

Yann ESTEVES  
Louis GENSOU  
Kylian OLLIVIER  
Alexandre PARNAUDEAU

04/01/2024

# SAE 103: GNU Radio

# Table des matières

<b>Présentation GNU radio:</b>	<b>3</b>
<b>Installation:</b>	
<b>Installation sous Ubuntu</b>	<b>4</b>
<b>Installation sous Windows</b>	<b>5</b>
<b>Interface utilisateur:</b>	<b>6</b>
<b>Interface affichage des signaux, spectres:</b>	<b>7</b>
<b>Les éléments à savoir:</b>	<b>8</b>
<b>Principaux blocs:</b>	<b>10</b>
<b>QT GUI Range</b>	<b>10</b>
<b>Signal Source</b>	<b>11</b>
<b>QT GUI Sink</b>	<b>11</b>
<b>Throttle</b>	<b>12</b>
<b>Filtre</b>	<b>12</b>
<b>Fonction logique</b>	<b>13</b>
<b>Prise en main de la partie python:</b>	<b>13</b>
<b>Les démonstrations:</b>	<b>16</b>
<b>1ème Démonstration (complexes)</b>	<b>16</b>
<b>2ème Démonstration (réels)</b>	<b>17</b>
<b>3ème Démonstration</b>	<b>19</b>
<b>4ème Démonstration</b>	<b>20</b>
<b>5ème Démonstration</b>	<b>21</b>
<b>Présentation des catégories:</b>	<b>23</b>
<b>Vidéo:</b>	<b>29</b>
<b>GNU Radio organisation:</b>	<b>30</b>

# Présentation GNU radio:

GNU Radio est un logiciel open source qui fournit un ensemble d'outils pour la conception, la simulation, le prototypage et le traitement du signal logiciel (SDR). Il est basé sur les principes de la liberté logicielle, permettant aux utilisateurs d'étudier, de modifier, et de distribuer le logiciel. Il est possible d'installer GNU Radio sous différents systèmes d'exploitation comme Windows, ou Ubuntu. Ce logiciel offre une flexibilité exceptionnelle car nous pouvons combiner autant de blocs que nous voulons. De plus, les utilisateurs peuvent créer des blocs ou les personnaliser en utilisant du python ou du C++. GNU Radio prend en charge une grande variété de matériel SDR, permettant aux utilisateurs de construire des systèmes compatibles. Bien sûr, ce logiciel offre une bibliothèque étendue de blocs de traitement du signal pour moduler, démoduler, filtrer, analyser. GNU Radio permet aux utilisateurs de concevoir et simuler des systèmes radio du monde réel. Il comprend une très grande bibliothèque de blocs de traitement qui peuvent être facilement combinés pour créer des systèmes de traitement du signal. GNU Radio est utilisé pour de vastes applications radio.



Ce logiciel est très utilisé car il prend en charge la radio logiciel (SDR). SDR signifie Software Defined Radio en anglais, ce qui se traduit en français par Radio Définie par Logiciel. SDR est une technologie qui permet la conception et la mise en œuvre de systèmes de communication radio utilisant principalement un logiciel informatique qui utilise des algorithmes, dans notre cas nous allons utiliser GNU Radio. Cela va nous permettre de concevoir des systèmes de communication sans fil personnalisés, comme des récepteurs FM et des émetteurs AM. De plus GNU Radio offre des outils permettant d'analyser divers signaux. Nous pouvons analyser et visualiser des signaux en temps réel, effectuer des transformations de Fourier. De plus, il est possible de concevoir des filtres, des égaliseurs, et d'autres blocs de traitement de signal pour manipuler les signaux radio selon vos besoins. GNU Radio offre un environnement de développement pour la conception et la mise en œuvre d'algorithmes de traitement du signal. Il est également possible d'émuler différents types de canaux sans fil pour tester la robustesse et les performances de vos systèmes de communication.

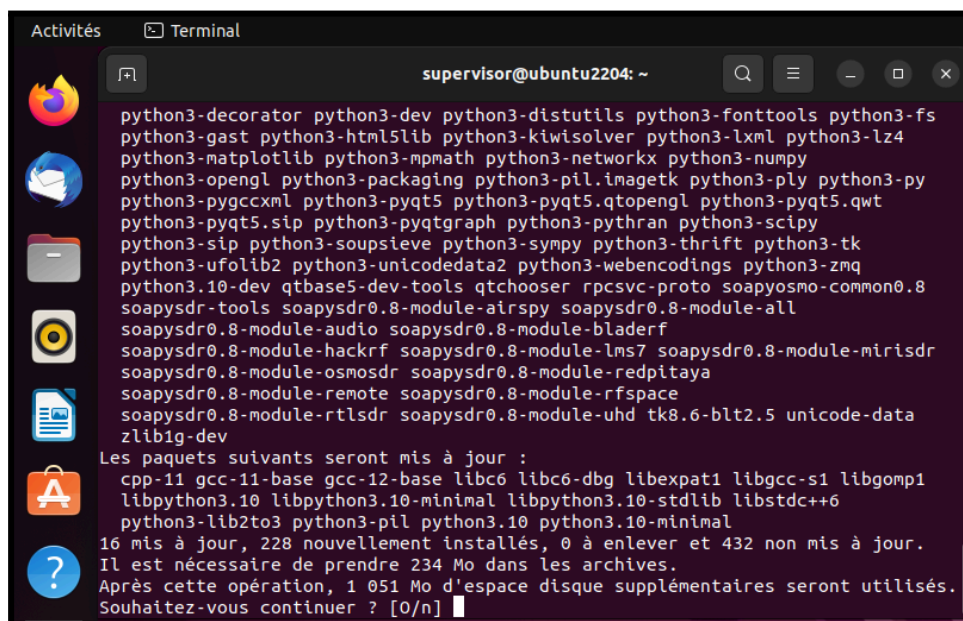
GNU Radio et GNU Radio Companion sont deux composants interdépendants d'un même écosystème utilisé pour le traitement du signal. Tout d'abord GNU Radio est la bibliothèque logicielle sous-jacente qui fournit les blocs de traitement du signal et les outils nécessaires pour concevoir des systèmes. Les applications GNU Radio sont généralement développées en utilisant Python ou C++. Les utilisateurs peuvent écrire du code directement dans le logiciel en utilisant les bibliothèques de GNU Radio. GNU Radio Companion est l'interface graphique de développement de GNU Radio. Plutôt que d'écrire du code, les utilisateurs peuvent construire visuellement des graphes de traitement du signal en plaçant des blocs sur une toile et en les connectant. GNU Radio Companion simplifie le processus de développement pour les utilisateurs qui ne sont pas nécessairement des programmeurs. Il est souvent utilisé pour les phases de prototypage rapide permettant aux débutants de se familiariser avec GNU Radio.

# Installation:

## Installation sous Ubuntu:

L'installation sous Ubuntu est la plus simple car il suffit d'entrer deux lignes de commandes. Avant d'installer n'importe quel logiciel sous linux via un terminal de commandes. Il faut d'abord mettre à jour le système d'exploitation, garantissant l'obtention de la dernière version de votre logiciel à installer. Pour cela, il faut rentrer dans le terminal, et écrire la commande: **~\$ sudo apt update**. Pour que la commande puisse s'exécuter avec succès, il faut rentrer votre mot de passe vous permettant d'accéder à votre ordinateur si vous êtes sur votre ordinateur personnel. Dans le cas où vous souhaitez installer le logiciel sur un ordinateur d'entreprise, vous devez entrer le mot de passe administrateur.

Après l'exécution de cette commande, nous pouvons installer le logiciel via la commande: **~\$ sudo apt install gnuradio**. Ensuite confirmer en entrant "O". Le logiciel utilise environ 1 Go de stockage sur votre machine.



