

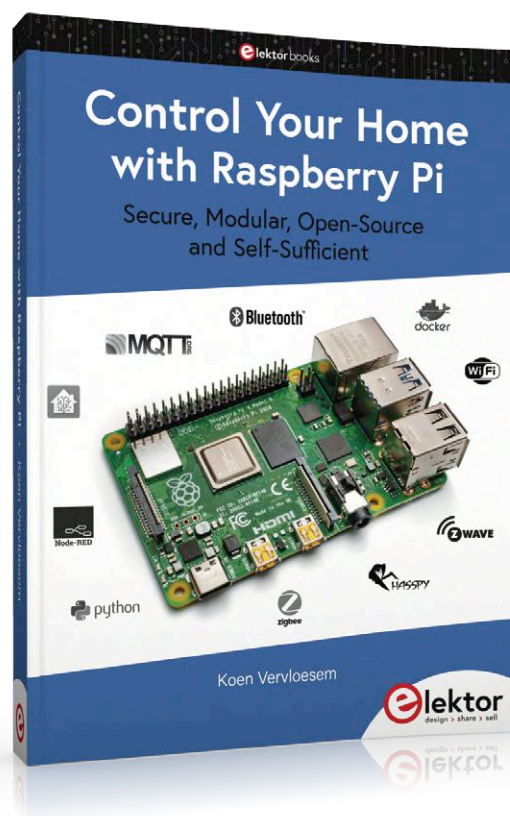
Raspberry Pi

aux commandes de la maison

RPi furète sur 433,92 MHz

Koen Vervloesem (Belgique)

Voici un chapitre extrait de l'ouvrage *Control Your Home with Raspberry Pi* de Koen Vervloesem récemment publié par Elektor. Il explique comment transformer votre RPi en un puissant récepteur et décodeur pour la plupart, sinon la totalité, des télécommandes et signaux que vous pouvez capter sur 433,92 MHz, l'une des fréquences ISM (industrielles, scientifiques et médicales) les plus connues et les plus utilisées.



En Europe, où je vis, tous les bricoleurs connaissent la fréquence de 433,92 MHz utilisée par de nombreux appareils sans fil bon marché : ouvre-portes de garage, capteurs météo et sonnettes. En outre, le matériel pour communiquer avec ces appareils est tout aussi peu cher.

La plupart de ces appareils utilisent une communication radio hélas non chiffrée, et s'ils emploient une forme de sécurité, elle est assez faible et/ou avec un algorithme propriétaire qui n'inspire pas confiance. Il y a tellement d'appareils et ils sont tellement peu chers que vous ne pouvez les ignorer. Par sécurité, je n'utilise que des capteurs 433,92 MHz : j'en ai un dans presque toutes les pièces de ma maison, ainsi qu'à l'extérieur. Je ne leur ferais pas confiance pour des tâches critiques.

Je montre ici comment lire les mesures de ces capteurs de température sans fil et comment les transmettre à votre courtier MQTT pour les injecter ensuite dans votre système domotique.

Note : je parle de 433,92 MHz, mais selon l'endroit où vous vivez, ce sera une autre fréquence, par exemple, en Amérique, 915 MHz. Achetez les bons appareils pour votre pays.

Protocoles 433,92 MHz

Les appareils qui utilisent la fréquence 433,92 MHz opèrent dans la bande de fréquence ISM sans licence. La fréquence est une chose, le protocole utilisé en est une autre. Il n'y a pas de protocole standard pour cette fréquence. Il n'y a pas de Z-Wave ou de

Zigbee. Toutefois, beaucoup de protocoles de ces appareils ont été décodés par rétro-ingénierie et vous pouvez dialoguer avec eux à condition d'avoir un émetteur-récepteur pour la bande autour de 433,92 MHz et le bon logiciel pour décoder et/ou coder le protocole.

