Un guide pratique pour mettre en place rapidement un tracker et/ou un iGate APRS via LoRa.

Présentation

Dans ce document, on va expliquer concrètement comment mettre en place un module iGate ou tracker pour faire de l'APRS via LoRa.
Le LoRa est une technique de modulation propriétaire basée sur le *spectre étalé par sauts de glissements de fréquences*.
Dans ce document, on va parler de ce qu'il est nécessaire d'avoir avec vous pour faire fonctionner les modules ainsi que la marche à suivre pour les programmer.

Pour ceux qui seraient intéressés d'aller plus loin dans le sujet, des cadres bleus apparaîtront dans ce document. Ils contiennent des informations plus poussées mais leur lecture n'est pas requise pour comprendre ce document et utiliser l'APRS via LoRa.

Le Matériel

La plupart des modules tournent avec des microcontrôleurs ESP32, mais sont compatibles avec Arduino. Les modules recommandés sont les suivants :

- Pour un iGate, le module TTGO LoRa32 v2.1 de chez LilyGo
- Pour un tracker, le module TTGO T-Beam V1.1 de chez LilyGo également

Les modules peuvent être commandés entre autres sur Aliexpress.
Voici deux liens Aliexpress vers les pages des modules. Attention à sélectionner la bonne fréquence sur la page ! Tracker : https://fr.aliexpress.com/item/4001178678568.html, iGate : https://fr.aliexpress.com/item/32915894264.html.
Le module iGate coûte une vingtaine d'euros avec des frais de livraison d'environ 6 €. Le tracker coûte une quarantaine d'euros et les frais de livraison sont inclus. Si le montant total de la commande est inférieur à 150 €, il n'y aura normalement pas de frais de douane.

Les points d'attention particuliers sont les suivants :

- Les modules ne sont conçus que pour une seule bande de fréquence. Pour l'APRS via LoRa, la bande utilisée est celle des 70 cm (433,775 MHz). Il faut donc commander des modules prévus pour cette bande et non 868 MHz ou 900 MHz.
- Tous les modules ne sont pas forcément fournis avec un petit écran OLED. Il faut de préférence en choisir un qui vient avec un écran déjà soudé sur la carte.

- Les trackers sont fournis sans batterie, mais possèdent un support pour batterie au lithium (tension de 3,7 V) taille 18 650 dite « flat-top » (il n'y a pas de « bouton » sur le terminal « + »).
- Les antennes fournies par défaut avec le module ne sont pas particulièrement de bonne qualité.

Les modules 433 MHz sont pourvus d'une puce Semtech SX1278 alors que les autres modules sont pourvus d'une puce Semtech SX1276. La première est conçue pour les fréquences de 137 à 525 MHz et la deuxième pour les fréquences de 137 à 1020 MHz. Cependant, il y a fort à parier que la sortie RF qui couvre la bande 433 MHz dans la puce SX1276 n'est pas connectée à la sortie RF du module, le rendant incompatible avec l'usage LoRa-APRS.

Le logiciel

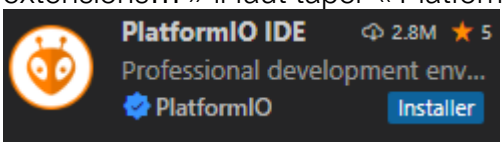
Après avoir vu quel matériel acheter et comment, il reste à programmer les modules et les configurer.

Logiciel de programmation

Pour pouvoir programmer le module, il faut installer PlatformIO (https://platformio.org/) qui est un module supplémentaire du logiciel Visual Studio Code. Voici la marche à suivre :

1. Il faut premièrement installer le logiciel Visual Studio Code au lien suivant : (https://code.visualstudio.com/).
2. Ensuite, il faut démarrer le logiciel installé (l'icône ressemble à un alpha bleu)
3. Il faut maintenant installer le module PlatformIO.

   Pour cela il faut cliquer sur cette icône à gauche de la fenêtre : 
4. En haut du volet qui vient de s'ouvrir, dans le champ « Rechercher des extensions… » il faut taper « Platformio IDE ».

