id=ced1c61b-8514-9736-1c14>

You will be asked to enter your new password.

This link is valid until 2017-02-10 12:28:43 (GMT +01:00). After this period, you can get a new one by clicking on the "Lost password" link.

Thanks,
SIGFOX Team

 Confidentialité



Cliquez sur le lien dans le message pour terminer votre inscription.

Passwords length must be at least 8 with 1 lower case, 1 upper case, 1 digit and 1 symbol. [?](#)

Saisissez votre mot de passe (2 fois) en respectant les consignes : 8 caractères, comprenant au moins une minuscule, une majuscule, un chiffre et un caractère spécial !

Cliquez pour agrandir.

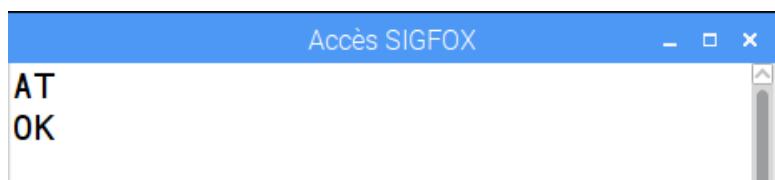
Vous avez maintenant accès à la page de votre périphérique SigFox et allez pouvoir configurer certains paramètres, mais aussi voir les messages envoyés par votre carte.

Votre carte SigFox étant enregistrée sur le réseau, nous pouvons passer aux essais 😊

Envoi de données via SigFox

Mode manuel

On va envoyer 12 octets de données à partir de putty :



Confidentialité

AT\$SF=00112233445566AABBCCDDEE

OK

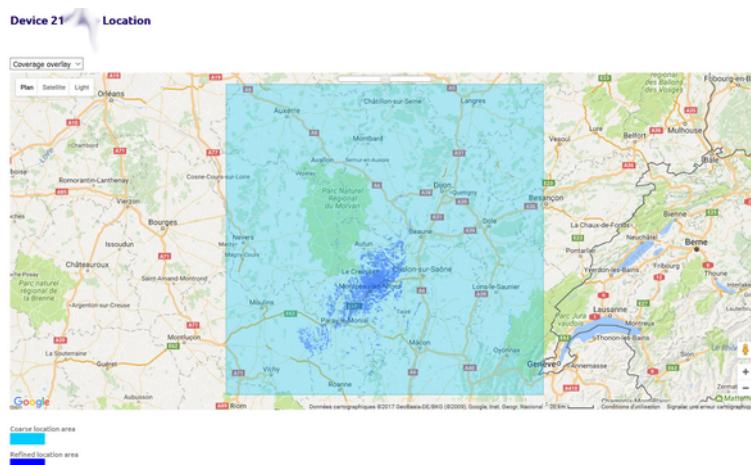
Une commande AT pour voir si la carte répond... OK ! c'est bon ça fonctionne et on entre la commande d'envoi d'un message : **AT\$SF=** suivie des 12 octets en hexadécimal.

Ensuite on appuie sur la touche Entrée et... on attend quelques secondes, le temps que les données soient envoyées par radio (n'oubliez pas qu'on a une grande portée, mais un faible débit). Quand le OK revient, c'est fait, votre message est envoyé !

On vérifie ?

| page 1 | | | | |
|---------------------|-------------------------|----------|--------------|-----------|
| Time | Data / Decoding | Location | Link quality | Callbacks |
| 2017-02-14 17:52:26 | 00112233445566aabbccdee | | | |

Bon... Sur la page de mon module , sur le site de SigFox, on retrouve quoi ? La date et l'heure de mon message et le contenu que j'ai envoyé. Super !



En cliquant sur le symbole circulaire sous **Location**, on a l'affichage d'une carte qui indique la grande région dans laquelle le signal a été capté (carré coloré) et la zone de forte probabilité de la provenance (zones plus sombres). Ça correspond bien à ma position (Le Creusot – 71)

Programme Python

Si vous souhaitez envoyer des messages depuis une application domotique ou depuis une application que vous avez écrite, il est intéressant de le faire de façon automatique. Je vous propose ce programme traduit et adapté du [github de SNOC](#).

Confidentialité

```

#!/usr/bin/env python
# -*- coding: utf-8 -*-

## @package rpisigfox
# Ce script contrôle la carte BRKWS01 pour l'envoi d'un message
# V1.0 Permet l'envoi d'un message normal sur le réseau Sigfox
# syntaxe :
# ./tx.py MESSAGE
# où MESSAGE est une chaîne encodée en HEXA. La longueur est fixe à 4 octets
# Exemple : ./tx.py 00AA55BF envoie les 4 octets 0x00 0xAA 0x55 0xBF
#
import time
import serial
import sys
from time import sleep

class Sigfox(object):
    SOH = chr(0x01)
    STX = chr(0x02)
    EOT = chr(0x04)
    ACK = chr(0x06)
    NAK = chr(0x15)
    CAN = chr(0x18)
    CRC = chr(0x43)

    def __init__(self, port):
        # permet de choisir le port série – par défaut /
        portName = port

        print 'Serial port : ' + portName
        self.ser = serial.Serial(
            port=portName,
            baudrate=9600,
            parity=serial.PARITY_NONE,
            stopbits=serial.STOPBITS_ONE,
            bytesize=serial.EIGHTBITS
        )

    def getc(self, size, timeout=1):
        return ser.read(size)

    def putc(self, data, timeout=1):
        ser.write(data)
        sleep(0.001) # temporisation pour permettre au c

    def WaitFor(self, success, failure, timeOut):
        return self.ReceiveUntil(success, failure, timeOut)

    def ReceiveUntil(self, success, failure, timeOut):
        iterCount = timeOut / 0.1
        self.ser.timeout = 0.1
        currentMsg = ''
        while iterCount >= 0 and success not in currentMsg:
            sleep(0.1)
            while self.ser.inWaiting() > 0 : # bunch of
                c = self.ser.read()
                currentMsg += c
            iterCount -= 1
        if success in currentMsg :
            return currentMsg
        elif failure in currentMsg :
            print 'Erreur (' + currentMsg.replace('\r\n'
        else :
            print 'Délai de réception dépassé (' + currentMsg
        return ''

    def sendMessage(self, message):
        print 'Sending SigFox Message...'

        if(self.ser.isOpen() == True): # sur certaines p
            self.ser.close()

        try:
            self.ser.open()

