

Le nano-ordinateur Raspberry Pi3 B ou B+ est un outil très performant et idéal pour les radiocommunications numériques portables ou nomades. Les applications de radiocommunication numériques en mode FT8 ou JS8 nécessitent une synchronisation horaire parfaite à l'heure UTC et ce, à la seconde près, pour établir une liaison fiable avec toutes les stations mondiales utilisant les applications WSJT-X ou JS8call. Voici un petit tutoriel qui vous montre comment connecter simplement un GPS sur le port USB de votre Raspberry Pi 3 B ou B+ afin d'avoir une synchronisation horaire parfaite et un QTH locator automatique avec JS8call.

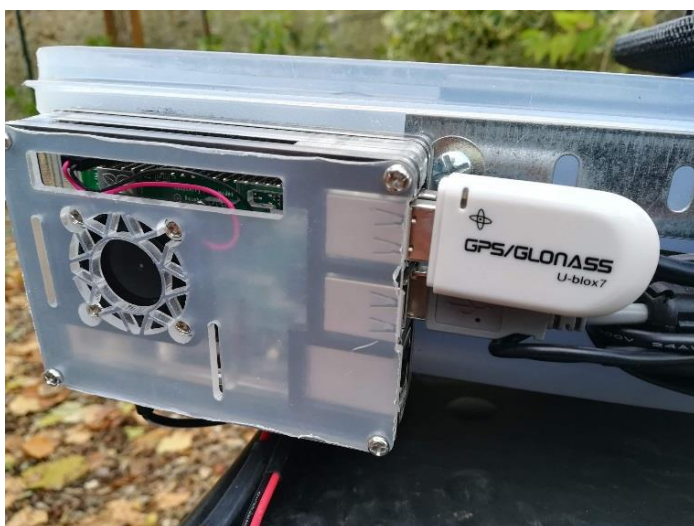
Pour ce faire, j'ai utilisé un module GPS USB VK-172 très performant et peu coûteux (prix moyen 15€ sur [Amazon.fr](https://www.amazon.fr))



Ils - VK-172 USB Récepteur GPS Dongle
Adaptateur Module Antenne Smart pour
Gmouse Glonass



Personnellement, j'utilise ce GPS USB sur ma station HF de randonnée « Backpack » équipée d'un Yaesu FT-891 et d'un Raspberry Pi3 B+ qui pilote l'interface Yaesu SCU-17 connectée au transceiver :



J'ai aussi installé un GPS USB sur ma station « Maquisard » QRP équipée d'un transceiver Xiegu X1M à 4.5W HF.



Voici la procédure d'installation du GPS (remerciements à **KM4ACK** du groupe **JS8call** pour cette procédure) utilisée pour la synchronisation automatique de l'heure UTC de votre Raspberry PI3 B ou B+ :

En mode terminal sur le RPI3, tapez la commande suivante :

- **`sudo apt-get install gpsd gpsd-clients python-gps chrony`**

```
pi@raspberrypi:~ $ sudo apt-get install gpsd gpsd-clients python-gps chrony
Reading package lists... Done
Building dependency tree
Reading state information... Done
chrony is already the newest version (3.0-4+deb9u1).
gpsd is already the newest version (3.16-4).
gpsd-clients is already the newest version (3.16-4).
python-gps is already the newest version (3.16-4).
The following packages were automatically installed and are no longer required:
  erlang-base erlang-crypto erlang-syntax-tools libboost-thread1.62.0 libqt5scintilla2-12v5
  libqt5scintilla2-110n libqt5x11extras5 libqt5-qt5-6 libscsynth1 libscpt1
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 103 not upgraded.
pi@raspberrypi:~ $
```

Puis éditez le fichier de configuration du gps en tapant la commande :