5. Un des résultats de recherche devrait ressembler à ceci :
6. Il faut ensuite cliquer sur le bouton « Installer »

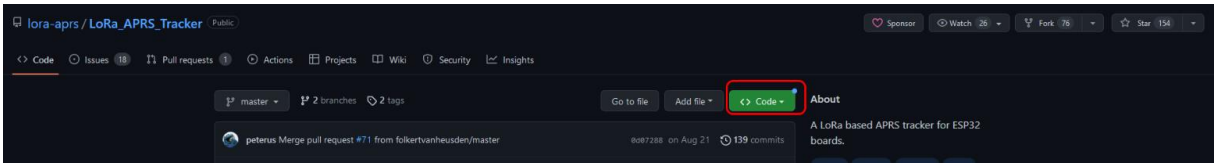
L'extension va alors s'installer automatiquement. Cela peut prendre un peu de temps et il sera peut-être nécessaire de redémarrer le programme. Si Visual Studio Code demande à être redémarré, il faut le faire.

À ce stade-ci, une nouvelle icône devrait être apparue à gauche. Elle ressemble à un petit extra-terrestre : 

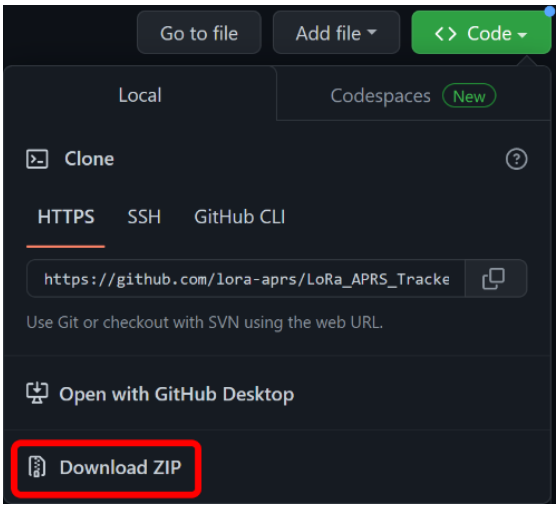
Le firmware :

Le firmware est le logiciel qui est exécuté sur le module. Il gère la communication entre les différents composants tels que le module LoRa, la puce GPS, l'écran… Il va falloir le télécharger, le configurer, le compiler et le programmer sur le module. Le firmware du module et du tracker sont différents, mais la procédure à suivre est la même.

1. Suivre ce lient (https://github.com/lora-aprs/LoRa_APRS_Tracker) pour le tracker et suivre ce lien (https://github.com/lora-aprs/LoRa_APRS_iGate) pour l'iGate
2. Dans la page, cliquer sur le bouton au-dessus à droite de la liste des fichiers marqués « code ».