```

Confidentialité

```

        except serial.SerialException as e:
            sys.stderr.write("Ouverture du port série impossible")
            sys.exit(1)

        self.ser.write('AT\r')
        if self.WaitFor('OK', 'ERROR', 3):
            print('SigFox Modem OK')

        self.ser.write("AT$SF={0}\r".format(message))
        print('Envoi des données ...')
        if self.WaitFor('OK', 'ERROR', 15):
            print('OK Message envoyé')

    else:
        print 'Erreur Modem SigFox'

    self.ser.close()

if __name__ == '__main__':
    if len(sys.argv) == 3:
        portName = sys.argv[2]
        sgfx = Sigfox(portName)
    else:
        sgfx = Sigfox('/dev/ttyAMA0')

    message = "1234CAFE"
    if len(sys.argv) > 1:
        message = "{0}".format(sys.argv[1])
    sgfx.sendMessage(message)
    #time.sleep(600) #mise en sommeil pendant 10 mn

```

Ce programme s'utilise en ajoutant en paramètre les 12 octets à envoyer. Comme parfois l'éditeur de WordPress fait des siennes avec les programmes, vous pouvez aussi [le télécharger ici](#).

Essai du programme SigFox en Python

```

pi@raspberrypi:~ $ ./tx.py 012345678901AABBCCDDEEFF
Serial port : /dev/ttyAMA0
Sending SigFox Message...
Délai de réception dépassé ()
Erreur Modem SigFox

```

Quand ça ne fonctionne pas voilà ce qui se passe. En fait ici j'avais laissé putty connecté au port série. Le programme n'a pas pu accéder au port qui était déjà occupé et a retourné une erreur.

```

pi@raspberrypi:~ $ ./tx.py 012345678901AABBCCDDEEFF
Serial port : /dev/ttyAMA0
Sending SigFox Message...
SigFox Modem OK
Envoi des données ...
OK Message envoyé

```

Cette fois le message a bien été envoyé. Un tour sur le site SigFox pour confirmer :

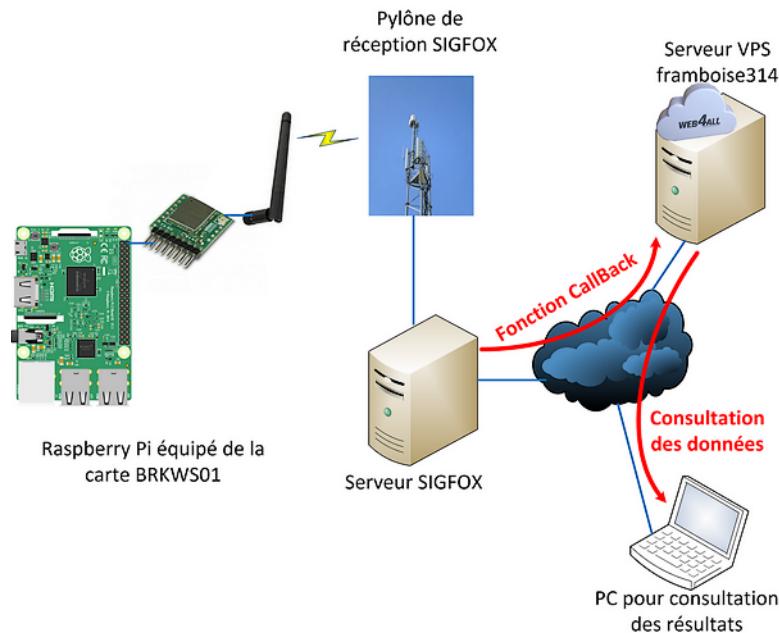
| page 1 | | | | |
|---------------------|--------------------------|----------|--------------|-----------|
| Time | Data / Decoding | Location | Link quality | Callbacks |
| 2017-02-14 18:26:26 | 012345678901aabbcdddeeff | ❖ | ██████ | 🕒 |
| 2017-02-14 17:52:26 | 00112233445566aabbcdddee | ❖ | ██████ | 🕒 |


 Confidentialité

Vous voyez que le précédent message est "descendu" d'un cran et que le nouveau message est bien affiché sur la première ligne.