- **`sudo nano /etc/default/gpsd`**

Entrez les paramètres encadrés en rouge :

```
pi@raspberrypi: ~
File Edit Tabs Help
GNU nano 2.7.4 File: /etc/default/gpsd

# Default settings for the gpsd init script and the hotplug wrapper.

# Start the gpsd daemon automatically at boot time
START_DAEMON="true"

# Use USB hotplugging to add new USB devices automatically to the daemon
USBAUTO="true"

# Devices gpsd should collect to at boot time.
# They need to be read/writeable, either by user gpsd or the group dialout.
DEVICES="/dev/ttyACM0"

# Other options you want to pass to gpsd
GPSD_OPTIONS="-n"

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^N Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

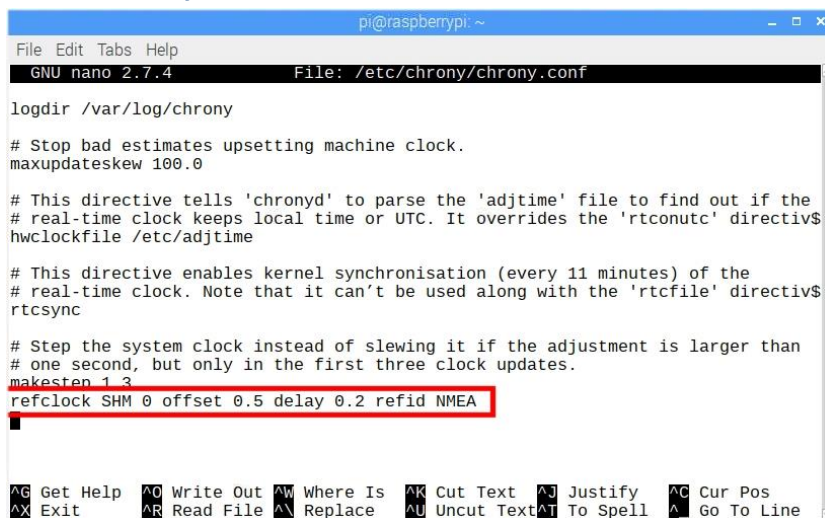
Puis tapez **Ctrl-X** puis **Y** pour sauver les données.

Tapez ensuite la commande :

- **sudo nano /etc/chrony/chrony.conf**

Entrez les paramètres encadrés en rouge à la fin du fichier de paramètres :

- **refclock SHM 0 offset 0.5 delay 0.2 refid NMEA**



```
File Edit Tabs Help
GNU nano 2.7.4 File: /etc/chrony/chrony.conf

logdir /var/log/chrony

# Stop bad estimates upsetting machine clock.
maxupdateskew 100.0

# This directive tells 'chronyd' to parse the 'adjtime' file to find out if the
# real-time clock keeps local time or UTC. It overrides the 'rtconutc' directive
hwclockfile /etc/adjtime

# This directive enables kernel synchronisation (every 11 minutes) of the
# real-time clock. Note that it can't be used along with the 'rtcfile' directive
rtcsync

# Step the system clock instead of slewing it if the adjustment is larger than
# one second, but only in the first three clock updates.
makesstep 1.3
refclock SHM 0 offset 0.5 delay 0.2 refid NMEA

Get Help Write Out Where Is Cut Text Justify Cur Pos
Exit Read File Replace Uncut Text To Spell Go To Line
```

Puis tapez **Ctrl-X** puis **Y** pour sauver les données.

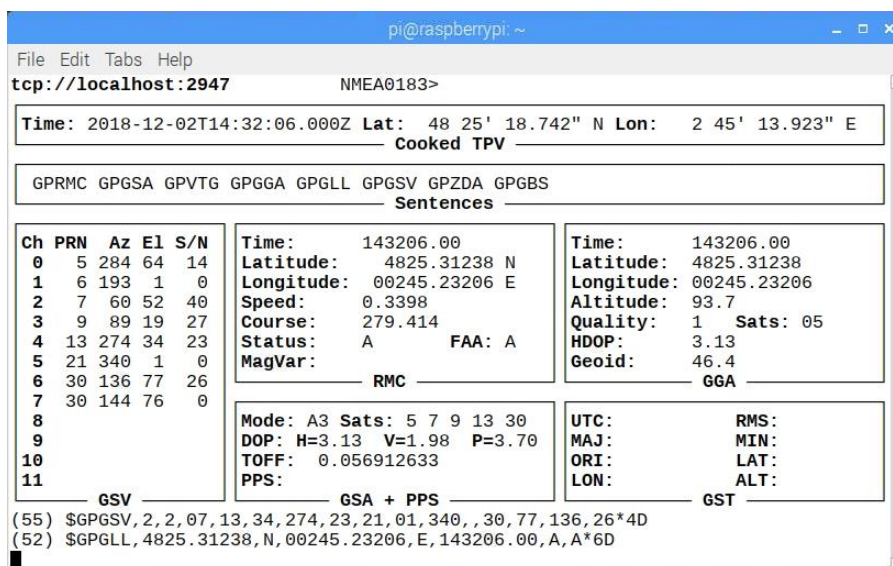
Tapez ensuite la commande **Reboot** pour redémarrer le RPI3

Après redémarrage du RPI3, vérifiez que le GPS est opérationnel en tapant

- **systemctl is-active gpsd**
- La réponse doit être **active**
- **systemctl is-active chronyd**
- La réponse doit être **active**

Pour visualiser les données en temps réel du GPS vous pouvez taper la commande :

- **gpsmon -n**