Quelques appareils intéressants :

- > capteurs de température, d'humidité et de météo d'Alecto, Cresta, La Crosse, et Oregon Scientific ;
- > détecteurs d'ouverture de porte/fenêtre avec capteur à effet Hall ;
- > interrupteurs et variateurs d'Energenie, KlikAanKlikUit et LightwaveRF ;
- > carillons de porte de Byron et Chacon.

Vous pouvez aussi trouver chez AliGood et BangExpress des appareils encore moins chers qui prennent en charge les mêmes protocoles. Et il y a même de petits circuits imprimés tel que l'émetteur STX882 que vous pouvez connecter à un microcontrôleur ou à une carte Arduino pour créer vos propres cartes de capteurs sans fil. Je mets l'accent ici sur les premiers types d'appareils, les capteurs de température et d'humidité. Vous pouvez en trouver pour moins de 10 € (**fig. 1**), même encore moins chez AliMachin ou BangTruc. Leur portée est assez bonne : je peux recevoir les capteurs partout dans ma maison, y compris dans mon réfrigérateur et mon congélateur, et même sur ma terrasse.

Figure 1. Pour environ 5 €, vous pouvez trouver un capteur de température et d'humidité chez DANIU qui transmet ses valeurs sur 433,92 MHz et les affiche en clair sur un écran.



Besoins en matériel

Pour que votre RPi puisse recevoir les mesures d'un capteur 433,92 MHz, il vous faut un récepteur et une antenne.

Récepteur

Un type de récepteur répandu pour les projets en 433,92 MHz est une clé DVB basée sur le RTL2832 de Realtek. Oui, vous avez bien lu, DVB comme dans *Digital Video Broadcasting* (= diffusion vidéo numérique). La puce RTL2832 de beaucoup de ces clés peut faire beaucoup plus que de decoder des signaux vidéo numériques : avec le bon logiciel, ça peut faire une véritable radio logicielle (SDR) !

Donc, si vous récupérez une vieille clé DVB, vous pourriez l'utiliser pour recevoir les signaux de vos capteurs météo. Sinon, le RTL-SDR (<https://www.rtl-sdr.com>) est un bon choix (**fig. 2**). Vous trouverez des variantes de cette clé pour 25 € et en kit avec une antenne et d'autres accessoires pour ±45 € [1]. J'ai vu des clés DVB à environ 7 € chez AliAllezy qui fonctionnent très bien pour cet usage, mais ne les ai pas testées.

Antenne

Ce qu'il vous faut ensuite c'est une bonne antenne. Il y a des livres entiers sur ce vaste sujet, mais je ne suis pas spécialiste. Je sais que pour choisir l'antenne il faut connaître sa longueur qui dépend de la longueur d'onde du signal. La longueur d'onde est égale à la vitesse (en m/s) divisée par la fréquence (en Hz), et se mesure en mètres (m). Faisons le calcul pour la communication en 433,92 MHz. Dans l'air, la vitesse de l'onde est pratiquement la vitesse de la lumière. D'où la longueur d'onde :

$$299\,792\,458\text{ m/s} / 433\,920\,000\text{ Hz} = 0,69\text{ m.}$$

La longueur d'onde entière est donc de 69 cm, la demi-longueur d'onde de 34,5 cm et le quart d'onde de 17,25 cm. Ce sont les longueurs optimales théoriques d'une antenne pour recevoir des transmissions à 433,92 MHz. En pratique, divers facteurs influencent les caractéristiques de l'antenne, dont son emplacement, et une règle empirique est de soustraire 5 % à cette longueur théorique. Ceci n'est pas un livre sur les antennes, je ne m'étendrai pas davantage. De plus, pour lire des capteurs dans votre maison, la qualité de votre antenne est secondaire. Il y a des chances que cela fonctionne juste avec une « antenne 433 MHz » achetée pour quelques euros chez AliLi ou BangLaba. Si vous ne voulez pas prendre de risque, prenez une antenne incluse dans un kit avec le RTL-SDR. Le kit d'antenne officiel (**fig. 3**) comporte des antennes dipôles télescopiques extensibles de 5 cm à 1 m, ce qui couvre les longueurs d'onde optimales pour la fréquence 433,92 MHz [2].