Envoyer le message vers un site web

Principe



La première partie de l'aventure reste la même : le Raspberry Pi envoie un message à la carte SigFox. La carte transmet les 12 octets maxi) par radio au réseau SigFox. A réception du message le serveur SigFox va transférer les données vers un serveur désigné par l'utilisateur. C'est ce qu'on appelle un "callback".

Paramétrage du callback

Device type SNOC framboise314 kit - Callback edition

Callbacks

| | | |
|-----------------------|---|--------|
| Type | DATA | UPLINK |
| Channel | URL | |
| Send duplicate | <input type="checkbox"/> | |
| Custom payload config | temp:uint:8 humidite:uint:8 pression:uint:8 inclinaison:uint:16 | |
| Url pattern | http://vps.goblink.fr/sigfox.php?id=[device]&data=[data] | |
| Use HTTP Method | GET | |
| Send SNI | <input type="checkbox"/> (Server Name Indication) for SSL/TLS connections | |
| Headers | header | value |

Ok Cancel

Confidentialité

Dans l'onglet "Device Type" du site SigFox, en cliquant sur le nom de la carte, on fait apparaître dans la colonne de gauche un menu comportant l'option Callback. En cliquant sur cette option, on ouvre la fenêtre ci-dessus. Le lien encadré en rouge est appelé après réception du message. Le numéro de la carte SigFox **{device}** et les données **{data}** reçues sont passées en paramètres à la page ***sigfox.php***, présente sur le serveur VPS hébergé chez web4all.

Vous pourrez aussi passer les infos en PUT et en json si ça vous fait plaisir 😊 RTFM

Le fichier ***sigfox.php***

```
<?PHP
// Création de la chaîne contenant les données
$data = '';
$data .= htmlspecialchars($_GET["id"]).PHP_EOL;
$data .= htmlspecialchars($_GET["data"]).PHP_EOL;

// Création du nom de fichier horodaté
$name = 'data_'.date('Y-m-d H:i:s').'.txt';

$maj = fopen($name,"w+");
// On ouvre le fichier en
fseek($maj,0);
// On se place en début de
fputs($maj, $data);
// On écrit dans le fichier
fclose($maj);
// On ferme le fichier
?>
```

Bon, j'avoue que je ne me suis pas foulé 😊 mais l'idée ici c'est de voir si ça fonctionne. Après, quand vous saurez récupérer vos données vous en ferez bien ce que vous voulez... Et puis ça vous fera un peu bosser aussi 😊

```
root@prebeta1:~# cd /var/www/html
root@prebeta1:/var/www/html# ls -al
total 17004
drwxrwsr-x 12 www-data sambauusers      54 Feb 14 20:46 .
drwxrwxr-x  3 root    www-data          3 Sep 25  2015 ..
-rw-r--r--  1 www-data sambauusers      31 Feb 13 14:03 data_2017-02-13 14:03:17.txt
-rw-r--r--  1 www-data sambauusers      31 Feb 13 18:37 data_2017-02-13 18:37:51.txt
-rw-r--r--  1 www-data sambauusers      33 Feb 14 17:52 data_2017-02-14 17:52:31.txt
-rw-r--r--  1 www-data sambauusers      33 Feb 14 18:26 data_2017-02-14 18:26:29.txt
-rw-r--r--  1 www-data sambauusers      30 Feb 14 19:40 data_2017-02-14 19:40:07.txt
-rw-r--r--  1 www-data sambauusers      28 Feb 14 19:48 data_2017-02-14 19:48:28.txt
-rw-r--r--  1 www-data sambauusers      29 Feb 14 19:49 data_2017-02-14 19:49:09.txt
-rw-r--r--  1 www-data sambauusers      34 Feb 14 20:45 data_2017-02-14 20:45:02.txt
-rw-r--r--  1 www-data sambauusers      32 Feb 14 20:46 data_2017-02-14 20:46:52.txt
-rwxr--r--  1 francois sambauusers     607 Feb 14 20:46 sigfox.php
```

Dans le dossier du site web (accès au VPS avec putty) sur le serveur VPS on retrouve le fichier PHP ***sigfox.php*** qui récupère les données et les enregistre dans un fichier (*par sécurité je vais le renommer avant de publier cet article 😊 😊*). On voit aussi les fichiers horodatés créés lors des tests de la carte BRKWS01.

Les données enregistrées

```
root@prebeta1:/var/www/html# cat data_2017-02-14 20:46:5
21xxxx
012345678901aabccddeed
```

Chaque fichier de données contient le numéro de la carte SigFox qui a envoyé le message et sur la ligne suivante le message lui-même.



Confidentialité

Libre à vous de stocker ces infos dans une base de données, d'en extraire des moyennes, des statistiques et/ou de jolies courbes...

Dans 12 octets qu'est ce que tu veux mettre ?

Je viens d'un autre siècle. Non, je n'ai pas profité d'une faille spatio-temporelle ni effectué un voyage dans le temps. Mon premier ordinateur avait 1 Ko de mémoire. Pour les plus jeunes : il n'y a pas d'erreur ! Le ZX 81 possédait une mémoire de 1024 octets. Bon, j'ai vite ajouté une mémoire supplémentaire de 16 Ko sinon on ne faisait pas grand chose. Mais à cette époque, Monsieur, on coupait les Bytes en 4. Je m'explique : la place était tellement chère qu'on jonglait avec les données en mémoire pour occuper le moins de place possible. Décalages, rotations et autres masques à base de Et ou de OU logique étaient de rigueur. Aujourd'hui plus personne ne s'en préoccupe avec les Go de RAM disponible.

