



# ma première radio définie par logiciel

réalisée en moins de 15 minutes

Clemens Valens (Elektor Labs)

Saviez-vous que vous pouvez créer un récepteur radio FM simplement en dessinant quelques blocs et lignes sur une toile virtuelle ? Si le dessin n'est pas encore votre passion, elle le deviendra après avoir lu cet article.



Figure 1. L'adaptateur SDR utilisé dans cet article est HackRF One de Great Scott Gadgets.

La radio logicielle, également connue sous le nom de SDR (*Software-Defined Radio*), consiste à moduler et démoduler les signaux radio par logiciel. Au lieu de disposer de circuits électroniques spéciaux pour ce faire, comme un mélangeur ou un démodulateur, la SDR utilise un logiciel. Bien sûr, la transformation des signaux radio analogiques en signaux numériques et vice-versa implique un certain matériel, mais c'est tout ce dont vous avez besoin. Veuillez noter que pour garder les choses simples, nous nous limiterons dans cet article à la réception de signaux radio, mais la plupart des éléments présentés sont également valables pour l'émission. Elle peut fonctionner dans les deux sens.

## La SDR peut tout faire

La radio logicielle offre de nombreux avantages par rapport à la radio analogique. Tout d'abord, elle comprend toutes les techniques de modulation auxquelles vous pouvez penser, et plus encore. Cela signifie que vous pouvez non seulement écouter la radio AM et FM, mais aussi faire du wifi ou du Bluetooth, décoder la télévision numérique ou le DAB+, faire de la radio amateur, etc. C'est possible parce qu'elle met en œuvre la technique de modulation ou de démodulation dans le logiciel qui est facile à modifier. Comme tout est fait par logiciel, elle peut également faire des choses très difficiles, voire impossibles, à réaliser avec des composants électroniques uniquement.

## Une taille unique

Un deuxième avantage de la radio logicielle est que vous n'avez besoin que d'un seul matériel pour faire tout cela.

Pas besoin de disposer d'un récepteur FM et d'un récepteur wifi et d'un récepteur de télévision numérique, etc. Elle utilise le même adaptateur pour tout.

Cet adaptateur est un convertisseur analogique-numérique (CA/N) avec une antenne qui transforme les signaux radio analogiques en signaux numériques pouvant être traités par un ordinateur. Pour la transmission, un convertisseur numérique-analogique (CN/A) est nécessaire. En réalité, cet appareil est un peu plus qu'un simple CA/N et/ou CN/A. Il comporte également un mélangeur pour amener le signal RF d'intérêt dans la gamme qu'il peut échantillonner (et l'inverse lors de la transmission), mais sa fonction de base reste la même.

Les convertisseurs SDR existent en plusieurs versions, et vous devez en choisir un qui corresponde à votre budget et à vos souhaits. Plus le convertisseur est cher, plus vous pouvez faire de choses avec. Ici, nous utilisons le HackRF One de Great Scott Gadgets [1] (figure 1). Ce n'est pas l'adaptateur le moins cher, mais il supporte la réception et l'émission, et il fonctionne jusqu'à 6 GHz. De plus, et ce n'est pas sans importance, il est supporté par de nombreuses plateformes logicielles SDR comme GnuRadio [2], SDR Sharp (SDR#) [3] et SDR++ [4]. Pour le projet décrit ci-dessous, d'autres frontaux SDR (moins chers) fonctionneront probablement aussi bien tant qu'ils sont capables de fonctionner dans la bande radio FM (87.5...108 MHz).

## Open Source et matériel

Un autre avantage de la radio logicielle est qu'elle est en grande partie open-source et open matériel, créée par une grande communauté de passionnés. Ils conçoivent et publient des schémas pour les adaptateurs et développent des outils logiciels et des algorithmes pour les applications SDR. HackRF One est également open-source et les fichiers de conception sont disponibles sur GitHub [5].

## Conditions préalables

Comme mentionné précédemment, pour ce projet SDR, j'ai utilisé HackRF One comme adaptateur avec une antenne télescopique de 88 cm. Nous utiliserons GNU Radio pour la partie logicielle. En fait, nous utiliserons GNU Radio Companion, alias GRC, l'interface graphique de GNU Radio qui est disponible pour Linux, Windows et macOS. Il y a donc de fortes chances que votre ordinateur puisse également l'exécuter. Il fonctionne même sur un Raspberry Pi.