Recevoir des valeurs de capteur avec rtl_433

Du côté logiciel, une solution courante pour lire les signaux 433,92 MHz est « rtl_433 » (https://github.com/merbanan/rtl_433), qui malgré son nom est un récepteur de données générique, principalement pour les bandes ISM 433,92 MHz, 868 MHz (SRD), 315 MHz, 345 MHz et 915 Hz.

Toutes les clés DVB à base de RTL2832 devraient fonctionner avec rtl_433, y compris la clé officielle RTL-SDR. J'utilise la clé RTL-SDR avec une antenne dipôle du kit d'antenne SDR-RTL. Connectez juste l'antenne au RTL-SDR et le RTL-SDR à un port USB de votre RPi.

Avertissement : Le RTL-SDR chauffe beaucoup en fonctionnement. Ventilez-le bien.

Figure 2. Le RTL-SDR decode de nombreux signaux sans-fil, y compris les stations météo qui émettent en 433,92 MHz.



Figure 3. Avec le trépied, le support de dipôle et les antennes télescopiques du kit RTL-SDR, vous avez tout ce qu'il faut pour recevoir les mesures de tous vos capteurs 433,92 MHz.

Installation de rtl_433toMQTT

Le programme rtl_433 est maintenu activement et contient plus de 150 décodeurs de protocole pour divers appareils qui transmettent sur 433,92 MHz. De plus, il peut envoyer et recevoir les valeurs à un courtier MQTT. Par chance, quelqu'un a créé un conteneur Docker avec rtl_433 dans ce but précis (https://github.com/bademux/rtl_433toMQTT) [3].

Créez d'abord un répertoire pour le conteneur :

```
mkdir -p /home/pi/containers/rtl433tomqtt
```

Puis il faut ajouter la définition du conteneur à votre fichier `docker-compose.yml` :

```
version: '3.7'
service:
  mosquitto:
    # mosquitto config
  rtl433tomqtt:
    image: bademux/rtl_433tomqtt:latest
    container_name: rtl433tomqtt
    restart: always
    volumes:
      - ./containers/rtl433tomqtt:/home/user/.config/
rtl_433:ro
  - /etc/localtime:/etc/localtime:ro
devices:
  - /dev/bus/usb:/dev/bus/usb
```

Entrez la configuration du conteneur Docker pour Mosquitto au lieu de `#mosquitto config`. Le conteneur doit avoir accès au bus USB pour communiquer avec le RTL-SDR. Notez aussi que le répertoire que vous avez créé est monté en tant que volume. Vous n'avez pas à créer un fichier de configuration pour l'instant. Créez d'abord une règle `udev` pour donner les bonnes autorisations au dispositif USB :

```
sudo nano /etc/udev/rules.d/20.rtl-sdr.rules
```

Saisissez la ligne suivante :

```
SUBSYSTEM=="usb", ATTRS=="0bda",
  ATTRS=="2838",OWNER="pi", MODE=="0660"
```

Enregistrez le fichier avec CTRL+O puis quittez nano avec CTRL+X. Débranchez puis rebranchez le RTL-SDR. Examinez maintenant la liste des dispositifs USB connectés :

```
lsusb
```

Recherchez dans cette liste une ligne comme celle-ci :

```
Bus 001 Device 008: ID 0bda:2838 Realtek
  Semiconductor Corp. RTL2838 DVB-T
```

Notez les numéros de bus et de dispositif : 001 et 008. Dans les fichiers de dispositifs, recherchez à présent le bus 001 :

```
ls -l /dev/bus/usb/001
```