Sauf que...

Quand on n'a que 12 octets... Comment qu'on fait ?

Gestion des bits de données

| octet 1 | octet 2 | octet 3 | octet 4 | octet 5 | octet 6 | octet 7 | octet 8 | octet 9 | octet 10 |
|---|--|---|--|---------------------|---------------------|------------------------------|---------------------------------------|--|----------|
| température sol | température air | pression | vitesse vent | direction vent | pluviométrie | temp batterie | tension panneau | courant charge | |
| Premier bit = signe trois bits = premier chiffre quatre bits = deuxième quatre bits = dernière | Premier bit = signe Trois bits = première chiffre Quatre bits = deuxième Quatre bits = dernière | Patm = 480 +/- entre 0 et 127 7bits | de 0 à 100 m/s soit 360 km/h 7 bits (de 0 à 128) | 16 valeurs 4bits | 12 bits 0 à 2048 | 0 à 4.2v 8 bits 0 à 54 | 0 à 7.5v 7bits 0 à 123 12.5v | 0 à 3.3v 8 bits 0 à 128 12.5v | |

Cliquez pour agrandir.

Un exemple sera plus parlant. Ce n'est qu'un exemple, hein, on peut certainement faire mieux et on peut certainement faire pire ! C'est pour faire comprendre à ceux qui n'ont jamais pratiqué ce genre d'exercice...

Une station météo isolée

C'est une demande fréquente dans la région (bin oui, je suis en Bourgogne) de disposer d'une station météo autonome pour surveiller une vigne. Elle doit être alimentée par panneau solaire, et pouvoir renvoyer par SigFox la température sol, la température de l'air, la pression atmosphérique, la vitesse et la direction du vent, la pluviométrie, la tension de batterie, la tension en sortie du panneau solaire et le courant de charge... en 12 octets.

Le bon informaticien replie son ordinateur portable, range sa tablette et sort quelques feuilles de brouillon, un crayon et une gomme. Le mauvais informaticien saute sur son clavier et se met à taper (*n'importe quoi*



Confidentialité

(en général) frénétiquement... Enfin ça, c'est juste mon avis de vieil informaticien.

Le tableau est juste la transcription de réflexions "papier". Après, vous ferez bien comme vous voulez 😊

| octet 1 | octet 2 | octet 3 | octet 4 |
|--|--|---|---------|
| température sol | température air | Pression | |
| Premier bit = signe Trois bits = premier chiffre quatre bits = chiffre 2 quatre bits décimale | Premier bit = signe Trois bits = premier chiffre quatre bits = chiffre 2 quatre bits décimale | Patm - 940 => entre 0 et 127 7 bits | |

Température

- Un bit est nécessaire pour le signe (+ ou -)
- Trois bits codent le premier chiffre (0 à 7)
- Quatre bits codent le deuxième chiffre (0 à 9)
- Quatre bits codent pour la décimale (0 à 9)

Avec 12 bits on code une température entre -79,9 °C et +79,9 °C. On peut réduire le nombre de bits utilisés, au détriment de la facilité de traitement. c'est vous qui verrez en fonction de vos besoins de transmission d'information... Si vous avez eu ce genre de réflexion, n'hésitez pas à en faire part dans les commentaires. Toutes les idées sont bonnes !

Pression

Les [valeurs extrêmes](#) enregistrées mondialement sont de 870 hPa et 1086 hPa. Plus modestement en France on peut tabler sur 940 à 1067 hPa. J'ai donc choisi de représenter la valeur (pression - 940) ce qui donne une valeur entre 0 et 127 qu'on peut coder sur 7 bits.

Pour les autres valeurs du [tableau](#), je vous laisse regarder, critiquer...

En bonus quelques commandes supplémentaires

SNOC fournit dans sa documentation les commandes suivantes :

BRKWS01 Communication command

The module is controlled with serial AT commands sent on TX/RX pins.
Below is the communication specification and AT commands to use.

Serial communication: 9600 bauds, 8bits, 1 stop bit, no parity

AT commands:

| | |
|--|---|
| Communication test: | AT |
| Get Module ID: | AT\$ID=10 |
| Get PAC code: | AT\$ID=11 |
| Send a SIGFOX message: | AT\$SF=XXXXXXXXXXXX (Hexadecimal value) |
| Send a SIGFOX message with downlink frame: | AT\$SF=XXXXXXXXXXXX,1 (Hexadecimal value) |

Confidentialité

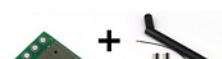
Le jeu de commandes disponibles est assez limité. Heureusement la documentation AX-SFEU vient au secours de l'utilisateur curieux :

| | |
|------------|--|
| AT | Allo ? y-a quelqu'un ? |
| OK | |
| AT\$T? | Température intérieure en dizièmes de degrés |
| 0291 | |
| AT£ERROR: | parse error |
| V | Oups ! j'ai remplacé le \$ par £ le module renvoie une erreur |
| AT\$V? | Retourne la tension actuelle (en mV) |
| 3230 | |
| 3215 | et la tension mesurée lors de la dernière émission de données |
| AT\$DR? | Afficher la fréquence de réception |
| 0869525000 | |
| AT\$IF? | Afficher la fréquence d'émission |
| 0868130000 | |

Présentation vidéo de la SNOC



Conclusion



Kit Carte Breakout Sigfox BRKWS01 +
Antenne

Confidentialité



Toute petite mais pleine de possibilités, cette carte ne coûte que 23,88 € avec les connecteurs, le câble de raccordement et l'antenne, plus un an d'abonnement à SigFox. Rien à redire de ce côté là.