## Programmation par glisser-déposer

GRC est un outil de programmation graphique qui vous permet de créer une application radio sans faire de réelle programmation. Vous faites simplement glisser et déposer des blocs sur un canevas que vous interconnectez avec des fils virtuels. La programmation reste limitée à la configuration des blocs. GRC a d'emblée une large bibliothèque de blocs permettant de créer toutes sortes de systèmes de modulation et de démodulation.

Nous commençons par un canevas neuf. Il n'est pas complètement vide, car il contient déjà deux blocs : un bloc *Options* et un bloc *Variable*. Un double clic gauche de la souris sur un bloc l'ouvre, ce qui vous permet de modifier ses paramètres. Le bloc *Options* permet de définir quelques options de bas niveau, et vous pouvez saisir un titre pour le canevas, voir **figure 2**. Nous allons tout laisser aux valeurs par défaut.

## Taux d'échantillonnage global

Le bloc *Variable* définit une variable globale de fréquence d'échantillonnage nommée *samp\_rate*. Elle est utilisée dans d'autres blocs. Strictement parlant, vous n'en avez pas besoin, mais c'est pratique, et donc nous la gardons. Cependant, sa valeur est trop faible pour l'échantillonnage de signaux FM, et nous la changeons donc en 10 millions. La valeur maximale pour HackRF One est de 20 millions. Si vous la fixez trop haut, votre ordinateur risque d'avoir du mal à suivre.

## Ajouter une source RF

Maintenant, nous avons besoin d'un dispositif d'entrée RF. En GRC, cela s'appelle une source. Il n'y a pas de bloc source spécial HackRF One. La section *Soapy* a une *Soapy HackRF Source*, mais cela n'a pas fonctionné pour moi. La source *osmocom*, elle, a fonctionné.

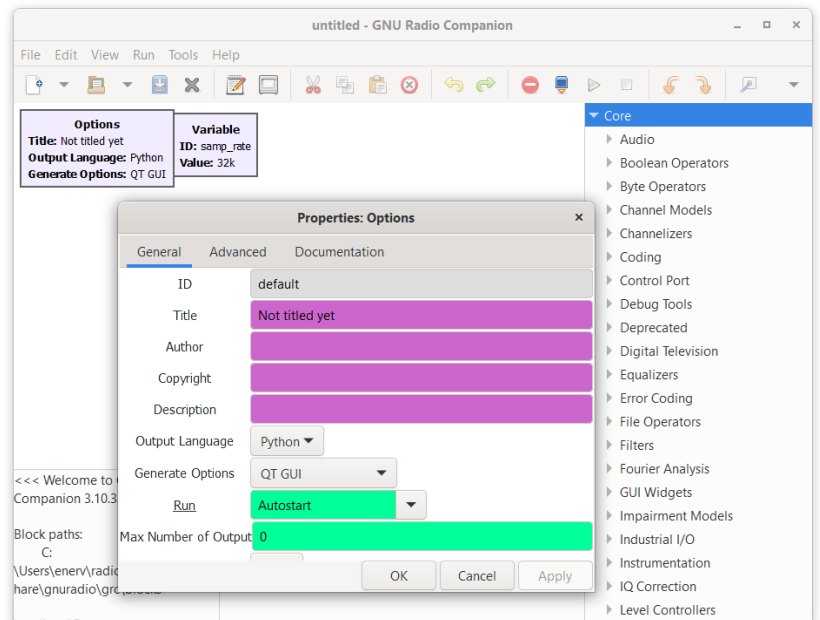
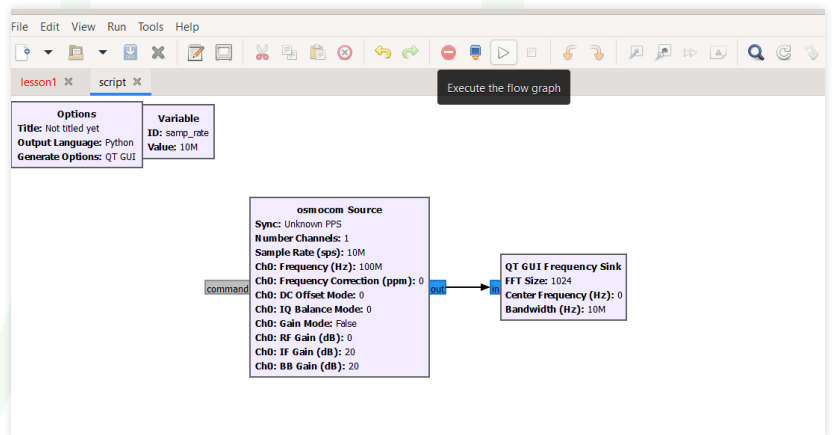


Figure 2. Un nouveau projet dans GNU Radio Companion (GRC) a d'emblée deux blocs. Un double-clic sur un bloc ouvre ses propriétés.