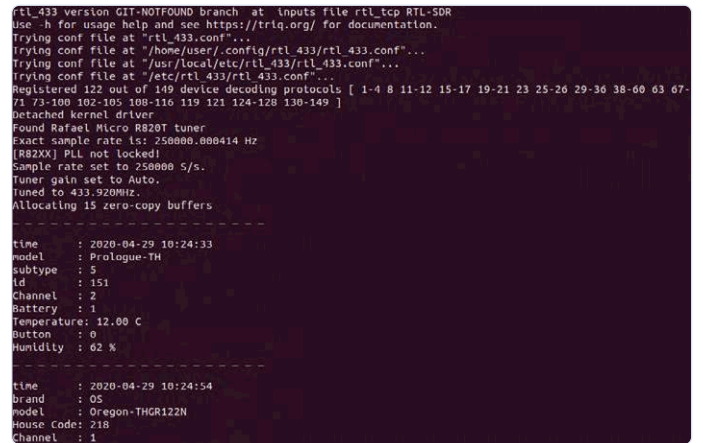


Figure 4. La commande rtl_433 trouve automatiquement le récepteur RTL-SDR et commence à afficher les mesures reçues des capteurs.

Vous devriez voir une ligne comme celle-ci :

```
crw-rw---- 1 pi root 189, 10 May 7 20:22 008
```

Elle montre que le dispositif appartient à l'utilisateur `pi`, qui dispose des droits de lecture et d'écriture.

Après cela, créez le conteneur avec :

```
docker-compose up -d
```

Après la création du conteneur, consultez son journal :

```
docker logs -f rtl433tomqtt
```

Comme la **fig.4** le montre, vous devriez voir quelques messages du programme rtl_433 qui a essayé de trouver un fichier de configuration en divers endroits, enregistré plus de 120 décodeurs de protocoles et trouvé un dispositif de réception. Vous devriez ensuite voir un message « Tuned to 433,92 MHz » et si tout va bien vous devriez voir maintenant des valeurs de capteurs arriver. Soyez patient, car beaucoup de ces capteurs n'émettent qu'une fois par minute.

Configurer rtl_433

Au début du journal, vous avez vu que rtl_433 essayait de trouver des fichiers de configuration. Il n'en a pas trouvé et a donc utilisé une configuration par défaut, ce qui était bien pour tester mais il n'a pas utilisé MQTT. Copions maintenant un exemple de fichier de configuration à un endroit parcouru par rtl_433 :

```
docker cp rtl433tomqtt:/usr/local/etc/rtl_433/
  rtl_433.example.conf /home/pi/containers/
  rtl433tomqtt/rtl_433.conf
```