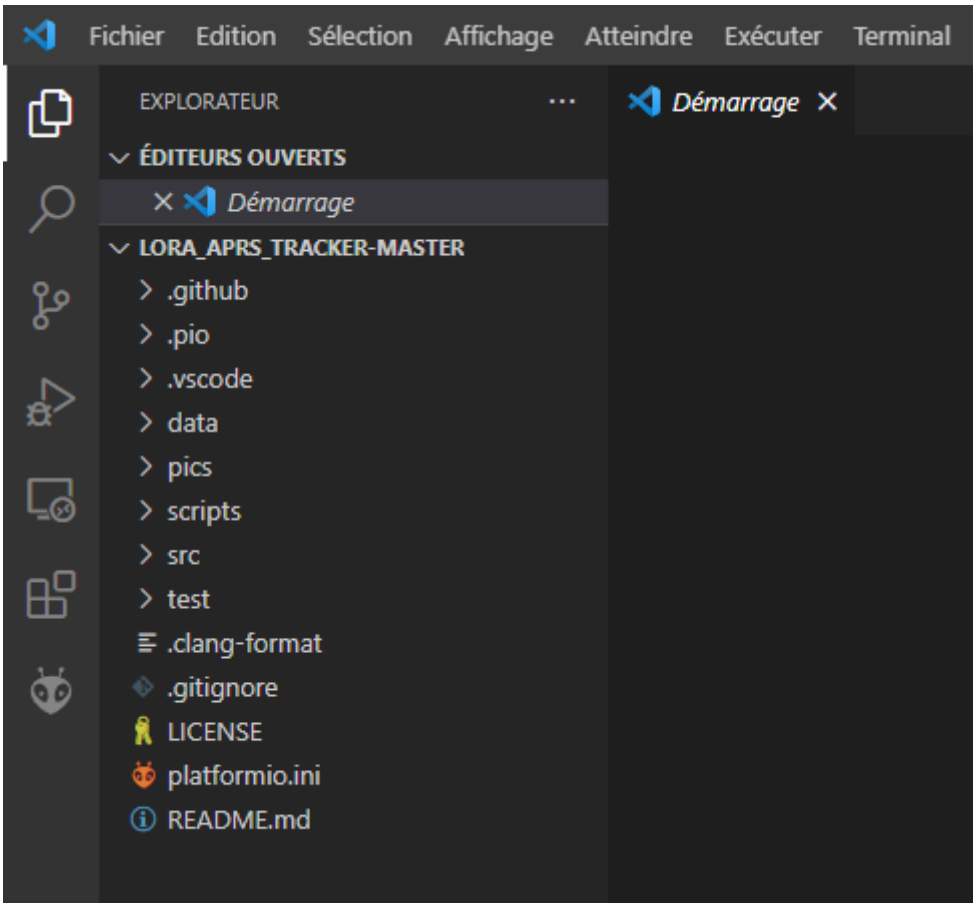


3. Dans la fenêtre qui s'ouvre, cliquer sur « Download ZIP ».

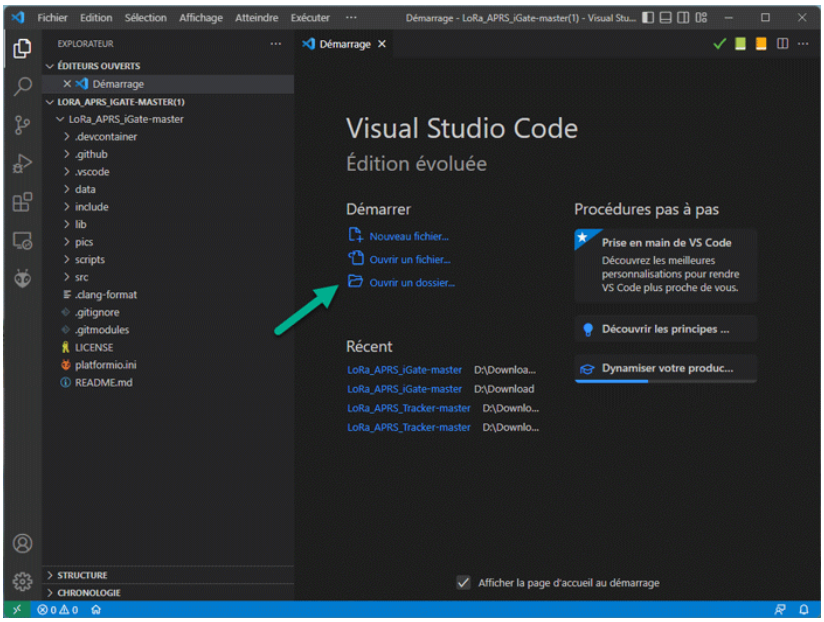


4. Un fichier au format ZIP (archive) va être enregistré sur l'ordinateur. Il faut le décompresser.
5. Placer le contenu (un dossier nommé « LoRa_APRS_Tracker-master » ou « LoRa_APRS_iGate-master ») dans un dossier connu sur l'ordinateur (Documents, Bureau…).
6. Dans VS Code, cliquer sur Ficher -> Ouvrir le dossier. Dans la fenêtre qui s'ouvre, sélectionner le dossier de l'étape précédente. Si un dialogue s'ouvre demandant si vous faites confiance aux auteurs de ce dossier, cliquer sur oui. Le panneau de gauche devrait ressembler à la capture ci dessous :
   Si le logo de l'extension PlatformIO (la tête d'extra-terrestre) n'est pas encore

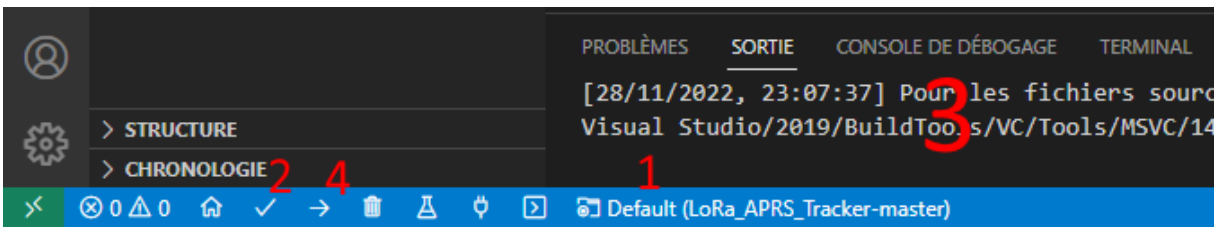
apparu, il faut peut-être attendre une petite minute. Si après ça elle n'apparaît toujours pas, il faut vérifier que l'extension est bien installée.