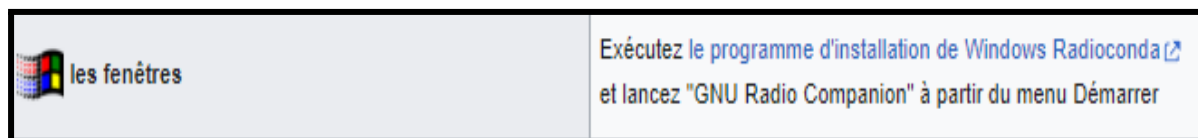
```
supervisor@ubuntu2204: ~  
python3-decorator python3-dev python3-distutils python3-fonttools python3-fs  
python3-gast python3-html5lib python3-kiwisolver python3-lxml python3-lz4  
python3-matplotlib python3-mpmath python3-networkx python3-numpy  
python3-opengl python3-packaging python3-pil.imagetk python3-ply python3-py  
python3-pygccxml python3-pyqt5 python3-pyqt5.qtopengl python3-pyqt5.qwt  
python3-pyqt5.sip python3-pyqtgraph python3-pythran python3-scipy  
python3-sip python3-soupsieve python3-sympy python3-thrift python3-tk  
python3-ufolib2 python3-unicodedata2 python3-webencodings python3-zmq  
python3.10-dev qtbase5-dev-tools qtchooser rpcsvc-proto soapysmo-common0.8  
soapysdr-tools soapysdr0.8-module-airspy soapysdr0.8-module-all  
soapysdr0.8-module-audio soapysdr0.8-module-bladerf  
soapysdr0.8-module-hackrf soapysdr0.8-module-lms7 soapysdr0.8-module-mirisdr  
soapysdr0.8-module-osmosdr soapysdr0.8-module-redpitaya  
soapysdr0.8-module-remote soapysdr0.8-module-rfspace  
soapysdr0.8-module-rtlsdr soapysdr0.8-module-uhd tk8.6-blt2.5 unicode-data  
zlib1g-dev  
Les paquets suivants seront mis à jour :  
cpp-11 gcc-11-base gcc-12-base libc6 libc6-dbg libexpat1 libgcc-s1 libgomp1  
libpython3.10 libpython3.10-minimal libpython3.10-stdlib libstdc++6  
python3-lib2to3 python3-pil python3.10 python3.10-minimal  
16 mis à jour, 228 nouvellement installés, 0 à enlever et 432 non mis à jour.  
Il est nécessaire de prendre 234 Mo dans les archives.  
Après cette opération, 1 051 Mo d'espace disque supplémentaires seront utilisés.  
Souhaitez-vous continuer ? [O/n]
```

Après confirmation, nous avons le logiciel prêt à l'emploi. Cette procédure fonctionne pour toutes les distributions de Linux, sauf sous Fedora où la commande apt correspond à dnf.

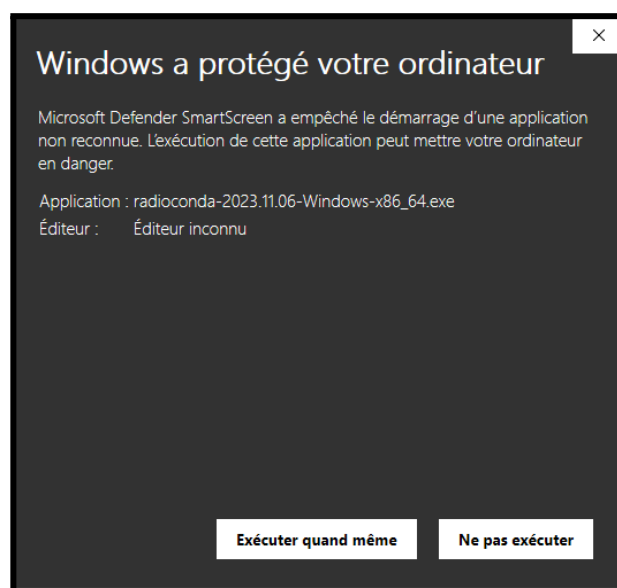
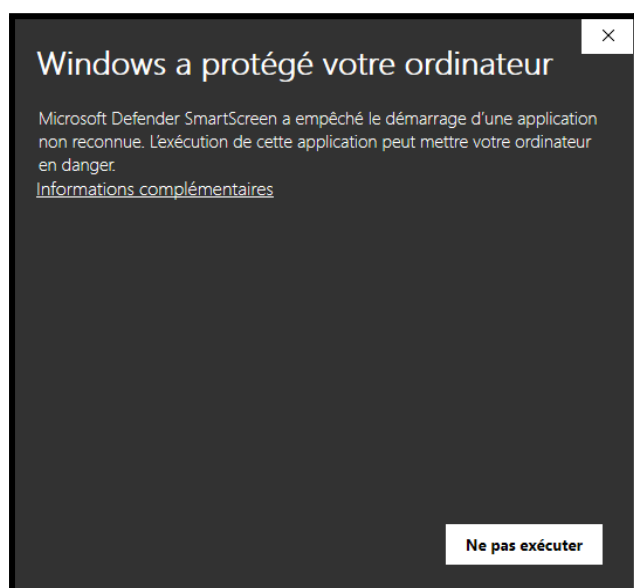
## Installation sous Windows:

Pour installer GNU Radio sous Windows, c'est plus compliqué que sur Ubuntu, il faut d'abord se rendre sur le site internet <https://www.gnuradio.org>, puis aller sur la documentation qui se situe dans le menu en haut. Ceci va nous rediriger vers une page Wikipédia de GNU Radio. Il faut ensuite cliquer sur "Installation de GNU radio" dans les menus en haut à gauche

Ensuite, nous avons un tableau indiquant comment installer GNU radio sous Windows. Pour cela, nous devons installer le programme d'installation de Windows Radioconda en cliquant sur le lien hypertexte

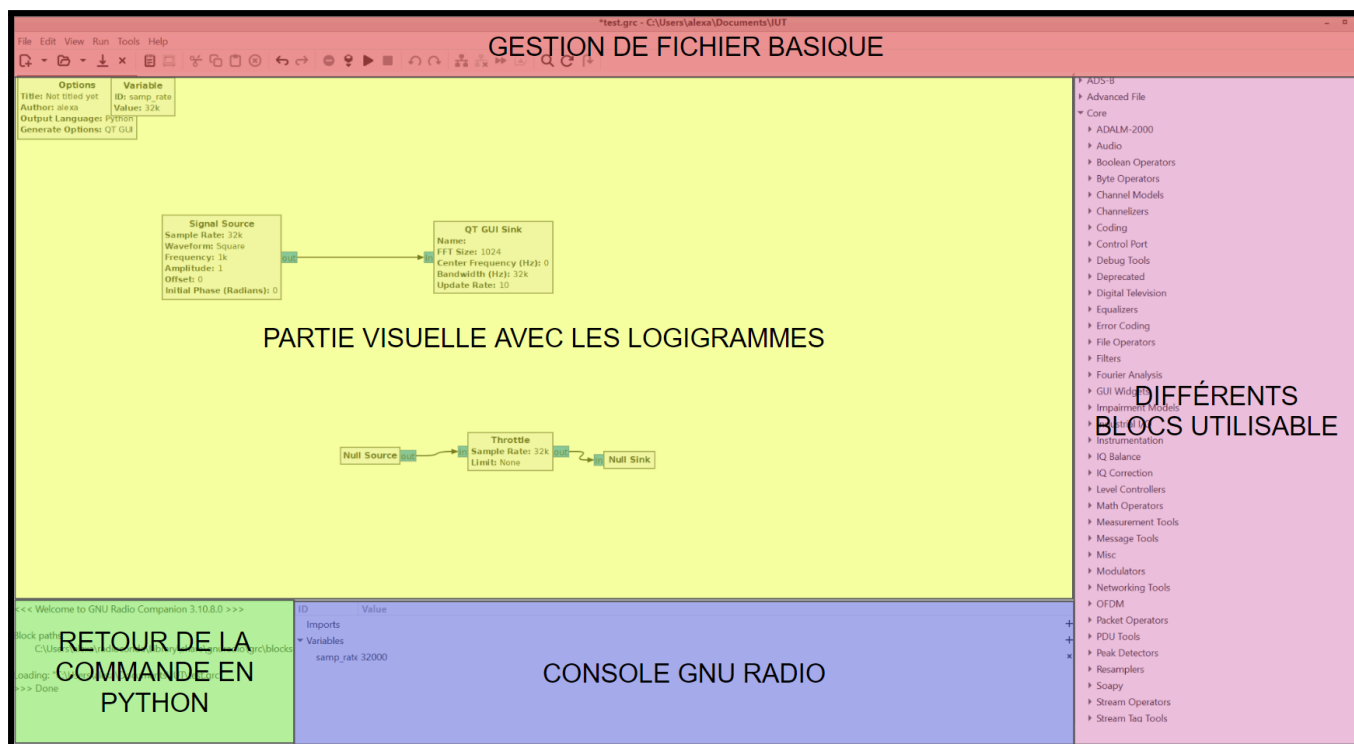


Lorsqu'on a installé ce programme, nous devons le lancer. Windows va bloquer ce programme car l'éditeur lui est inconnu. Ne vous en faites pas, ce logiciel n'est pas malveillant, vous pouvez l'exécuter en toute sécurité.