Le fonctionnement est conforme à ce qui est attendu et le traitement des données ne pose aucun problème.

De mon côté j'aurais apprécié d'avoir un câble d'antenne un peu plus long. Il sera peut-être possible d'en trouver dans les accessoires.

Selon le montage qu'on fera, la carte sera au plus près de l'antenne (c'est mieux pour les ondes radio) et reliée au Raspberry Pi via une liaison série. Ici on n'a pas de contrainte de distance aussi forte. Du coup, j'aurais bien aimé trouver un trou de fixation sur le module, pour le monter sur une colonnette à proximité de l'antenne.

Vu la longueur de cet article, j'ai choisi de l'arrêter ici mais il y aura un second épisode car on peut aussi recevoir 4 messages par jour via SigFox (*download*) et piloter à distance certains équipements (*mettre une chaudière en route, régler une température...*). Plus *prosaïquement*, je prévois d'allumer une LED à distance en réglant sa brillance en PWM ☺

Sources

- <https://yadom.fr/reseaux-iot/sigfox/carte-breakout-sfm10r1.html>
- <https://github.com/SNOC/rpisigfox>
- https://yadom.fr/downloadable/download/sample/sample_id/162/
- http://www.onsemi.com/pub_link/Collateral/AX-SFEU-D.PDF

Partager

À propos François MOCQ

Électronicien d'origine, devenu informaticien, et passionné de nouvelles technologies, formateur en maintenance informatique puis en Réseau et Télécommunications. Dès son arrivée sur le marché, le potentiel offert par Raspberry Pi m'a enthousiasmé j'ai rapidement créé un blog dédié à ce nano-ordinateur (www.framboise314.fr) pour partager cette passion.

Confidentialité

Auteur de plusieurs livres sur le Raspberry Pi publiés aux Editions ENI.
[Voir tous les articles de François MOCQ →](#)

42 réflexions au sujet de « Carte de prototypage SIGFOX par SNOC »

**Dodutils**

15 février 2017 à 23 h 23 min

Le truc qui me dérange le plus avec SigFox c'est l'impossibilité d'avoir un ACK (vu le peu de message autorisés dans l'autre sens), on ne peut donc jamais être sûr qu'un message important a été correctement reçu et traité à l'autre bout de la chaîne et le second point se sont les zones blanches parce que la super portée elle est toute théorique surtout quand on a un peu de relief (je parle même pas de montagnes) et la carte de couverture sur le site de SigFox est plutôt fiable de ce côté (et des zones blanches il y en pas mal quand on zoom un peu), ceci dit avec cette antenne je suppose qu'on atteint un niveau SigFox class 0.

Je pense que LoRa aura une meilleure couverture et gagnera le marché sur SigFox, mais l'avenir nous dira si j'ai raison ou pas 😊

En lisant la doc technique je vois T° et Pression dommage il manque l'humidité mais bon on peut toujours l'ajouter via un BME280 ou HTU21D mais le truc qui est dommage c'est que ce chipset soit SPI et pas I2C.

On peut du coup tester cette techno pour pas cher car si le chipset de base coûte 3€, il y a le PCB avec deux ou 3 composants SMD dessus, l'antenne, et l'abonnement annuel SigFox pour 25€ ce qui est tout à fait raisonnable, comparé à d'autres modules à base de TD1205/TD1208 coûtant le double ou le triple.

D'ailleurs quid du tarif de renouvellement de l'abonnement ?

**FK1VA**

16 février 2017 à 11 h 35 min

<http://orivaille.free.fr/MyMeteo/7/MyMeteo7.php>

C'est du PI sur l'ancienne carte SNOC
juste pour montrer ce qu'on peut faire avec 7 octets :):)

pour la guerre LORA / SIGFOX décrite par Dodutils l'avenir nous le dira mais ce sont clairement des produits complémentaires ... quand à l'ACK décrit par Dodutils sur LORA j'espere qu'il n'est pas brouilles sur 869.4 sinon point d'ACK
L'avenir nous dira si LORA respecte bien les DutyCycle de la bande ISM



François MOCQ Auteur de l'article

16 février 2017 à 16 h 43 min

merci pour ce retour Alexandre !
comme quoi avec quelques octets on peut envoyer de l'info 😊
merci pour le lien
73's
François / F1GYT



Dodutils

16 février 2017 à 16 h 46 min

ceci dit comme la taille des messages est limitée, vu le nombre de messages utilisés par jour et si on doit pas en envoyer des tonnes par jours rien n'empêche d'envoyer les infos sur 2 ou 3 messages de long si on peut pas tout faire tenir en un seul.