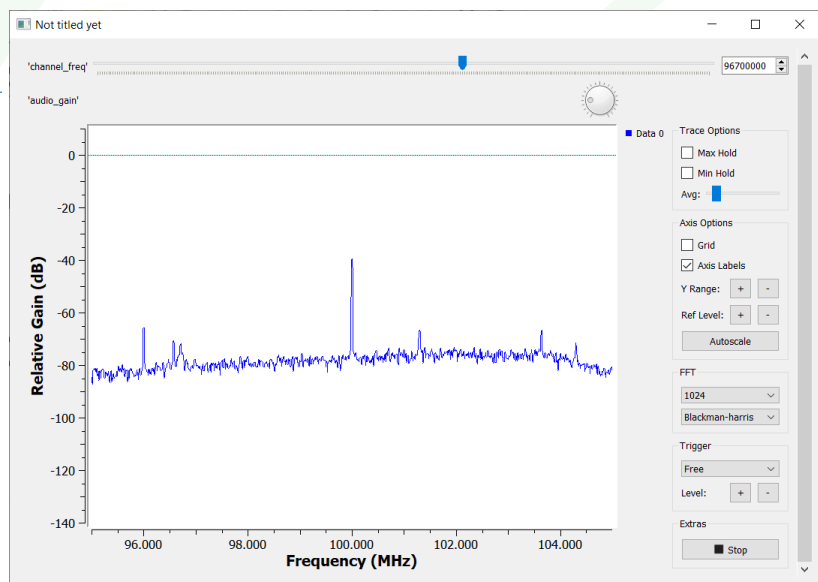


L'adaptateur OsmoSDR n'existe plus, mais le bloc supporte aussi d'autres matériels SDR, y compris HackRF One. Double-cliquer dessus ouvre ses propriétés, et nous voyons que le taux d'échantillonnage est défini par *samp\_rate*, le nom du bloc variable que GRC nous a donné. Nous n'avons qu'un seul paramètre à modifier ici, et c'est le gain RF du canal 0, qui est trop élevé pour les signaux radio FM qui ont tendance à être plutôt forts. Nous pouvons le mettre à zéro.

## Analyseur de spectre

Pour voir rapidement si les choses fonctionnent, nous pouvons ajouter un bloc analyseur de spectre au canevas et le connecter à la sortie du bloc source RF. Ce bloc s'appelle le *QT GUI Frequency Sink* (« puits de fréquence ») et se trouve dans la section *Instrumentation* de GRC. La façon dont ces blocs sont nommés peut être un peu déroutante, ce qui les rend parfois difficiles à trouver. Pour connecter l'analyseur de spectre à la source RF, il suffit de cliquer sur son entrée, puis sur la sortie de la source. L'inverse fonctionne également, de même que le dessin avec le bouton gauche de la souris enfoncé. Notez que vous ne pouvez connecter que des entrées et des sorties de la même couleur, bleue dans ce cas.

▲  
Figure 3. Cliquez sur le bouton Play pour exécuter votre premier graphique de flux.



▲  
Figure 4. L'activation du panneau de contrôle sur le QT GUI Frequency Sink donne accès à plusieurs contrôles.

### Premier essai

Vous pouvez maintenant cliquer sur le bouton **Play** (avec le petit triangle) pour exécuter le diagramme de flux (figure 3). Si vous ne l'avez pas déjà fait, vous devez enregistrer votre canevas avant de pouvoir continuer. Si tout va bien, un graphique de fréquence (le spectre) s'ouvrira. En jouant avec l'antenne, vous devriez obtenir des résultats légèrement différents, mais ils seront peut-être difficiles à percevoir. Pour améliorer un peu les choses, nous devons configurer le graphique de fréquence.