Comme j'ai fait coïncider le répertoire `containers/rtl433tomqtt` avec `/home/user/.config/rtl_433` dans votre répertoire d'accueil de l'utilisateur `pi`, vous pouvez maintenant éditer ce fichier de configura-


```
rtl_433 version Git-NOTFOUND branch at inputs file rtl_tcp.Rtl-SDR
use -h for usage help and see https://triq.org/ for documentation.
Trying conf file at "/home/user/.config/rtl_433/rtl_433.conf"...
Trying conf file at "/home/user/.config/rtl_433/rtl_433.conf"...
Reading conf from "/home/user/.config/rtl_433/rtl_433.conf".
Registered 120 out of 149 device decoding protocols [ 2-3 8 11-12 15-17 19-21 23 25-26 29-36 38-60 63 67-71 73-100 102-105 108-116 119 121 124-128 130-149 ]
Detached kernel driver
Found Rafael Micro R820T tuner
Exact sample rate is: 250000.000414 Hz
(R820T) PLL not locked!
Sample rate set to 250000 S/s.
Tuner gain set to Auto.
Tuned to 433.920MHz.
Allocating 15 zero-copy buffers
{"time": "2020-04-29 11:01:58.043744", "protocol": 12, "brand": "OS", "model": "Oregon-THGR122N", "id": 218, "channel": 1, "battery_ok": 1, "temperature_C": -21.000, "humidity": 45, "mod": "ASK", "freq": 433.927, "rssi": -0.100, "snr": 5.945, "noise": -0.105}
{"time": "2020-04-29 11:02:36.885857", "protocol": 12, "brand": "OS", "model": "Oregon-THGR122N", "id": 218, "channel": 1, "battery_ok": 1, "temperature_C": -21.000, "humidity": 45, "mod": "ASK", "freq": 433.925, "rssi": -0.082, "snr": 5.968, "noise": -0.049}
{"time": "2020-04-29 11:02:37.045495", "protocol": 12, "brand": "OS", "model": "Oregon-THGR122N", "id": 218, "channel": 1, "battery_ok": 1, "temperature_C": -21.000, "humidity": 45, "mod": "ASK", "freq": 433.926, "rssi": -0.085, "snr": 6.055, "noise": -0.140}
{"time": "2020-04-29 11:03:03.976743", "protocol": 31, "model": "TFA-TwinPlus", "id": 42, "channel": 2, "battery_ok": 1, "temperature_C": -50.000, "humidity": 39, "nmc": "CHECKSUM", "mod": "ASK", "freq": 433.900, "rssi": -0.115, "snr": 5.970, "noise": -0.085}
{"time": "2020-04-29 11:03:15.807792", "protocol": 12, "brand": "OS", "model": "Oregon-THGR122N", "id": 218, "channel": 1, "battery_ok": 1, "temperature_C": -21.000, "humidity": 45, "mod": "ASK", "freq": 433.922, "rssi": -0.088, "snr": 5.807, "noise": -0.895}
{"time": "2020-04-29 11:03:16.047633", "protocol": 12, "brand": "OS", "model": "Oregon-THGR122N", "id": 218, "channel": 1, "battery_ok": 1, "temperature_C": -21.000, "humidity": 45, "mod": "ASK", "freq": 433.924, "rssi": -0.096, "snr": 6.080, "noise": -0.176}
```

Figure 5. La commande `rtl_433` peut afficher les valeurs des capteurs sous de nombreux formats, y compris JSON.

```
rtl_433/cdoedceb2dc2/events {"time": "2020-04-29 11:10:12.855670", "protocol": 12, "brand": "OS", "model": "Oregon-THGR122N", "id": 218, "channel": 1, "battery_ok": 1, "temperature_C": -21, "humidity": 45, "mod": "ASK", "freq": 433.92355, "rssi": -0.151859, "snr": 5.69722, "noise": -5.84908}
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/time 2020-04-29 11:10:12.855670
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/protocol 12
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/id 218
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/channel 1
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/battery ok 1
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/temperature_C -21
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/humidity 45
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/mod ASK
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/freq 433.92355
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/rssi -0.151859
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/snr 5.69722
rtl_433/cdoedceb2dc2/devices/Oregon-THGR122N/1/218/noise -5.84908
```

Figure 6. La commande `rtl_433` peut envoyer les valeurs reçues des capteurs vers votre courtier MQTT.

tion et redémarrer le conteneur pour l'utiliser. Par exemple, vous pouvez désactiver les protocoles dont vous n'avez pas besoin ou activer des protocoles désactivés par défaut.

Le fichier de configuration est abondamment commenté, ce qui devrait vous aider à déterminer quoi changer. De plus, vous trouverez de nombreuses informations dans la documentation en ligne de `rtl_433` (https://triq.org/rtl_433/), notamment les étapes pour prendre en charge un capteur qui ne l'est pas (encore). Après avoir modifié votre fichier de configuration, redémarrez le conteneur :

```
docker restart rtl433tomqtt
```

Si tout va bien, le journal devrait montrer que `rtl_433` s'arrête de chercher un fichier de configuration après le second fichier et le trouve :

```
Reading conf from «/home/user/.config/rtl_433/rtl_433.conf».
```

Avec le fichier de configuration en exemple, `rtl_433` montre maintenant la sortie de chaque capteur sous forme de dictionnaire JSON (fig. 5). C'est déjà un grand pas pour l'intégration de vos capteurs 433,92 MHz dans votre installation domotique, mais la dernière étape est la publication de ces valeurs auprès de votre courtier MQTT.