Cliquez sur informations complémentaires et cliquez sur Exécuter quand même. Dès que l'on exécute le fichier, on clique une première fois sur next, puis après on accepte les conditions d'utilisation. Ensuite on nous demande si on veut installer le logiciel seulement pour nous ou pour tous les utilisateurs, dans le cas où la machine appartient à une entreprise il nous faudra bien sûr le mot de passe administrateur pour confirmer. On nous demande l'endroit où l'on veut enregistrer le logiciel GNU Radio. L'installation risque de prendre plusieurs minutes. Lorsqu'on lance GNU Radio, nous avons l'interface graphique qui se lance et en même temps, nous avons l'invite de commande qui s'ouvre.

# Interface utilisateur:



L'interface de GNU Radio est divisée en plusieurs parties:

- **En rose**, la partie la plus à droite, est l'endroit où l'on trouve tous les blocs qui sont déjà intégrés dans GNU Radio. Nous allons très souvent nous servir de cette partie car elle est très facile d'utilisation. Les blocs sont tous triés dans des catégories et même des sous catégories.

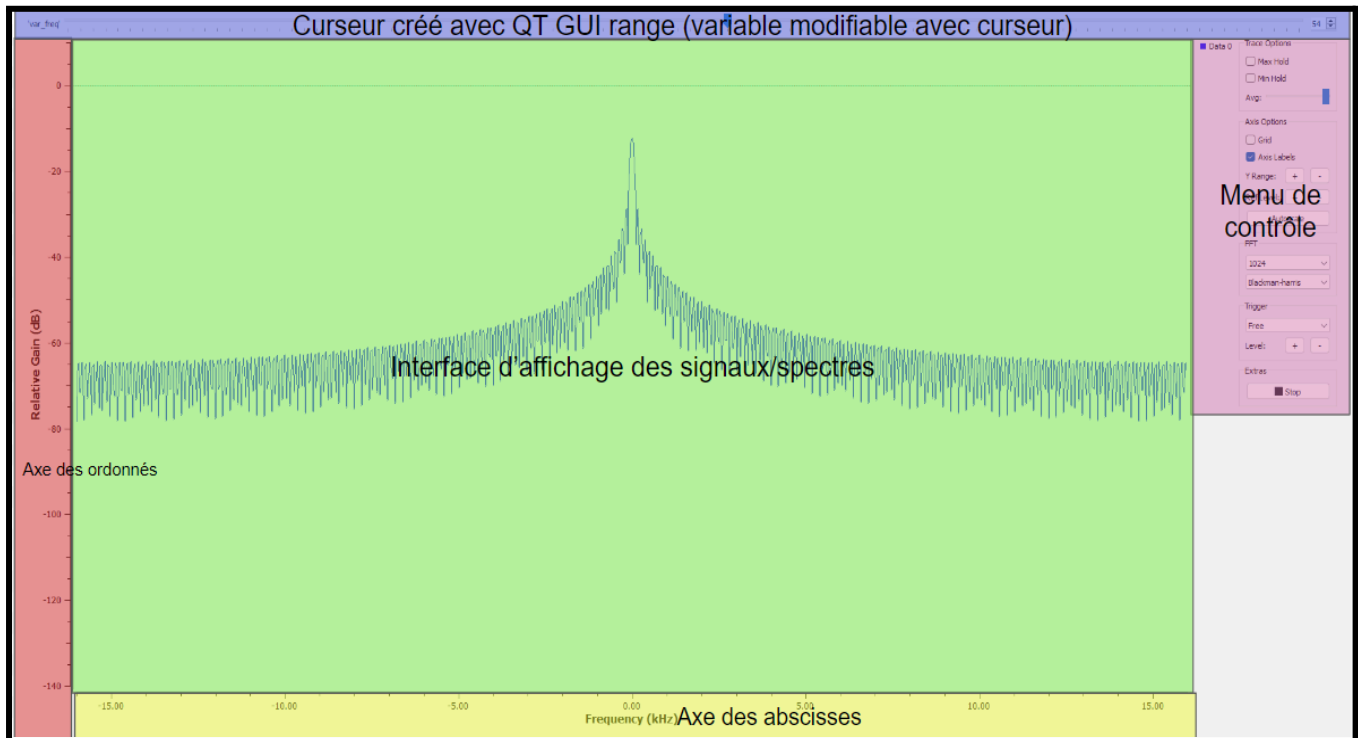
- **En jaune**, la partie du milieu va nous permettre d'afficher tout les blocs choisis dans la partie en rose. C'est à cet endroit que nous allons relier les blocs entre eux afin de réaliser des logigrammes. Si nous avons beaucoup de bloc nous pouvons dézoomer à l'aide de la molette afin d'avoir une vue globale sur les logigrammes.

- **En rouge**, la partie la plus haute correspond à la barre d'outil du projet et du logiciel. Dans le menu file (fichier) vous pouvez enregistrer le projet, ouvrir un nouveau projet, ou faire une capture d'écran du logigramme. Dans le menu edit, on peut copier, coller, couper, annuler, refaire, tourner un bloc, et accéder à ses propriétés. Dans le menu view on peut accéder aux messages d'erreur. Dans le menu Run, on peut compiler, exécuter, ou stopper le processus. Le menu Tools nous permet de convertir notre logigramme en fichier python. Enfin dans le menu help on peut avoir accès à la documentation de GNU Radio. Dans la barre d'outils, située juste au-dessus du logigramme, reprend des fonctions présentes dans les menus afin de faciliter leur accès.

- **En violet**, la partie tout en bas permet de voir toutes les variables que nous utilisons dans le projet.

- **En vert**, la partie tout en bas à gauche est un terminal de commandes. Il permet d'afficher les retours, les potentielles erreurs que comporte notre programme. Il nous indique également si le programme a pu correctement s'exécuter.

## Interface affichage des signaux, spectres:



Cette interface apparaît lorsque que l'on exécute notre logigramme dans GNU Radio. Cette dernière peut se découper en plusieurs parties qui nous donnent différentes informations:

- **En rouge**, la partie de gauche représente l'axe des ordonnées. Son affichage dépendra du bloc choisi dans la catégorie: "sink". L'échelle en ordonnée peut être modifiée manuellement dans la catégorie à droite (menu de contrôle) ou dans les paramètres du bloc "sink" selon le type de "sink" choisi.

- **En jaune**, la partie en bas représente l'axe des abscisse. Son affichage va dépendre du bloc choisi dans la catégorie: "sink". L'échelle en abscisse peut être modifiée manuellement dans la catégorie à droite (menu de contrôle) mais cela est assez rare. Souvent nous pouvons modifier l'échelle depuis les paramètres du bloc "sink" choisi.

- **En bleu**, la partie en haut est un curseur créé avec le bloc QT Range permettant de modifier la valeur d'une variable. Cette variable peut être modifiée à l'aide d'un curseur que l'on peut bouger en même temps que l'exécution du logigramme.

- **En vert**, la partie du milieu correspond à la partie principale, c'est ici que tous les spectres ou signaux sont affichés selon le type de système que vous avez réalisé. Si vous souhaitez avoir des informations supplémentaires sans avoir l'onglet de droite, en rose sur l'image vous pouvez réaliser un clique molette. Cette manipulation va vous permettre d'obtenir les mêmes informations que le menu de contrôle. Et même dans certains il est

possible que vous ayez plus d'informations que le menu de contrôle. En effet dans le cas de l'affichage d'un spectre d'un signal il est possible de modifier directement la couleur de la ligne alors que dans le menu de contrôle cela n'est pas possible. Sinon vous pouvez modifier la couleur de la ligne en allant dans les paramètres du bloc "sink" choisi, puis aller dans l'onglet Config puis modifier le paramètre: Color.