Cliquez sur le bouton **Stop** (avec le petit carré) pour arrêter le graphique de flux. Double-cliquez sur le bloc **Frequency Sink** pour ouvrir ses propriétés et définir le moyennage sur moyen. Réglez la fréquence centrale sur la même valeur que la fréquence du canal 0 de la source *osmocom*, soit 100 MHz dans notre cas. Ensuite, cliquez sur l'onglet **Config** et réglez le Panneau de configuration sur **Yes**. Fermez la fenêtre des propriétés et cliquez à nouveau sur **Play**. Maintenant, le tracé de fréquence a un panneau de contrôle, et vous pouvez cocher la case **Max Hold** (figure 4). Le fait de tripoter l'antenne devrait devenir plus visible et si c'est le cas, vous savez que vous pouvez recevoir des signaux RF.

### Plug 'n' Play !

Une chose agréable à noter est que nous n'avons rien eu à faire pour que le diagramme de flux parle à HackRF One. Il n'y a pas de pilotes à installer, pas de ports à sélectionner, c'est tout, *plug and play* !

### Les variables sont pratiques

Nous venons de définir la fréquence centrale du **Frequency Sink** à la même valeur que celle de la source RF, ce qui signifie qu'elle est nécessaire à au moins deux endroits. Par conséquent, si nous la remplaçons par une variable globale, comme GRC l'a fait pour la fréquence d'échantillonnage, nous pouvons la définir à un seul endroit.

Copiez le bloc variable de *sample rate*, ouvrez ses propriétés et changez l'ID en **center\_freq**. Définissez la valeur à 100 MHz, saisissez **center\_freq** pour la fréquence du canal 0 de la source *osmocom*, et saisissez **center\_freq** pour la fréquence centrale du **Frequency Sink**. Les deux paramètres sont maintenant connectés. Si nous modifions la valeur de la variable **center\_freq**, elle sera également modifiée dans tous les autres blocs qui l'utilisent.

### Mélanger

Pour réaliser quelque chose avec la radio logicielle, vous devez bien sûr avoir quelques connaissances sur le fonctionnement des radios. Ses algorithmes font les mêmes choses que les circuits électroniques d'une radio analogique. Ainsi, pour recevoir une station de radio FM, nous devons la syntoniser, la démoduler, la filtrer et la rendre audible.

L'accord peut être effectué à l'aide d'un mélangeur ou d'un multiplicateur de fréquence. Lorsqu'il est effectué avec des signaux en quadrature ou complexes, il permet de décaler les fréquences vers le haut et vers le bas. Dans GRC, les entrées et sorties bleues correspondent à des signaux complexes à virgule flottante qui conviennent au mélangeur.

Le but est de décaler la fréquence d'intérêt vers le centre du diagramme de fréquence, d'où elle est plus facile à extraire. Pour se syntoniser sur une station de radio, nous avons donc besoin d'un multiplicateur et d'un autre signal pour contrôler le décalage de fréquence. Un bloc **Multiplieur** est disponible dans la section **Math Operators**. Pour le signal d'accord, nous pouvons utiliser le bloc **Signal Source** de la section **Waveform Generators**.

### Syntoniser une station

Les stations de radio FM ont une fréquence connue, que nous appellerons la fréquence du canal. Comme nous voulons pouvoir la changer facilement, nous créons pour elle une variable globale que nous appelons **channel\_freq**.