```
pi@raspberrypi: ~
File Edit Tabs Help
tcp://localhost:2947 NMEA0183>

Time: 2018-12-02T14:32:06.000Z Lat: 48 25' 18.742" N Lon: 2 45' 13.923" E
Cooked TPV

GPRMC GPGSA GPRMG GPRGA GPRGLL GPGSV GPZDA GPGGBS
Sentences

Ch PRN Az El S/N Time: 143206.00 Time: 143206.00
0 5 284 64 14 Latitude: 4825.31238 N Latitude: 4825.31238
1 6 193 1 0 Longitude: 00245.23206 E Longitude: 00245.23206
2 7 60 52 40 Speed: 0.3398 Altitude: 93.7
3 9 89 19 27 Course: 279.414 Quality: 1 Sats: 05
4 13 274 34 23 Status: A FAA: A HDOP: 3.13
5 21 340 1 0 MagVar: RMC Geoid: 46.4
6 30 136 77 26
7 30 144 76 0
8
9 Mode: A3 Sats: 5 7 9 13 30 UTC: RMS:
10 DOP: H=3.13 V=1.98 P=3.70 MAJ: MIN:
11 TOFF: 0.056912633 ORI: LAT:
GPS GSV GSA + PPS LON: ALT:
(55) $GPGSV,2,2,07,13,34,274,23,21,01,340,,30,77,136,26*4D
(52) $GPGLL,4825.31238,N,00245.23206,E,143206.00,A,A*6D
```

Si vous visualisez les données GPS en temps réel, **tout est OK et bien installé** sur votre RPI3. La LED verte du GPS doit clignoter vous indiquant l'acquisition correcte de trames GPS.

Vous n'aurez plus besoin de refaire la procédure d'installation et de paramétrage. Maintenant, à chaque mise en route du RPI3, la date et l'heure seront mises à jour automatiquement via le GPS...

Avec le GPS installé et fonctionnel vous pouvez obtenir directement le QTH locator de votre station à partir des coordonnées temps réel du GPS via un petit script :

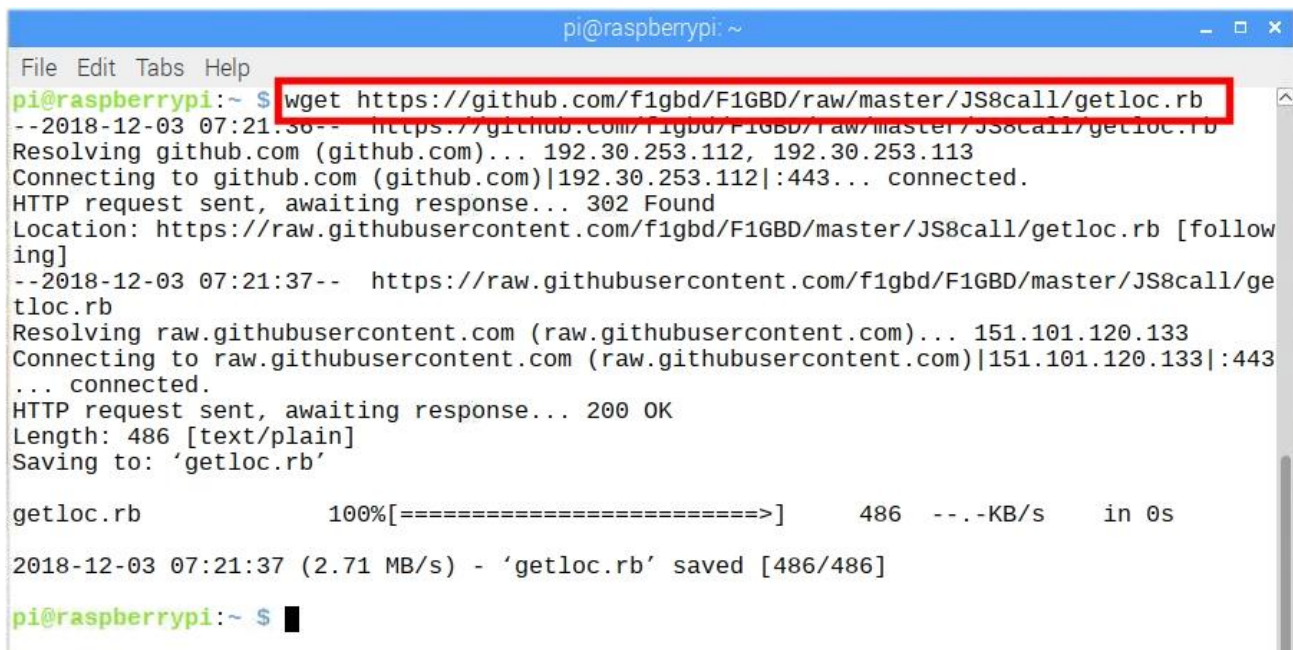
Voici la procédure d'installation de ce script :

En mode terminal sur le RPI3, tapez les commandes suivantes :

- `sudo apt-get install ruby2.3`
- `sudo gem install gpsd_client`
- `sudo gem install maidenhead`

Voici la procédure d'installation du script **getloc.rb**. Tapez la commande suivante :

- `wget https://github.com/f1gbd/F1GBD/raw/master/JS8call/getloc.rb`



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~$ wget https://github.com/f1gbd/F1GBD/raw/master/JS8call/getloc.rb  
--2018-12-03 07:21:36-- https://github.com/f1gbd/F1GBD/raw/master/JS8call/getloc.rb  
Resolving github.com (github.com)... 192.30.253.112, 192.30.253.113  
Connecting to github.com (github.com)|192.30.253.112|:443... connected.  
HTTP request sent, awaiting response... 302 Found  
Location: https://raw.githubusercontent.com/f1gbd/F1GBD/master/JS8call/getloc.rb [following]  
--2018-12-03 07:21:37-- https://raw.githubusercontent.com/f1gbd/F1GBD/master/JS8call/getloc.rb  
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.120.133  
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.120.133|:443  
... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 486 [text/plain]  
Saving to: 'getloc.rb'  
  
getloc.rb          100%[=====]          486  --.-KB/s    in 0s  
  
2018-12-03 07:21:37 (2.71 MB/s) - 'getloc.rb' saved [486/486]  
  
pi@raspberrypi:~$
```