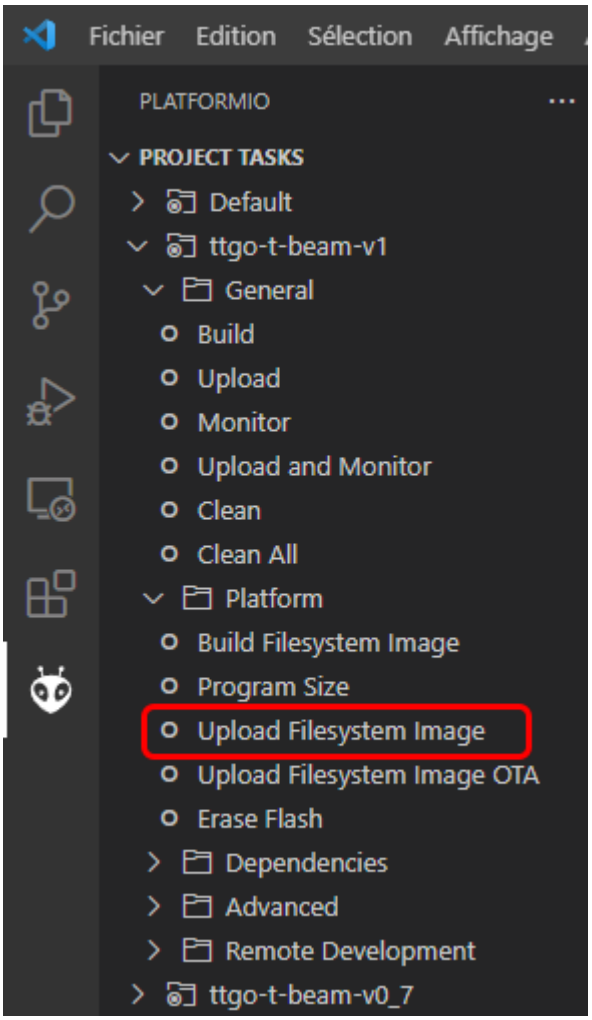
7. Après avoir décompressé le fichier LoRa_APRS_Tracker-master ou LoRa_APRS_iGate-master, regarder si le dossier possède encore un dossier du même nom à l'intérieur, c'est lui qu'il faut ouvrir.
En fait, le dossier qu'on ouvre doit posséder un fichier nommé « platformio.ini » à sa racine (c'est à dire, pas dans un sous-dossier).



8. Dans le bas de la fenêtre, dans le bandeau bleu, se trouve maintenant une série de boutons. Cliquer sur Default (1 sur l'image ci-dessous). En haut de la fenêtre, cliquer sur « env : ttgo-t-beam-v1 » (pour le tracker) ou « env : lora_board » (pour l'iGate).
9. Cliquer ensuite sur la coche (2 sur l'image ci-dessous) pour compiler le code. La console en bas de la fenêtre (3 sur l'image ci-dessous) va afficher la progression. Cela peut prendre une ou deux minutes. L'étape est sensée se terminer avec « SUCCESS » écrit en vert. Si « FAILED » apparaît, il se peut qu'il faille attendre un peu avant de ré-essayer ou redémarrer le logiciel. Si les fichiers de code source n'ont pas été modifiés, l'étape est censée se terminer par un succès. Si ce n'est pas le cas, supprimer tout le dossier et recommencer à l'étape 4.
10. Il faut ensuite configurer le module. Pour cela dans la racine ouvrir «data» puis {}is-cfg.json. Le contenu des fichiers de configuration est décrit plus loin dans ce document.
11. Brancher le module à l'aide d'un câble micro-USB type B. Programmer le module en cliquant sur la flèche (4 sur l'image ci-dessous). À nouveau, l'étape doit se terminer avec « SUCCESS ». Si ce n'est pas le cas, vérifier que le câble USB est bien connecté. Si l'erreur persiste, essayer avec un autre câble.