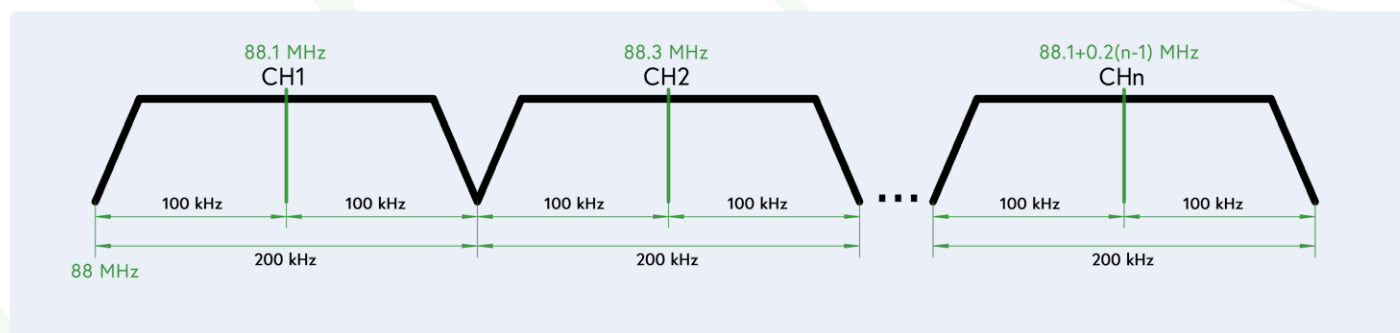
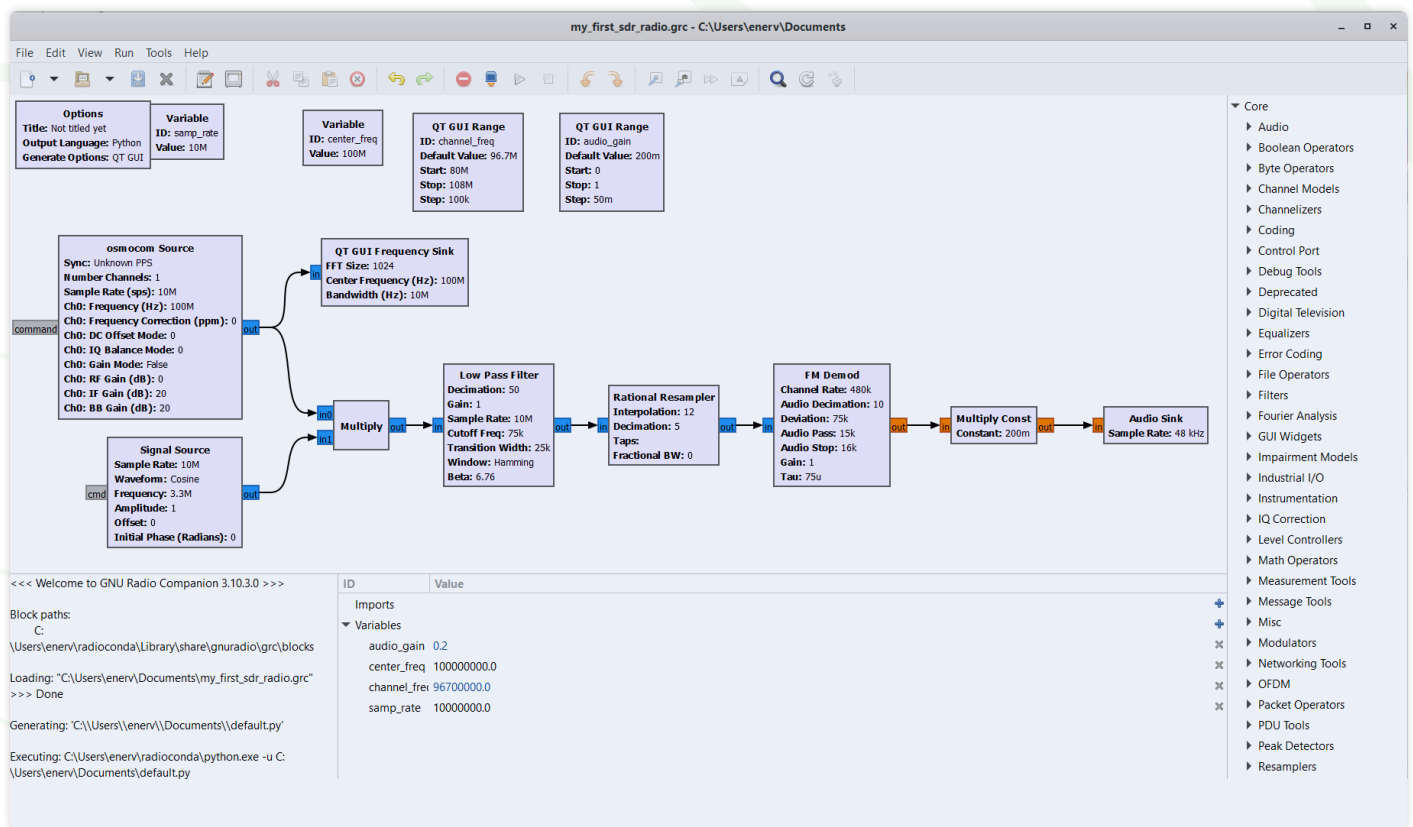


Figure 5. La bande de fréquences radio FM est divisée en canaux de 200 kHz de largeur.