FK1VA

16 février 2017 à 17 h 10 min

Pas de souci
Par contre point de callback mais une mise à jour journalière des 96 points de mesures
(1 toutes les 15mn)
C'était juste un proof



Confidentialité

**Alexis**

16 février 2017 à 13 h 14 min

+1 pour la question sur le renouvellement de l'abonnement

**FKAVA**

16 février 2017 à 17 h 08 min

Il faut demander à SNOC combien il facture l'abonnement

mais on doit tourner autour de 15€ / an au vue du prix de la puce
WISOL

**msg**

18 février 2017 à 18 h 53 min

Les températures peuvent être encodés sur 7 bits (0 à 127) .
50 valeurs négatives + 77 positives = 127 points.

Encodage = Temp + 50

Décodage = Valeur_encodé - 50

Ex : 25° sera encodé à 75 (25+50)

-10° sera encodé à 40 (-10 + 50)

**msg**

18 février 2017 à 19 h 20 min

Les tensions peuvent être encodés sur 5 bits (0 à 31)

Pour les batteries au plomb (je sais pas pour les autres) , la tension de charge se situe à 15% au dessus de la tension nominale .

Soit $12V \times 1,15 = 13,8 V$

ou $12 \times (1 + 15 / 100) = 13,8 V$

la tension la plus basse se situe à environ 10% en dessous de la tension nominale .

Si la tension tombe à 10V ou moins , la batterie est bonne pour le recyclage .

soit $12 / 1,10 = 10,9V$

Confidentialité

ou $12 \times (1 - 10 / 100) = 10,8V$

Là , ça tombe pas pareil , mais je ne sais pas quels est la bonne formule à utiliser .

Pas spécialiste en pourcentages ni sondages .

Bref de 10,9 V à 14,0 V (109 à 140) , ça doit suffire (soit 32 points).

Encodage = (V x 10) - 109

Decodage = (Valeur_encodé +109) / 10



msg

18 février 2017 à 19 h 39 min

ou sur 8 bits (0 à 255) si on veut plus de prévision .

11,45 V à 14,00 V (1145 à 1400 soit 255 points)

Ça dépend de la précision du CAN utilisé .



fk1va

18 février 2017 à 22 h 37 min

<https://aws.amazon.com/fr/blogs/iot/connect-your-devices-to-aws-iot-using-the-sigfox-network/>



Loic

26 février 2017 à 18 h 21 min

Bonsoir François

Encore merci pour tous vos articles et particulièrement celui ci concernant SIGFOX. Je cherche une solution de géolocalisation pour mon raspberry pour un "projet mobile". Je suis donc intéressé par la possibilité qu'offre SIGFOX de transmettre des données en étant peu gourmand en consommation d'énergie et à bas coût.

J'aurais besoin de vos conseils alors je me lance :

- Existe t il une solution de géolocalisation pour le raspberry couplant un capteur GPS et la transmission via le réseau SIGFOX ?

Confidentialité

- Les cartes SIGFOX permettent elles, à elles seules, d'obtenir une localisation précise du module ?

- J'ai trouvé un module grove gps pour PI l'avez vous déjà utilisé ?
(<http://www.lextronic.fr/P28809-module-grove—gps.html>), les données recueillies pouvant transiter via la 3G sous réserve de couverture (en réalité réseau 2G)

Merci d'avance à tous des solutions que vous pourriez me proposer.

Cordialement

Loic



fk1va

26 février 2017 à 20 h 55 min

Bonjour je vais répondre a la place de Francois ...

il existe des dizaines d'objets <https://www.hidnseek.fr>,
<http://www.capturs.com/fr/>, <http://www.sensolus.com> maintenant si c'est pour faire purement du GPS avec du raspberry alors mieux vaut vous tourner vers le TD1205 qui intègre un GPS dans une puce SIGFOX que vous trouverez ici <http://www.telemetrieshop.nl/product/11/Telecom-Design-TD1205-power-Sigfox-gateway+> par exemple

En ce qui concerne le positionnement des objets sur les cartes SIGFOX a elle seules (c'est à dire sans GPS) c'est le projet SIGFOX SPOT'IT mis en service il y a 10j
<https://www.sigfox.com/fr/news/sigfox-to-transform-global-asset-tracking-spot-it-world-s-lowest-cost-internet-of-things-iot>

en espérant avoir répondu

Alexandre



Eric DAUNAY

19 mars 2017 à 9 h 44 min

Bonjour,
Merci pour ce super tuto !
Je débute dans le domaine donc je recherche des infos.

Confidentialité

Je cherche à faire un objet connecté de taille réduite et autonome,
qui enverrait au réseau Sigfox des infos provenant de capteurs.
Peut-on connecter des capteurs directement à la breakout Wisol
sans passer par un raspberry ou un arduino ?
Merci !
Eric

**François MOCQ**

Auteur de l'article

20 mars 2017 à 10 h 51 min

Bonjour Eric
Au vu de la doc... je dirai oui
mais il n'y a pas beaucoup d'infos disponible.
Je serais plutôt favorable à l'utilisation d'un arduino (ou d'un Pi) qui
pilotera la carte SIGFOX.
La doc disponible est là très importante 😊
cordialement
François