12. Cliquer ensuite sur le logo de PlatformIO (la tête d'extra-terrestre). Dans la liste à droite de l'icône, cliquez sur « Upload Filesystem Image »



13. À nouveau, cette étape est censée se terminer par « SUCCESS ». Si c'est le cas, le module devrait être programmé et l'écran devrait maintenant afficher son statut.

La configuration :

Pour configurer le module, il faut modifier un fichier texte qui se trouve dans le dossier « data ». Ce fichier porte l'extension « .json ». Les options disponibles sont listées par type de module avec une brève description et des valeurs recommandées. Les options qui doivent être modifiées pour configurer un module sont écrites en rouge dans ce document. Les autres sont utilisées pour des usages avancés, voire même exotiques. Les fichiers sont dans le dossier « data », dans le dossier principal du firmware.

Tracker

Fichier « ./data/tracker.json »
• Beacons→

- Callsign → Votre indicatif suivi du SSID classique APRS. Pour le SSID, il est recommandé d'utiliser : « -7 » pour un tracker portable, « -9 » pour une station

mobile telle qu'une voiture, « -11 » pour les aéronefs et finalement « -1 », « -2 », « -3 », « -4 » sont des SSID génériques.

- Path → Il s'agit du chemin classique APRS. Dans le cas de l'APRS, parce que les trames sont longues et ont rapidement une portée importante, il est recommandé de n'utiliser que « WIDE1-1 ».
- Message → Message qui sera ajouté à chaque trame transmise. Il est recommandé de ne pas mettre de message (« message » : « ») parce que ça allonge drastiquement la durée de transmission et réduit fortement la probabilité que le message soit décodé sans erreur.
- Symbol → caractère qui correspond au logo que l'on veut voir affiché sur les cartes APRS.
  « [» représente un piéton, « > » représente une voiture, « ' » représente un avion…
- Overlay → Choisit le jeu de logos que l'on va utiliser pour la sélection du symbole. « / » pour la table primaire, « \ » pour la table secondaire.
- Smart Beacon → Cette fonctionnalité laisse adapter le temps entre deux émissions selon son déplacement
    -
        - Active → « false » pour désactiver la fonctionnalité, « true » pour l'activer (recommandé)
        - Turn_min → changement de cap minimum en degrés avant d'envoyer un nouveau paquet
        - Slow_rate → nombre de secondes entre deux paquets pour une vitesse inférieure ou égale à la vitesse de déplacement la plus basse
        - Slow_speed → Vitesse de déplacement la plus basse (pour slow_rate)
        - Fast_rate → nombre de secondes entre deux paquets pour une vitesse de déplacement supérieure ou égale à la vitesse de déplacement la plus élevée
        - Fast_speed → vitesse de déplacement la plus élevée (pour fast_rate)
        - Min_tx_dist → distance minimale en dessous de laquelle le smart beacon n'émettra pas de paquet.
        - Min_bcn → Temps minimum entre deux paquets du smart beacon
- Enhance_precision : Transmet des informations de position plus détaillées ainsi que l'altitude. Pour un véhicule terrestre, cette information est inutile et ne fera qu'allonger le temps de transmission et augmenter la probabilité d'erreur à la réception, il est recommandé de mettre cette option sur « false »

- Button → affecte une fonction au bouton du milieu sur le tracker

    - Tx → une pression brève émettra une trame
    - Alt_message → une pression longue affichera à l'écran le message envoyé par le beacon.

- Lora

    - Frequency_rx → fréquence d'écoute du module (inutile pour le tracker)
    - Frequency_tx → fréquence d'émission du tracker. L'IARU a défini la fréquence node vers gateway à 433,775 MHz pour la région 1.
    - Power → puissance d'émission en dBm
    - Spreading_factor → paramètre SF de la modulation LoRa. Vaut 12 par défaut, à ne pas modifier.
    - Signal_bandwidth → bande passante du signal émit. Vaut 125 000 par défaut, à ne pas modifier.
    - Coding_rate4 → paramètre CR de la modulation LoRa. Vaut 5 par défaut, à ne pas modifier.