Là où je vis, il y a une station de radio à 96,7 MHz, donc je mets cette valeur dans `channel_freq`. Le mélangeur doit décaler la fréquence du canal vers la fréquence centrale, donc la fréquence de la source du signal doit être la différence des deux. La différence entre 100 MHz et 96,7 MHz est de 3,3 MHz et c'est bien la fréquence indiquée dans le bloc *Signal Source*.

## Conversion vers le bas

Avec la station à la fréquence centrale, nous devons extraire son signal et l'abaisser pour qu'il puisse être démodulé. Extraire signifie filtrer et abaisser en SDR correspond à réduire ou décimer la fréquence d'échantillonnage. Le *Low Pass Filter* (filtre passe-bas) de la section *Filters* combine les deux opérations en un seul bloc. La bande FM est divisée en canaux. La largeur varie selon les pays, mais se situe généralement autour de 200 kHz. Une station de radio se trouve au milieu d'un canal, ce qui signifie qu'elle dispose d'une bande passante de 100 kHz de chaque côté (**figure 5**). La fréquence d'échantillonnage minimale requise pour un signal de 100 kHz est de 200 kHz (théorème de Nyquist-Shannon). Notre fréquence d'échantillonnage est de 10 MHz, et nous fixons donc la valeur de décimation du bloc filtre à 10 MHz / 200 kHz, soit 50. Pour extraire le signal d'intérêt, nous n'avons pas besoin d'un filtre super-puissant. Une fréquence de coupure de 75 kHz avec une largeur de transition de 25 kHz convient pour une bande passante totale de 100 kHz.

## Démodulation

Nous avons maintenant un signal en quadrature modulé en FM avec une fréquence d'échantillonnage de 200 kHz que nous voulons démoduler. Pour cela, nous pouvons utiliser le bloc *FM Demod* de la section *Modulators*.

La sortie du démodulateur est un signal audio que nous pouvons envoyer vers un récepteur audio, la carte son de l'ordinateur dans notre cas.

## Questions relatives à la fréquence d'échantillonnage

Le problème que nous rencontrons maintenant est que les fréquences d'échantillonnage sont toutes différentes. La carte son (*Audio Sink*, un « puits de son ») supporte plusieurs taux d'échantillonnage qui dépendent de votre carte son. Sur mon ordinateur, le rapport entre les taux d'échantillonnage audio possibles et le taux d'échantillonnage d'entrée de 200 kHz est une valeur non entière dans tous les cas. Or, le démodulateur FM ne peut décimer que par une valeur entière. Alors, comment faire pour que cela soit correct ?

La solution consiste à insérer un convertisseur de fréquence d'échantillonnage capable d'effectuer une conversion fractionnelle. GRC propose deux options pour cela : un rééchantillonneur fractionnel et un rééchantillonneur rationnel. Comme nos taux d'entrée et de sortie sont des valeurs entières, nous pouvons utiliser le rééchantillonneur rationnel de la section *Resamplers*. Un rééchantillonneur fractionnel peut également être utilisé, mais il consomme beaucoup plus de ressources de calcul précieuses.

Pour obtenir la meilleure qualité audio, nous fixons la fréquence d'échantillonnage audio à 48 kHz. Ensuite, si nous faisons travailler le démodulateur à, disons, dix fois cette fréquence, à 480 kHz, nous avons besoin d'une conversion de fréquence d'échantillonnage de 200 : 480. Ce rapport peut être simplifié en trouvant le plus grand diviseur commun des deux valeurs, qui est 40.  $200 : 40 = 5$ , et  $480 : 40 = 12$ , donc si nous commençons par sur-échantillonner ou interpoler par 12 et ensuite



Figure 6.  
Le diagramme de flux du récepteur radio FM terminé avec les commandes de syntonisation et de volume. Lorsque vous l'exécutez, il ressemblera à la figure 4.



par sous-échantillonner ou décimer par cinq, un taux de 200 kHz est transformé en un taux de 480 kHz. Nous devons définir ce taux dans le bloc *FM Demod*, ainsi qu'un facteur de décimation de dix, pour nous assurer que le taux de sortie correspond au taux d'entrée du récepteur audio.