- **En rose**, la partie à droite correspond au menu de contrôle. Il permet de gérer beaucoup de choses de l'interface. Il est possible de le faire apparaître en effectuant un clic molette dans cette interface. Il est également possible de le faire apparaître en se dirigeant dans les paramètres du bloc "sink" choisi puis d'aller dans la catégorie Config puis d'activer le Contrôle Panel. Ce menu permet notamment de mettre en pause la simulation afin de mieux observer un signal par exemple. Il est également possible de choisir le mode "autoscale" afin de choisir automatiquement la bonne échelle pour un spectre par exemple. Il existe une petite astuce permettant de zoomer sur une zone précise telle qu'un signal ou un spectre. Pour cela il faut tout d'abord effectuer un clic gauche glissé sur la zone voulu. Si vous vous êtes trompé vous pouvez faire un clique droit pour retrouver le zoom initial.

## Les éléments à savoir:

Tous les blocs de GNU Radio sont structurés de la même manière. En effet, les entrées des blocs sont toujours à gauche et les sorties sont toujours à droite. En double cliquant sur un bloc nous pouvons modifier ses propriétés. Dans les propriétés des blocs nous trouverons en général 3 menus. Un premier indiquant les paramètres généraux, un deuxième indiquant les paramètres avec plus de possibilités de modification (paramètres avancés) et un troisième menus donnant une description du bloc avec un lien redirigeant vers la documentation de GNU Radio à propos de ce bloc.

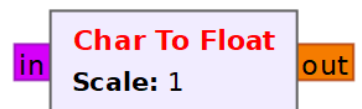
Lors de l'implantation de nouveau bloc dans notre projet nous sommes obligés dans certains cas de renseigner des informations complémentaires afin de pouvoir continuer. Les informations supplémentaires obligatoires sont toujours notées en rouge tant que vous n'avez pas renseigné le champ dans les paramètres du bloc. Par exemple, dans un bloc de génération de signal nous devons obligatoirement indiquer la fréquence du signal. De plus on peut observer qu'au niveau des entrées et des sorties des blocs il y a des cases en couleurs.

Complex Float 64
Complex Float 32
Complex Integer 64
Complex Integer 32
Complex Integer 16
Complex Integer 8
Float 64
Float 32
Integer 64
Integer 32
Integer 16
Integer 8
Bits (unpacked byte)
Async Message
Bus Connection
Wildcard

Chaque couleur correspond à un type de donnée. Voici le tableau faisant la correspondance entre la couleur et le type de données utilisées. Il faut savoir que la plupart du temps nous utilisons: float 64, float 32 et bits pour des raisons de facilité. Il est très important de toujours mettre les mêmes types de données entre blocs sinon cela ne fonctionnera pas.



Il se peut que parfois vous vouliez mettre un type de donnée différent en entrée d'un bloc. Vous pouvez utiliser des blocs convertisseurs dans GNU Radio permettant de passer d'un type de donnée à un autre type de donnée. Par exemple, on peut convertir une donnée de type char à une donnée de type float à l'aide d'un bloc. Il existe de nombreux blocs comme celui-ci. Ils se trouvent dans la catégorie Core puis dans la catégorie Type Converters.



En cliquant une fois sur un bloc, un contour bleu clair apparaîtra, ensuite nous pourrons aller dans le menu edit pour pouvoir faire tourner le bloc ou le désactiver. Il est également possible de faire tourner un bloc en maintenant le clic droit de la souris et en utilisant les flèches du clavier. Bien sûr tous les raccourcis clavier habituels fonctionnent avec GNU Radio que ce soit CTRL + X, CTRL + S, CTRL + C, CTRL + V. Vous pouvez trouver dans GNU Radio un tableau faisant un récapitulatif complet de tous les raccourcis compatibles dans l'onglet Help puis dans keys.

Une autre fonctionnalité intéressante de GNU Radio est qu'il est possible de désactiver et d'activer les blocs de nos logigrammes, ce qui les rendra "grisés". Cette manipulation va permettre de réaliser plusieurs projets sur le même fichier. De plus, il va être possible de tester différentes choses sur notre logigramme. Par exemple, si on veut voir l'impact d'un bloc sur notre projet on peut le désactiver, puis à nouveau lancer l'exécution de notre programme et observer les changements.

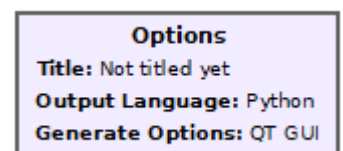


D'autres boutons sont aussi très importants, ces trois-là sont ceux qui sont le plus utilisés dans GNU Radio. Le plus à gauche permet de générer un document texte python qui contiendra tous les blocs dans leur version python. Le deuxième permet d'exécuter le logigramme et donc d'afficher le résultat de ce dernier. Le 3ème bouton lui permet de stopper la simulation en cours.



Dès le démarrage du projet, deux blocs sont déjà présents, par défaut, dans le logigramme: option et variable. Le bloc Option ne réalise pas directement une fonction de traitement de signal, mais il fournit des options de configuration pour le projet. Les paramètres du bloc option sont:

- ID: le nom du fichier
- title: titre du document
- author: auteur du document
- description: description du document
- window size: dimension de l'éditeur (largeur, hauteur) compris entre 300, et 4096
- generate options: nature du code généré
- run: détermine la méthode de lancement du processus (automatique, variable)
- max\_nouts: nombre maximum de sorties autorisées et 0 défini une valeur infini.



Pour la plupart des projets les valeurs n'auront pas besoin d'être modifiées car ces valeurs sont suffisantes.

Le bloc variable est utilisé pour introduire une variable. Elle peut être utilisée pour stocker une valeur qui peut être modifiée à tout moment, cela va nous permettre d'ajuster directement un paramètre sans avoir besoin de modifier la valeur manuellement. Cette fonctionnalité peut devenir très intéressante lorsque nous voulons utiliser plusieurs fois une même valeur. Très généralement nous nous servons du bloc pour définir une valeur d'échantillonnage que nous pouvons utiliser tout au long de nos manipulations. Si nous avons besoin de plus de variables différentes nous pouvons rajouter des blocs variables. De plus, nous pouvons avoir des variables associées à un curseur (QT GUI Range). Lorsqu'on double clique sur le bloc variable nous avons deux éléments à compléter qui sont:

<b>Variable</b>
<b>ID:</b> samp_rate
<b>Value:</b> 32k

- ID est le nom de la variable que nous allons utiliser par défaut elle s'appelle samp\_rate
- Value est la valeur que nous voulons associer à notre variable. Nous pourrons la modifier à n'importe quelle moment.

## Principaux blocs:

### QT GUI Range:

Le bloc QT GUI Range est un bloc ressemblant fortement à une variable. En effet ce bloc agit de la même manière que ces dernières, on lui attribue un nom qui peut être utilisé dans d'autres blocs. Cependant QT GUI Range possède une particularité, c'est qu'il est une variable changeante, lorsque que l'on exécute un logigramme avec QT GUI Range il crée alors une barre avec curseur qui permet de changer la valeur de la variable sur laquelle il agit.

<b>QT GUI Range</b>
<b>ID:</b> variable_qtgui_range_0
<b>Default Value:</b> 50
<b>Start:</b> 0
<b>Stop:</b> 100
<b>Step:</b> 1

Les principaux paramètres du bloc sont:

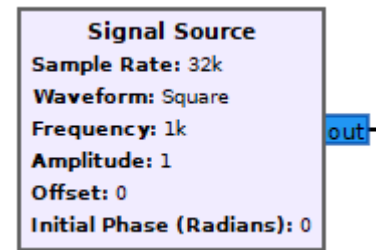
- *ID*, nom de la variable que nous allons utiliser
- *Label*, par défaut son nom est le même que l'ID mais il est possible de lui mettre un nom qui sera affiché lors du changement de la valeur à l'aide du curseur
- *Type*, correspond au type de donnée utilisée
- *Default value*, valeur initiale sans modification avec le curseur
- *Start*, valeur minimale du curseur (celle la plus à gauche de la barre)
- *Stop*, valeur maximale du curseur (celle la plus à droite de la barre)
- *Step*, lors du déplacement du curseur on modifie la variable toujours en augmentant ou en réduisant par le step (pas)
- *Widget*, est le moyen de modifier la variable (bouton rotatif, curseur, valeurs à mettre au clavier et meme peut etre des mélanges de plusieurs widget compatibles)

!\ Faire varier avec une grande plage de valeurs risque de causer des plantages de l'ordinateur.