Vous pouvez aussi copier-coller le script dans un fichier, voici le script **getloc.rb** :

```
# Gives the QTH locator with the GPS position  
# by F1GBD (ADRASEC 77) on dec,2 2018 - getloc.rb v 1.00  
# Opensource under GNU licence  
# https://github.com/f1gbd/F1GBD  
  
require 'gpsd_client'  
require 'maidenhead'  
require 'socket'  
require 'json'  
  
gpsd = GpsdClient::Gpsd.new()  
gpsd.start()  
  
# GPS ready ?  
if gpsd.started?  
  pos = gpsd.get_position  
  maid = Maidenhead.to_maidenhead(pos[:lat], pos[:lon], precision = 5)  
  puts "lat = #{pos[:lat]}, lon = #{pos[:lon]}, QTH loc = #{maid}"  
end
```




Lorsque le voyant vert du GPS clignote et que la position est acquise, il suffit de taper dans le terminal la commande :

- **ruby getloc.rb**

On obtient automatiquement le QTH locator du lieu de la station :

```

pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ ruby getloc.rb
lat = 48.421913333, lon = 2.753448833, QTH loc = JN18jk01jg
pi@raspberrypi:~ $
  
```



73' de F1GBD (Jean-Louis Naudin)

ADRASEC 77

Email : f1gbd@fnrasec.org

GitHub : <https://github.com/f1gbd/F1GBD/wiki>