Publier les valeurs du capteur 433,92 MHz dans MQTT

Le programme `rtl_433` prend déjà en charge l'émission des valeurs de capteur qu'il reçoit vers un courtier MQTT. Ouvrez le fichier de configuration (`/home/pi/containers/rtl433tomqtt/rtl_433.conf`) et trouvez cette ligne :

`output json`

Vous pouvez conserver cette ligne car on peut spécifier plusieurs sorties, ou la modifier en sortie `kv` si vous préférez la sortie par défaut plus conviviale. Ajoutez la ligne suivante pour définir une sortie MQTT :

```
output mqtt://mosquitto:1883,user=home,pass=PASSWORD
```

Assurez-vous d'indiquer un nom d'utilisateur et un mot de passe corrects pour votre courtier MQTT.

Avertissement : Le programme `rtl_433` ne prend pas en charge MQTT sur TLS. Si vous exécutez le conteneur `rtl_433toMQTT` sur le même RPi que votre conteneur `mosquitto`, ce n'est pas un problème qu'ils communiquent en clair, de toutes façons ils sont sur la même machine. Si votre récepteur RTL-SDR est sur une autre machine que votre courtier MQTT (par exemple parce que vous y avez une meilleure couverture), je vous recommande d'exécuter un conteneur `mosquitto` en plus du conteneur `rtl_433toMQTT` et de le configurer comme passerelle vers votre courtier MQTT principal via une connexion chiffrée. Voyez le supplément à la fin du livre pour plus de détails.

Après redémarrage de votre conteneur, vous devriez voir un message MQTT publié sous le sujet principal `rtl_433` pour chaque valeur de capteur. Vous pouvez le visualiser avec :

```
mosquitto_sub -t 'rtl_433/#' -v
```

Par exemple, chaque fois que le capteur de température et d'humidité à l'intérieur de mon congélateur transmet un message, je vois des données comme en fig. 6.

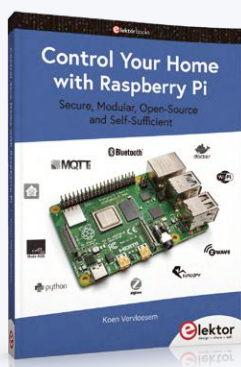
Vous pouvez maintenant modifier un peu la configuration de MQTT. Pour plus de détails, consultez les commentaires dans l'exemple de fichier de configuration. Par exemple, cette configuration modifie le sujet MQTT en quelque chose de plus court :

`output`

```
mqtt://mosquitto:1883,user=home,pass=PASSWORD,
  devices=rtl433/[model]/
  [channel]/[id]
```

Et si vous n'êtes pas intéressé par les métadonnées de bas niveau concernant la connexion radio, comme la modulation, la fréquence, le RSSI (indicateur de niveau du signal reçu), le SNR (rapport signal/bruit) et le bruit, il suffit de supprimer ou de commenter la ligne `report_meta level` dans le fichier de configuration.

Vous y êtes : toutes les mesures de vos capteurs 433,92 MHz sont envoyées à votre courtier MQTT, et vous pouvez souscrire à leurs sujets dans vos scripts Python ou vos autres logiciels de domotique.



LIVRE « CONTROL YOUR HOME WITH RASPBERRY PI »

Ce livre vous explique comment automatiser votre domicile avec un RPi. Vous apprendrez comment utiliser les divers protocoles sans fil pour la domotique, comme le Bluetooth, les ondes radio 433,92 MHz, Z-Wave et Zigbee. Bientôt, vous automatiserez votre domicile avec Python, Node-Red et Home Assistant, et vous serez même capable de parler

à votre système domotique. Tout ceci en toute sécurité, avec un système modulaire, à code source totalement ouvert, sans recourir à des services tiers. C'est vous qui serez aux commandes de votre domicile, et personne d'autre.

À la fin du livre, vous pourrez installer et configurer votre RPi comme une passerelle domotique très flexible pour les protocoles de votre choix, et joindre divers services avec MQTT pour en faire votre propre système.