### Signal Source:

Ce bloc Signal Source est généralement l'élément principal d'un projet. Il va permettre de générer un signal. Ce bloc se présente comme ceci. Voici les principales propriétés:

- *Sample Rate* représente le nombre d'échantillon par seconde
- *Waveform*, correspond à l'allure du signal carré, sinusoïdale
- *Frequency*, correspond à la fréquence, ici elle est de 1 kHz
- *Amplitude*, par défaut 1
- *Offset*, ici à 0
- *Phase à l'origine*

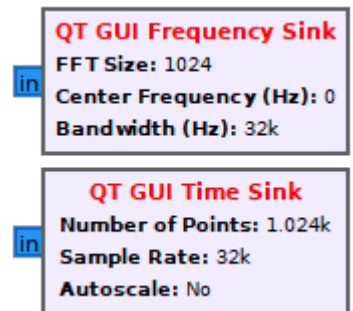


### QT GUI Sink:

Ce type de bloc est très souvent utilisé car il permet d'effectuer la représentation du signal d'entrée dans l'espace voulu tels que les fréquences, le temps... Bien sur il existe de nombreux blocs s'appelant QT GUI sink, le plus souvent nous allons utiliser les blocs:

- *QT GUI Frequency Sink*, va permettre d'effectuer la représentation du signal d'entrée dans l'espace des fréquences. Les fréquences seront sur l'axe des abscisses.
- *QT GUI Time Sink*, va permettre d'effectuer la représentation du signal d'entrée dans l'espace du temps. Le temps sera sur l'axe des abscisses.

Lorsque l'on double-clique sur ce bloc, nous devons aller dans l'onglet config pour configurer l'affichage.



Les principaux paramètres pour QT GUI Frequency Sink sont:

- *Type*, correspond au type de donnée utilisée
- *Name*, nom du bloc
- *Spectrum Width*, correspond à la largeur de bande utilisée, nous choisissons généralement half car dans ce cas là nous avons le spectre unilatéral. Si nous avons choisi full nous aurions eu le spectre bilatéral.
- *FFT size*, correspond au nombre de points utilisés dans le calcul de la transformée de Fourier
- *Window Type*, permet de choisir le type de représentation utilisé
- *Center Frequency*, correspond à la fréquence du milieu du spectre
- *Bandwidth*, correspond à la bande passante par défaut elle vaut 32 000 Hz
- *Grid*, permet l'affichage de carreau pour améliorer la lecture des valeurs du spectre
- *Autoscale*, permet d'ajuster automatiquement la fenêtre ainsi que l'échelle utilisé
- *Y min*, correspond à la valeur minimal sur l'axe des ordonnées
- *Y max*, correspond à la valeur maximal sur l'axe des ordonnées
- *Y label*, permet de donner un nom à l'axe des ordonnées
- *Y units*, permet de définir le type d'unité utilisé
- *Update Rate* correspond à la fréquence définissant le taux de mise à jour, l'actualisation

Les principaux paramètres pour QT GUI Time Sink sont:

- *Type*, correspond au type de donnée utilisée
- *Name*, nom du bloc
- *Y axis Label*, permet de donner un nom à l'ordonnée
- *Y axis Unit*, permet de définir le type d'unité utilisé
- *Sample rate*, correspond à la fréquence d'échantillonnage
- *Grid*, permet l'affichage de carreau pour améliorer la lecture des valeurs du spectre
- *Autoscale*, permet d'ajuster automatiquement la fenêtre ainsi que l'échelle

Les principaux paramètres pour QT GUI Time Sink sont:

- *Control Panel*, permet d'afficher le menu de contrôle dans la représentation du signal
- *Legend*, correspond à l'affichage de la légende
- *Axis Label* correspond à l'étiquetage des axes, permet l'affichage des unités sur l'axe des abscisses et ordonnée
- *Line Label*, permet d'appliquer les paramètres décrits ci-dessous au signal voulu. Par exemple signal1 peut avoir une largeur de 10 et une couleur bleu
- *Line Width*, permet de modifier l'épaisseur de trait
- *Line Color*, permet de modifier la couleur du trait

### Throttle:

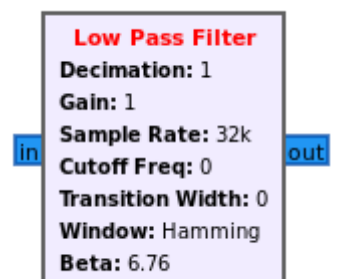
Throttle permet d'éviter que la machine soit instable. Ce bloc se présente comme ceci. Le sample Rate correspond à la fréquence d'échantillonnage. Plus la valeur est élevée et plus nous aurons de valeur à traiter. On peut donner une limite de vitesse, par défaut il n'y a aucune limite. Le but de ce bloc est de limiter la quantité d'information à traiter pour l'ordinateur. Cela va permettre de réduire la quantité de travail.



### Filtre:

Ce bloc nous permet de couper certaines fréquences. Les principaux filtres sont: passe bas, passe haut, passe bande, et coupe bande. Les paramètres des blocs sont tous quasiment identique dans le cas d'un filtre passe bas:

- *FIR Type* nous permet de choisir si on souhaite avoir un filtre de décimation ou un filtre d'interpolation. Ensuite selon le type de valeur que nous utilisons nous avons le choix entre complexe ou float.
- *Decimation*, permet de réduire la fréquence d'échantillonnage, par défaut elle vaut 1
- *Gain*, par défaut 1
- *Sample Rate* correspond à la fréquence d'échantillonnage que nous avons défini précédemment dans le bloc variable.
- *Cutoff Freq*, correspond à la fréquence de coupure
- *Transition Width*, correspond à la largeur de transition entre la bande d'arrêt et la bande passante
- *Window*, permet de choisir le type de représentation utilisé
- *Beta*, s'applique uniquement à la fenêtre Kaiser.



### Fonction logique:

Il existe de nombreux blocs permettant de réaliser des fonctions logiques s'appliquant sur des signaux. En effet, il est possible d'ajouter deux signaux entre eux afin d'obtenir en sortie un seul signal. Le bloc le plus souvent utilisé est le bloc multiply car il permet de réaliser de la modulation. Il va permettre de multiplier deux signaux entre eux, afin d'obtenir en sortie un seul signal. Les principaux paramètres modifiables sont:

- *IO type*: correspond aux types de variables utilisées en entrée et en sortie
- *Num inputs*: correspond aux nombres d'entrées que nous voulons.



## Prise en main de la partie python:

Comme nous venons de le voir, l'interface graphique de GNU Radio fonctionne avec des blocs de commandes permettant de faciliter la manipulation du logiciel. Si nous avons une erreur lors de l'exécution du projet, le terminal nous retourne l'erreur avec la ligne correspondante. Ainsi il est possible de modifier l'erreur survenue dans l'interface graphique de GNU Radio. Mais il est plus simple de modifier le fichier python afin de résoudre le problème.

Lorsqu'on crée un nouveau projet, GNU Radio nous demande d'effectuer une sauvegarde. Lors de cette sauvegarde deux fichiers seront enregistrés:

- un fichier python étant la version codée du logigramme (fichier.py)
- un fichier sauvegardant le logigramme (fichier.grc)

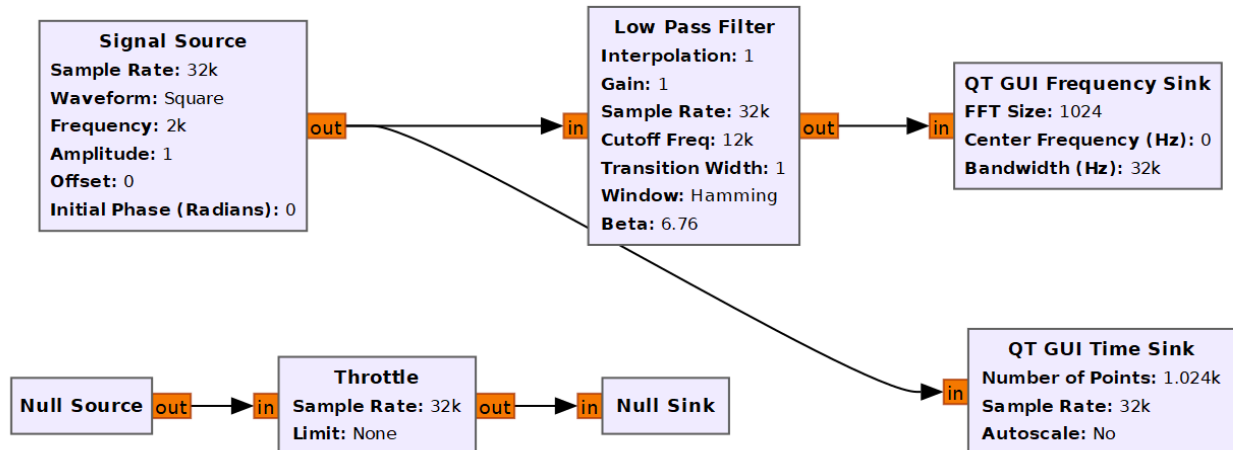
Dans notre explorateur de fichier, on pourra donc trouver le fichier python et le fichier correspondant au logigramme associé. La création d'un fichier python est très pratique si on souhaite modifier le fichier dans les moindres détails, ou même l'exécuter sans avoir besoin d'installer GNU Radio. En effet, il faut simplement exécuter le programme (python) en ayant les bibliothèques d'installées. Cependant il faut faire attention car un fichier python modifié (avec des ajouts de blocs ou autre éléments) ne modifiera pas votre logigramme automatiquement et pourrait causer des dysfonctionnements. Si vous souhaitez modifier le programme python afin de rajouter des informations supplémentaires ou simplement utiliser les fonctionnalités de GNU Radio sans avoir besoin de l'installer. Il faut ajouter des bibliothèques permettant d'importer les blocs que vous souhaitez utiliser.