### Ajouter les contrôles de l'utilisateur

Si vous exécutez ce diagramme de flux avec la fréquence du canal réglée sur une station de radio FM existante à proximité, vous devriez maintenant entendre le programme radio. C'est cool, non ?

Nous pouvons améliorer un peu notre radio en ajoutant une commande de syntonisation, afin que vous puissiez facilement syntoniser une autre station de radio. Pour cela, nous pouvons utiliser un bloc *QT GUI Range* de la section GUI Widgets. Définissez son ID sur `channel_freq`. Remplissez les paramètres en vous assurant qu'ils sont tous dans la plage voulue. Notez qu'il est inutile de définir une (très) petite valeur de pas, car la plupart des stations FM sont sur des fréquences MHz avec une seule décimale, 100 kHz conviendra probablement.

Vous pouvez choisir une forme pour le contrôle, mais *Counter + Slider* est le plus pratique parce qu'il vous permet de voir la fréquence comme une valeur numérique. De plus, vous pouvez taper une fréquence directement dans la boîte. Renommez ou supprimez l'ancien bloc variable `channel_freq`, car il n'est plus nécessaire et entre en conflit avec le *slider*. Vous pouvez maintenant régler la radio comme n'importe quelle autre radio.

### Couleurs des signaux

Une autre amélioration consiste à ajouter un contrôle du volume. Il s'agit d'un contrôle similaire à celui de l'accord, et vous pouvez le copier et ajuster ses paramètres. Renommez l'ID en `audio_gain`. Choisissez une *Const Multiply* dans la section *Math Operators* et insérez-la entre le bloc *FM Demod* et l'*Audio Sink*. Notez que les fils sont devenus rouges. C'est parce que l'entrée et la sortie du multiplicateur sont bleues au lieu d'être orange. Ils ne sont pas du même type.

Ouvrez les propriétés du multiplicateur et définissez le *Type IO* sur *float*. Saisissez également `audio_gain` dans

le champ intitulé *Constant* pour le connecter au curseur. Fermez le bloc et remarquez que les fils sont redevenus noirs, car toutes les entrées et sorties audio sont maintenant orange.

### Options de sortie

Par défaut, la sortie de GRC est un script Python avec une interface graphique telle que définie par les options de génération dans le bloc Options. Mais il existe d'autres possibilités, comme C++. Si vous supprimez tous les blocs GUI du diagramme de flux et que vous définissez *Generate Options* sur *No GUI*, le script peut être exécuté en dehors de GRC. Cela vous permet de construire des applications SDR autonomes.

Voilà, c'est tout. Exécutez le diagramme de flux et profitez de votre récepteur radio FM défini par logiciel ! Pour approfondir le monde fascinant de la radio logicielle, je vous recommande vivement de regarder l'excellent cours vidéo (en anglais) de Michael Ossmann [6] qui a inspiré cet article. ◀

220659-04 — VF : Maxime Valens

### Questions ou commentaires ?

Envoyez un courriel à l'auteur ([clemens.valens@elektor.com](mailto:clemens.valens@elektor.com)) ou contactez Elektor ([redaction@elektor.fr](mailto:redaction@elektor.fr)).



Récepteur radio FM basé sur la SDR chez Elektor.TV



### LIENS

- [1] HackRF One de Great Scott Gadgets : <https://greatscottgadgets.com/hackrf/>
- [2] GNU Radio et GRC : <https://www.gnuradio.org/>
- [3] SDR# alias SDR Sharp : <https://airspace.com/download/>
- [4] SDR++ : <https://www.sdrpp.org/>
- [5] HackRF One sur GitHub : <https://github.com/greatscottgadgets/hackrf>
- [6] Cours vidéo sur la SDR par Michael Ossmann : <https://greatscottgadgets.com/sdr/>



### Produits

- > Great Scott Gadgets HackRF One Software Defined Radio (1 MHz à 6 GHz) (SKU 18306) [www.elektor.fr/18306](http://www.elektor.fr/18306)
- > Great Scott Gadgets ANT500 antenne télescopique (75 MHz à 1 GHz) (SKU 18481) [www.elektor.fr/18481](http://www.elektor.fr/18481)
- > Elektor SDR kit de démarrage (SKU 19041) [www.elektor.fr/19041](http://www.elektor.fr/19041)