- › Transformez votre RPi en une passerelle fiable pour divers protocoles domotiques.
- › Rendez votre installation domotique reproductible avec Docker Compose.
- › Sécurisez toutes vos communications réseau avec TLS.
- › Créez un système de vidéo-surveillance pour votre domicile.
- › Automatisez votre domicile avec Python, Node-Red Home Assistant et AppDaemon.
- › Accès à distance sécurisé à votre tableau de bord domotique.
- › Commande vocale entièrement hors ligne dans votre propre langue.

Le logiciel et les errata du livre sont disponibles sur GitHub : <https://bit.ly/2JPXDAG>

› Livre imprimé :

www.elektor.com/control-your-home-with-raspberry-pi

› Livre numérique :

www.elektor.com/control-your-home-with-raspberry-pi-e-book

Résumé et pour aller plus loin

Les appareils sans fil dans la bande de fréquence de 433,92 MHz sont très répandus dans la communauté du bricolage. Ils sont bon marché et facile à trouver un peu partout. Les capteurs de température et d'humidité, utiles pour surveiller votre réfrigérateur, votre congélateur ou votre serre sont particulièrement intéressants.

Vous n'avez besoin que d'une clé DVB bon marché et d'une antenne pour lire ces capteurs. Le programme rtl_433 prend en charge plus de 150 protocoles d'appareils 433,92 MHz et permet de relayer les signaux reçus en messages vers votre courtier MQTT, afin que vos autres logiciels de domotique puissent en faire usage.

Il y a encore beaucoup de sujets intéressants à peine évoqués, comme le choix et le placement optimal de l'antenne pour une meilleure couverture. C'est aussi un exercice passionnant d'essayer d'ajouter la prise en charge d'un nouveau protocole par rtl_433. Le projet

dispose d'une documentation détaillée sur la façon de capturer les signaux bruts et d'essayer de faire la rétro-ingénierie du protocole. Vous pouvez aussi essayer d'autres récepteurs, par exemple la famille des dispositifs RFXtrx, qui comprend même un émetteur-récepteur capable d'actionner les volets-roulants Somfy RTS. Il y a même une bibliothèque Python, pyRFXtrx (<https://github.com/Danielhiversen/pyRFXtrx>), pour communiquer avec vos appareils 433,92 MHz avec un émetteur-récepteur RFXtrx. Home Assistant (voir le chap. 10) utilise cette bibliothèque pour prendre en charge les appareils 433,92 MHz. Toutefois, l'émetteur-récepteur RFXtrx coûte bien plus cher qu'un RTL-SDR.

À l'autre extrémité du spectre de prix, si vous voulez expérimenter avec l'équipement le moins cher possible, l'émetteur STX882 et le récepteur SRX887 sont de bonnes bases. ◀

200534-03

Votre avis, s'il vous plaît...

Pour toute question ou commentaire relatifs à cet article, contactez l'auteur koen@vervloessem.eu

Ont contribué à cet article

Auteur : Koen Vervloessem

Maquette : Giel Dols

Illustrations : Koen Vervloessem

Traduction : Denis Lafourcade

Rédaction : Jan Buiting

LIENS

- [1] **Kit RTL-SDR avec antennes, supports et câble d'extension :** www.elektor.fr/rtl-sdr-software-defined-radio-with-dipole-antenna-kit
- [2] **Utilisation d'une antenne dipôle avec RTL-SDR :** <https://www.rtl-sdr.com/using-our-new-dipole-antenna-kit/>
- [3] La fin de la rédaction de ce livre a coïncidé avec la fin du développement actif de cette image Docker (avec plus de 100.000 téléchargements). Il n'y aura plus de nouvelles fonctions. Une autre image prendra-t-elle le relais? À suivre...
- [4] Même si vous deviez en trouver une autre, vous n'aurez à faire que quelques modifications mineures pour l'utiliser : le fichier de configuration principal de rtl_433 reste le même.