**Eric DAUNAY**

20 mars 2017 à 12 h 24 min

Bonjour François,
Le problème de l'utilisation d'une carte Arduino ou Raspberry, c'est
la consommation. Je voudrais idéalement au moins 1 an d'autono-
mie avec une batterie LiPo !
La seule info que j'ai trouvé pour le moment sur le net pour la
connexion de capteurs en direct, c'est sur ce site :
<https://www.awelty.fr/blog/actualite-des-nouvelles-technologies-amiens-somme-picardie/un-capteur-de-niveau-d-eau-connecte.html>.
Ce n'est pas le même module mais je pense que la carte Wisol fait la
même chose. A en croire ce tuto, on ne peut détecter que des ni-
veaux haut ou bas, j'aurais voulu idéalement pouvoir envoyer des
valeurs provenant de capteurs.
Je suis en train de regarder de près les modules de chez PYCOM
(www.pycom.io) car ils ont l'air d'avoir une très faible consommation.
Cordialement,
Eric


Confidentialité

**Dodutils**

20 mars 2017 à 16 h 09 min

Un arduino en mode deep sleep c'est 4uA après faut voir son cycle d'allumage.

On peut aussi utiliser un Arduino 8Mhz eu lieu de 16Mhz pour économiser quand il est éveillé, tout dépend du besoin de mesure quand au WSSFM10R1 il utilise un chipset AX-SFEU qui possède 4 I/O analogiques.

Sinon si le projet peut se contenter d'un ATTiny 85 ça consommera encore moins.

Après ça dépend des capteurs que l'on veut utiliser, par exemple un IR ça consomme rien et ça peut allumer l'Arduino qui pilotera l'envoi du message SigFox et s'éteindra après.

**Eric DAUNAY**

20 mars 2017 à 16 h 18 min

Je ne connaissais pas ce mode deep sleep, intéressant !
J'ai besoin au minimum d'envoyer une ou 2 fois par jour les informations provenant de 2 accéléromètres.
Est-il possible de programmer l'Arduino pour qu'il s'allume à cette fréquence ?
L'idéal serait également de pouvoir envoyer une ou 2 fois par jour les valeurs maximales de jauge de déformation. Sachant que les valeurs provenant des jauge devront être analysées tout au long de la journée, l'Arduino ne pourra pas être mis en veille ?

**Dodutils**

20 mars 2017 à 17 h 01 min

Pour sortir l'Ardu du deep sleep c'est programmable par timer interne ou par interrupt externe

**Eric DAUNAY**

Confidentialité



22 mars 2017 à 18 h 35 min

Bonjour,

J'ai bien progressé dans mon projet et je pense que je vais partir sur un puce ATMega328 seule et utiliser le mode SLEEP_MODE_PWR_DOWN. La consommation devient alors très faible vu que je ne réveillerais la puce qu'un fois par jour.

J'ai une autre question que j'ai un peu honte de poser : mes études sont un peu loin et je ne me rappelle plus comment calculer le nombre d'octets que représente une valeur décimale convertie en hexa. Quelqu'un peut me donner une piste pour me renseigner ?

Merci !

Eric

**Dodutils**

22 mars 2017 à 18 h 46 min

Tout dépend de la précision de la décimale, un Float c'est 4 octets et un Double c'est 8 octets par exemple.

**Eric DAUNAY**

22 mars 2017 à 19 h 12 min

J'ai besoin de mesurer un angle entre +180 degrés et -180 degrés, sans chiffre après la virgule.

**Dodutils**

22 mars 2017 à 21 h 38 min

Donc pas de décimale juste un entier signé mais et comme il va de -180 à +180 ça ne sera pas un seul octet car un entier signé sur 8 bits (1 octet) car ça donnerait -127 à +127 donc ça sera 16 bits (2 octets) de -32767 à +32767

**Eric DAUNAY**

Confidentialité



22 mars 2017 à 21 h 50 min

OK, compris.

Merci bcq !

Eric

**marc**

15 avril 2017 à 18 h 07 min

Bonjour,

j'ai très bien suivi la procédure, et le port série fonctionne correctement.

Par contre, la commande at ne me renvoie rien.

Une idée ?

merci

**Eric DAUNAY**

24 avril 2017 à 8 h 20 min

Bonjour,

Avez vous essayé de mettre en veille le module ?

Les essais que j'ai fais avec la commande AT\$P=1 ne semble pas fonctionner correctement : le module n'émet plus mais la consommation ne chute pas.

Cordialement,

Eric DAUNAY

**François MOCQ**

Auteur de l'article

24 avril 2017 à 9 h 33 min

Bonjour Eric

non je n'avais pas fait ce test et actuellement le matériel n'est pas dispo

éventuellement essayez de contacter directement SNOC ?

cordialement

François

 Confidentialité

**krzysiu**

7 juillet 2017 à 9 h 18 min

Bonjour à tous,

merci pour ce tuto qui m'as bien aidé.
comme je suis sur un RPI3, l'exemple python ne fonctionnais pas...
alors comme google est notre amis, j'ai recherché le pourquoi du
comment et je suis tombe sur cet article

<http://spellfoundry.com/2016/05/29/configuring-gpio-serial-port-raspbian-jessie-including-pi-3/>

qui explique que pour le RPI3 il faut utiliser /dev/ttYS0 et pas
/dev/ttYAMA0 qui est pour le bluetooth

il me reste encore un problème à résoudre , c'est putty qui ne réagit
pas à mon clavier, si quelqu'un à une idée ?