Il peut arriver que le fichier python ne se génère pas automatiquement. Il est possible d'enregistrer le logigramme sous la forme d'un programme python en cliquant sur cette image ci-joint. Cette fonctionnalité se situe dans le menu en haut de la fenêtre d'édition.



### Exemple concret:

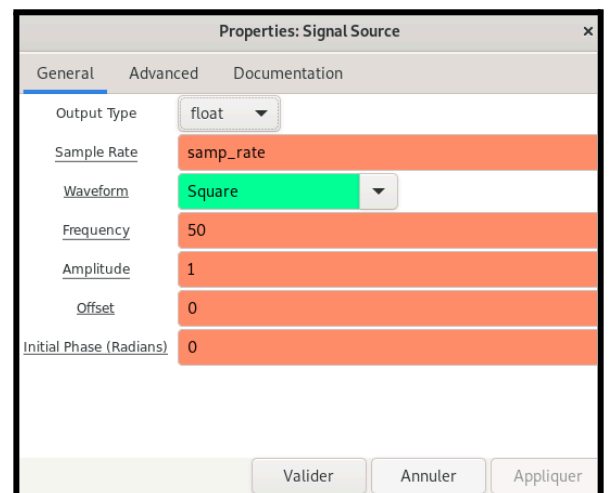
Nous allons créer un filtre passe bas ([démonstration n°3](#)) puis ensuite nous allons enregistrer le fichier contenant le logigramme (fichier.grc). Normalement le fichier python (fichier.py) se créera automatiquement.



Nous allons comparer le logigramme et le programme python afin de savoir si le programme python retranscrit bien les informations contenues des logigrammes. Voici les paramètres du bloc signal source dans l'interface graphique de GNU Radio. Nous pouvons paramétrer ce bloc via interface graphique ou via le programme python.


```
self.analog_sig_source_x_0 = analog.sig_source_f(samp_rate, analog.GR_SQR_WAVE, 50, 1, 0, 0)
```

On retrouve la configuration du signal source dans le fichier python. On peut aisément reconnaître les paramètres définis comme le sample rate, l'amplitude, la fréquence ou encore la forme du signal. En effet on remarque bien que les informations renseignées dans l'interface de GNU Radio correspondent bien aux informations vues dans le fichier python. Par exemple, nous avons défini une fréquence de 50Hz dans le bloc Signal Source sous GNU Radio et lorsque nous ouvrons le fichier python nous observons bien que la fréquence du signal généré par ce bloc est bel et bien de 50Hz.

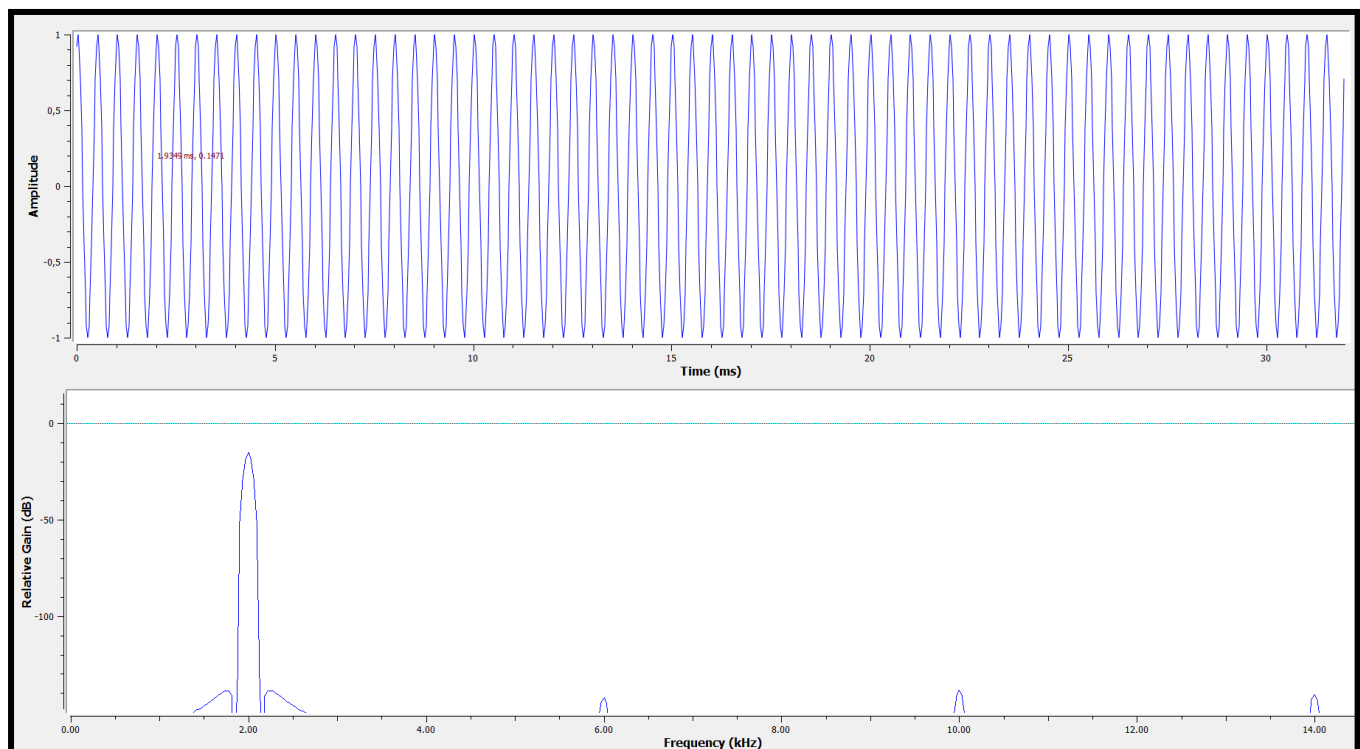


Le programme python peut être exécuté via un terminal si vous utilisez une distribution Ubuntu. En effet nous ne sommes pas obligés de passer dans l'invite de commande qui se situe dans l'interface graphique de GNU Radio. Dans un premier temps, il faut se positionner dans le répertoire du fichier.py pour cela nous allons utiliser la commande: **~\$ cd nom\_du\_répertoire**. Ensuite nous devons exécuter la commande: **~\$ python3 nom\_du\_fichier.py**. Cette commande permet d'exécuter le programme contenu dans le fichier.py avec la version 3 de python installé sur l'ordinateur.

Il est possible d'exécuter notre projet de deux manières différentes en utilisant:

- le fichier.py. Nous allons effectuer les commandes ci-dessus permettant d'exécuter le fichier.py. Puis ensuite nous allons voir la fenêtre apparaître comme si nous étions dans GNU Radio avec la même interface graphique.
- le fichier.grc. Nous allons simplement cliquer sur ce logo: . Ainsi la fenêtre ci-dessous apparaîtra. Mais il faut faire attention avec cette méthode car si vous avez modifié le programme python vous ne pourrez pas ouvrir ce fichier en .grc. En effet, le fichier .grc actualise le fichier en .py mais l'inverse n'est pas possible tout simplement parce que le fichier en .py est créé et mis à jour à partir du fichier en .grc lors de l'exécution dans GNU Radio.