- Ptt_output

    - Active → permet de mettre une sortie du module à 1 pendant les émissions du module, peut par exemple servir à allumer un amplificateur pendant les émissions
    - Io_pin → numéro de la broche à utiliser
    - Start_delay → délais (en ms) entre le basculement de l'état de la broche et le début de la transmission
    - End_delay → délais (en ms) entre la fin de la transmission et le basculement de l'état de la broche
    - Reverse → si mis à « true » le comportement sera inversé et la broche sera mise à 0 pendant la transmission et 1 le reste du temps.

iGate

Fichier « ./data/is-cfg.json »

- Callsign → Indicatif de l'iGate. Il est recommandé d'utiliser le SSID « -10 ».
- Network

    - DHCP → active l'attribution automatique d'une adresse IP à l'iGate. Il faut laisser « true »
    - Static → paramètres réseaux à utiliser si on n'utilise pas la configuration DHCP.
    - Hostname → hostname à utiliser sur le réseau. Par défaut ce sera le callsign, mais si « overwrite » est mis à « true » alors « name » sera utilisé.

- WiFi

    - Active → Active ou désactive le Wi-Fi. Il faut le laisser sur « true ».
    - AP → Liste des points d'accès Wi-Fi auxquels l'iGate peut se connecter. Pour chaque point d'accès supplémentaire, il faut ajouter une paire SSID et password.
        - SSID → Nom du réseau Wi-Fi auquel se connecter. Ne supporte que le Wi-Fi 2,4 GHz.
        - Password → Mot de passe du réseau Wi-Fi auquel se connecter

- Beacon
    - Message → Ce message sera affiché comme informations sur l'iGate sur les sites tels que aprs.fi. Ce message n'est pas transmis en LoRa et donc la longueur n'est pas critique, mais il faut de préférence le garder court et concis.
    - Position → C'est la position GPS où l'iGate est placé. Insérez les coordonnées GPS en degrés décimaux (ex : 1.23456 et -12.34567). Le système de coordonnées à utiliser est WGS84, les sites connus de cartes en ligne utilisent généralement ce système.
    - Use_gps → Cette option sert dans le cas très particulier où l'iGate serait itinérant (il aura donc besoin d'une puce GPS). Il faut le laisser sur « false ».
    - Timeout → temps d'attente minimum entre deux paquets dans le cas où l'iGate émet ses trames LoRa (non-recommandé). Cette valeur n'est pas utilisée en temps normal.

- Aprs_is
    - Active → active la connexion au serveur aprs-is. Il faut laisser cette option sur « true »
    - Passcode → sorte de mot de passe correspondant à l'indicatif de l'iGate. Obligatoire pour se connecter aux serveurs aprs-is. Peut être trouvé via cette page web : https://apps.magicbug.co.uk/passcode/

- Digi
    - Active → Active la fonction *digipeater* de l'iGate. Si le module est connecté à internet, il est fortement recommandé de laisser « false » pour éviter la congestion de la fréquence.
    - Beacon → Permet à l'iGate d'émettre des trames LoRa comme un tracker. La fonction est peu utile, il est aussi fortement recommandé de laisser « false ».

- Display
    - Always_on → Si « true », Garde l'écran OLED du module activé en permanence, si « false », éteint l'écran après « timeout » secondes.
    - Timeout → Nombres de secondes avant d'éteindre l'écran OLED si « always_on » est sur « false ».
    - Overwrite_pin → Soit 0, soit le numéro d'une broche. La broche sera configurée en entrée avec pull-up. La mettre à 0V activera l'écran.
    - Turn_180 → pivote le contenu affiché sur l'écran OLED de 180° (si le module est monté la tête en bas par exemple).

- ftp
    - Active → active le serveur FTP interne à l'iGate pour pouvoir modifier le fichier de configuration en Wi-Fi. Le protocole FTP n'est pas sécurisé, il est donc recommandé de laisser « false ».

    - User → Paires « name » et « password » par utilisateur autorisé à se connecter au FTP.

- MQTT

    - Active → active la publication de données via le protocole MQTT. Il est recommandé de laisser sur « false » sauf si vous savez ce que vous faites et utilisez ce protocole.

- Syslog

    - Active → Active la publication de logs via le protocole syslog. A laisser sur « false ».

• Ntp_server → adresse d'un serveur NTP sur lequel l'iGate ira chercher l'heure qu'il est. Si le réseau auquel l'iGate est connecté possède un serveur NTP fiable, l'adresse peut-être modifiée, autrement il est fortement recommandé de laisser la valeur par défaut (pool.ntp.org).

Morgan ON4MOD