Merci francois pour tes articles et à bientôt

bonne journée
Christophe

**François MOCQ**

Auteur de l'article

7 juillet 2017 à 9 h 27 min

Bonjour Christophe
Pourtant le programme a bien fonctionné sur un Pi 3 (tous les progs
que je publie sont testés)
après il y a effectivement un souci de croisement (fait discrètement
par la Fondation) des ports du Pi 3 et du Pi Zero W pour le
Bluetooth)
j'en avais aussi parlé ☺ <https://www.framboise314.fr/le-port-serie-du-raspberry-pi-3-pas-simple>
le principal c'est que ça fonctionne
"putty" ne réagit pas... c'est à dire rien de rien ? le login du Pi s'affiche
et puis plus rien ?
cordialement
François

Confidentialité

**krzysiu**

7 juillet 2017 à 14 h 04 min

oops j'avais pas vu... j'y vais de ce pas 😊

pour putty effectivement le curseur et rien, pourtant j'ai bien les paramétrages que toi. j'avais essayer la carte sur un PC avec le cordon qui va bien et putty à bien fonctionné.

c'est bizarre, il doit y avoir un truc que j'ai pas vu.

Christophe

Christophe

**Jérôme**

12 septembre 2017 à 16 h 04 min

Bonjour à tous,

En premier lieu je tiens à vous remercier pour le tuto qui m'a jusque là été très utile.

En suite j'ai une petite demande d'aide, dans le cadre d'un projet je souhaiterai envoyer les datas transmises par un arduino+sigfox vers une db firebase via je suppose un callback. Et c'est là que ça coince car malgré plusieurs essais je n'y arrive pas.

Auriez-vous déjà tenté cette opération et si oui pourriez-vous me transmettre les configurations des différentes étapes.

D'avance merci.

Jérôme

**François MOCQ**

Auteur de l'article

12 septembre 2017 à 17 h 47 min

Bonjour Jérôme

oui c'est ce que propose cette partie de l'article :

<https://www.framboise314.fr/carte-de-prototypage-sigfox-par-snoc/#Parametrag...>



Confidentialité

je renvoyais les infos vers un serveur VPS hébergé chez Yulpa
depuis je n'ai pas repris ces fonctions....
cordialement
François

 **Jérôme**
12 septembre 2017 à 18 h 50 min

Mmmmhmmmh j'ai déjà suivi ce tuto sans succès, peut être il y a eu erreur de ma part.
Je vais reprendre tout cela depuis le début .

Merci pour votre réponse rapide.

A bientôt

Jérôme

 **joris**
16 mars 2018 à 17 h 11 min

Bonjour,
déjà je vous remercie pour votre tuto il m'aide beaucoup mais j'ai un petit problème, quand j'envoie la commande pour la trame (AT\$SF=11223235465) sigfox ne reçois rien et ca me renvoie ERROR:
parse error.
je suis avec la raspberry 3 B

 **joris**
16 mars 2018 à 17 h 20 min

finalement ça a répondu je ne doit tous simplement pas pouvoir en-
voyer des lettre je ne sais pas pourquoi :/

merci beaucoup a bientôt

 Confidentialité

**joris**

16 mars 2018 à 17 h 25 min

ah ben j'ai parler trop vite xD sigfox ne reçoit rien alors que quand j'envoie ma Trame il me répond 'OK' . je ne comprend plus rien -_-'

**andré martin**

9 septembre 2018 à 10 h 44 min

je découvre et c'est excellent!!! andré

**François MOCQ**

Auteur de l'article

10 septembre 2018 à 8 h 45 min

merci 😊

**Jean-Sylvanus Olympio**

30 novembre 2018 à 11 h 59 min

Merci François pour tous ces éléments techniques.
Notre société BlueCloud, développe des IoT et nous sommes partenaires SigFox.
Nous venons d'intégrer le module de communication Wisol WSSFM10R1.
Cela fonctionne remarquablement bien.
notre site <http://www.blue-cloud.fr>
Je reste disponible pour tout complément d'information.
Merci encore pour votre excellent travail.
Cordialement,

Jean-Sylvanus Olympio

**ayoub**

9 mars 2019 à 22 h 43 min

 Confidentialité

Bonsoir,
Merci beaucoup pour ce tuto super intéressant.
peut on envoyer un message vide sans connecter le module avec
une uc (en utilisant uniquement le module)
Merci pour tout.

**Alexandre**

2 octobre 2019 à 22 h 25 min

Bonsoir

Merci François pour ce super tuto. Le temps a passé depuis, avez-vous pu tester la carte BRKWS20R1 proposée désormais par SNOC. Vraisemblablement, la puce ARM est programmable et permettrait donc ne plus faire intervenir un raspberry ou un arduino sur un produit final et ainsi gagner considérablement en autonomie.

Merci d'avance
Alexandre

**François MOCQ**

Auteur de l'article

3 octobre 2019 à 10 h 52 min

Bonjour Alexandre
Non, je n'ai pas testé cette carte, SNOC ne me l'a pas envoyée pour tests et... je ne peux pas acheter tout le matériel qui sort (il y en a tellement !)
cordialement
François

Ce site utilise Akismet pour réduire les indésirables. [En savoir plus sur comment les données de vos commentaires sont utilisées.](#)

[Haut de la page](#)

 Confidentialité