Chaque bloc présent dans l'interface GNU Radio se retrouve sous forme de ligne de code en python. Généralement les blocs dans python permettent une meilleure compréhension du programme. Il est également possible avec python de créer des blocs que l'on pourra soit utiliser dans le programme python ou alors l'importer dans l'interface graphique de GNU Radio afin de l'utiliser comme si c'était un bloc présent par défaut. Finalement, nous pouvons remarquer que python permet de rendre GNU Radio encore plus complet qu'il ne l'ait déjà.

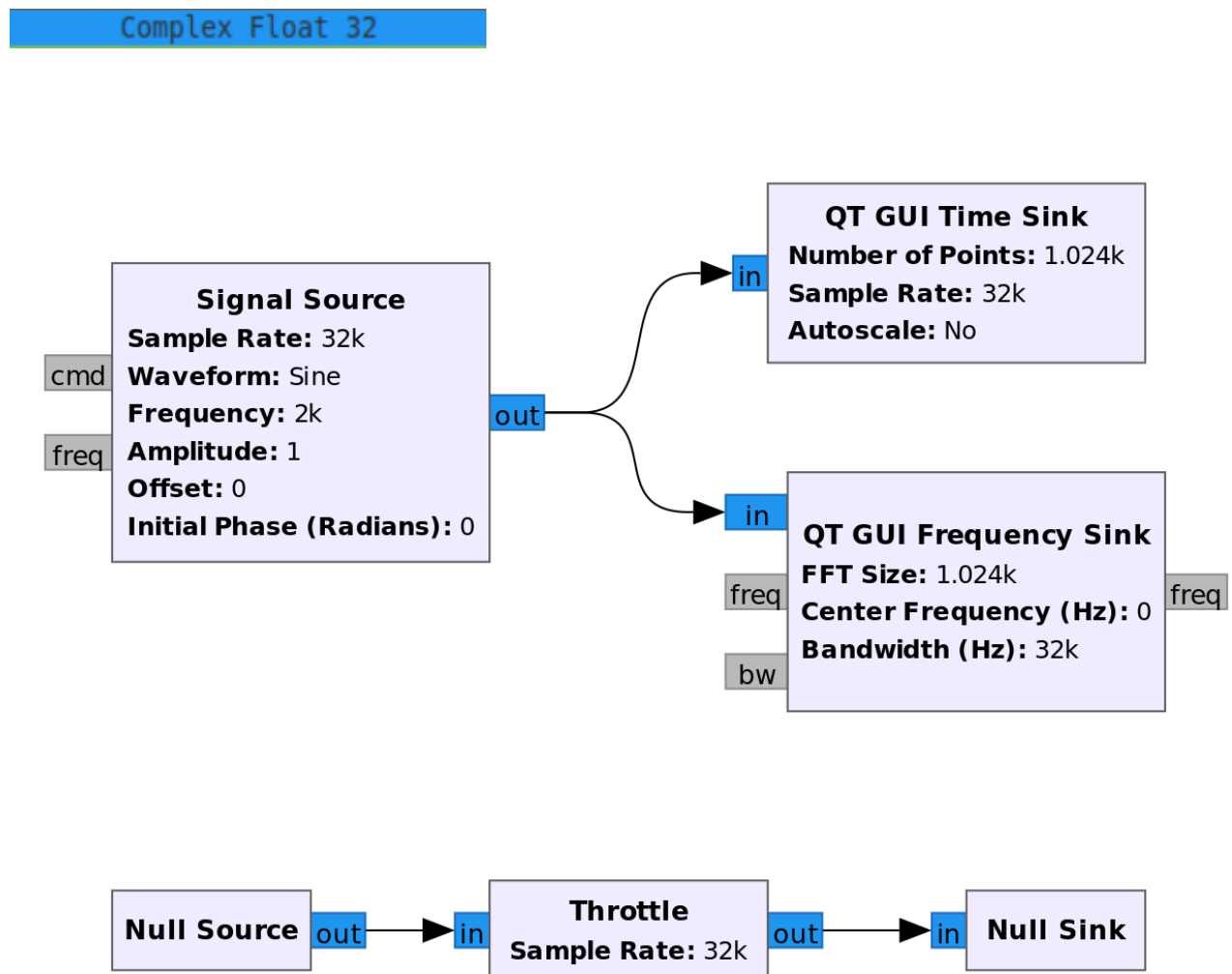


# Les démonstrations:

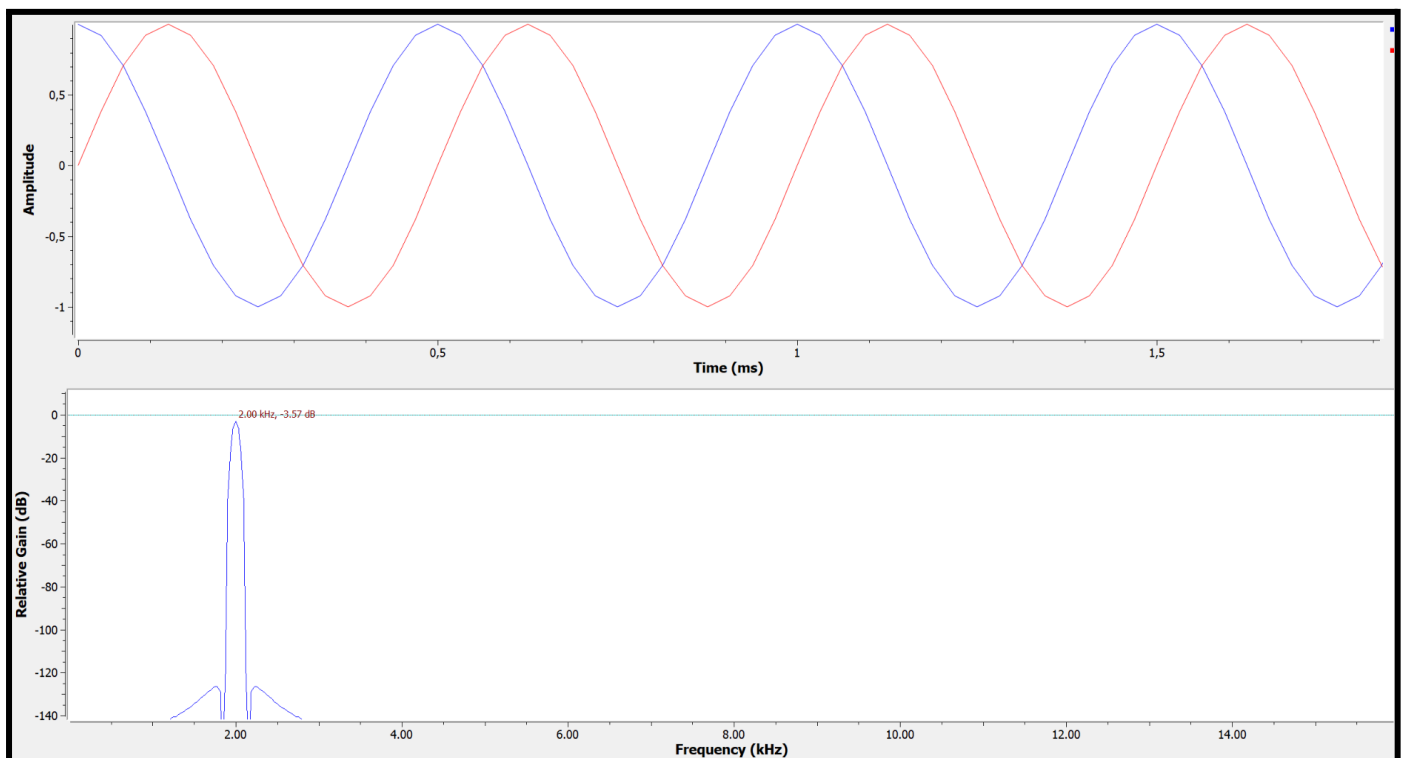
Les démonstrations ci-dessous ont été reproduites en [vidéo](#) pour ceux qui préfèrent ce type de format. La vidéo facilite la visualisation des démonstrations.

## 1<sup>ère</sup> Démonstration (complexes):

Cette démonstration permet d'afficher le spectre du signal dans le domaine des fréquences ainsi que dans le domaine du temps à l'aide des blocs QT GUI Time Sink et QT GUI Frequency Sink. En effet grâce au bloc Signal source nous pouvons créer un signal complexe en changeant le type de sortie du bloc. La couleur bleue correspond à un signal complexe. De plus, ce signal est généré à une fréquence de 2kHz, et a une allure sinusoïdale.



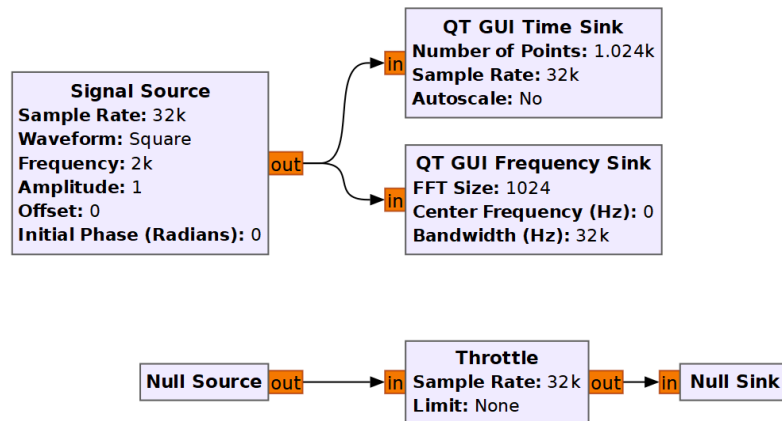




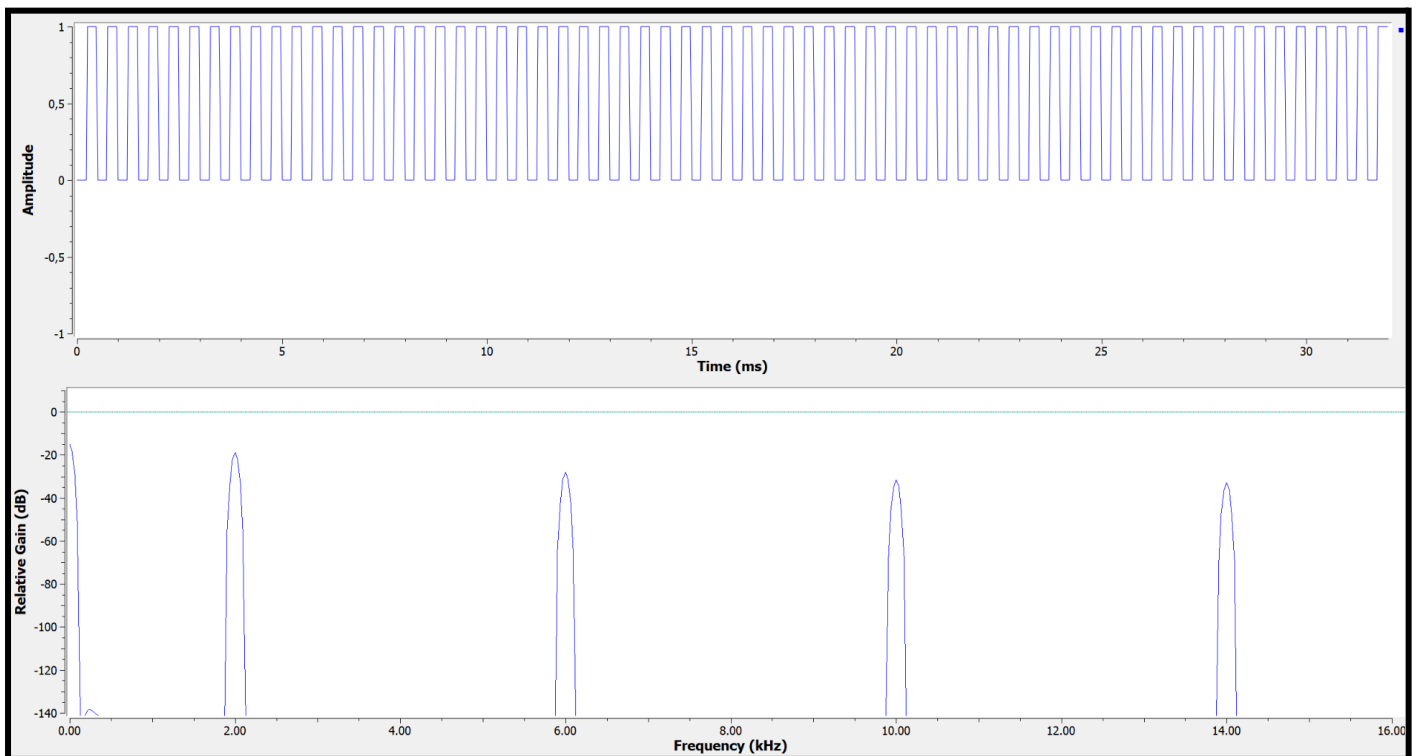
Nous pouvons remarquer que le signal complexe est composé de deux sinusoïdales, une pour les réels et une autre pour les imaginaires. Nous remarquons bien qu'il y a une raie dans le spectre. En effet ce pic correspond à  $1f$  soit 2kHz. Nous savons bien que le spectre d'une sinusoïdale possède seulement un pic à  $1f$ .

## 2<sup>ème</sup> Démonstration (réels):

Le but de cette démonstration est de tester et de comprendre le fonctionnement des blocs avec un logigramme simple. Ci dessous, on génère un signal carré de 50 Hz à l'aide d'un bloc nommé "signal source". Ensuite on récupère ce signal pour l'afficher avec l'interface graphique des blocs QT. Le bloc en haut à droite permet d'afficher le signal dans l'espace des fréquences tandis que celui du bas permet d'afficher le signal en fonction du temps. On observe aussi l'utilisation de 3 blocs en dehors du programme principal, "source null" qui est considéré par GNU Radio comme n'importe quelle entrée. De plus, le bloc Throttle permet de réduire la cadence d'envoi du signal source. Ce bloc est nécessaire car il évite à la machine de faire trop de calculs. Sans cette manipulation la machine risquerait de cesser de fonctionner. Enfin le bloc "sink null" à le même effet que le bloc "source null" mais pour les sorties. n'importe quelle sortie. Donc l'association de ces trois blocs permet ainsi à n'importe quel logigramme permettant de limiter la quantité d'information à traiter pour l'ordinateur.

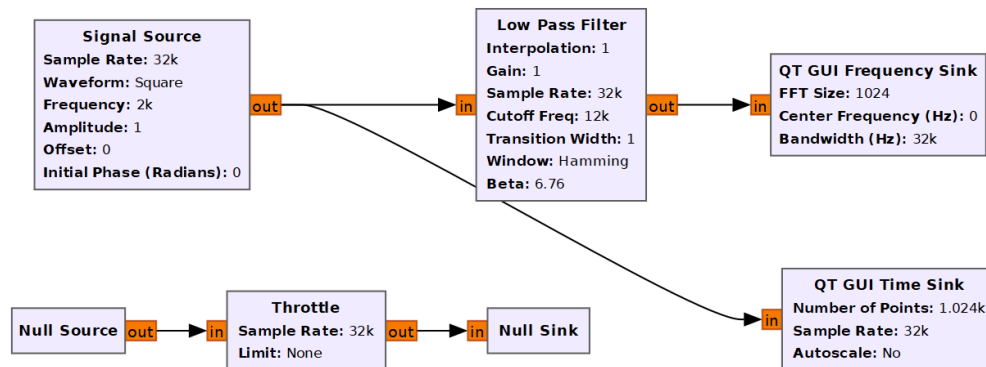


Ensuite on enregistre le projet puis on exécute le logigramme. Ci-dessous, on observe un signal carré. Le bloc QT GUI Time sink permet d'afficher ce signal dans l'espace du temps et QT GUI Frequency Sink permet d'afficher le spectre dans l'espace des fréquences. On observe bien que notre signal produit à une fréquence de 2000Hz. Si on compte le nombre de période pour 5 millisecondes alors le résultat est 10. Alors on cherche ensuite à calculer le nombre de milliseconde correspondant à une période, soit  $\frac{5}{10} ms = 0.5ms$  soit  $0.5 \cdot 10^{-3} s$  pour une période. Donc on sait que maintenant une période de ce signal dure 0.5 milliseconde alors la fréquence est:  $f = \frac{1}{T} = \frac{1}{0.5 \cdot 10^{-3}} = 2\,000\, Hz$ . De plus, on observe bien les raies sur le spectre à 1f, 3f, 5f, 7f soit 2kHz, 6kHz, 10kHz, 14kHz.

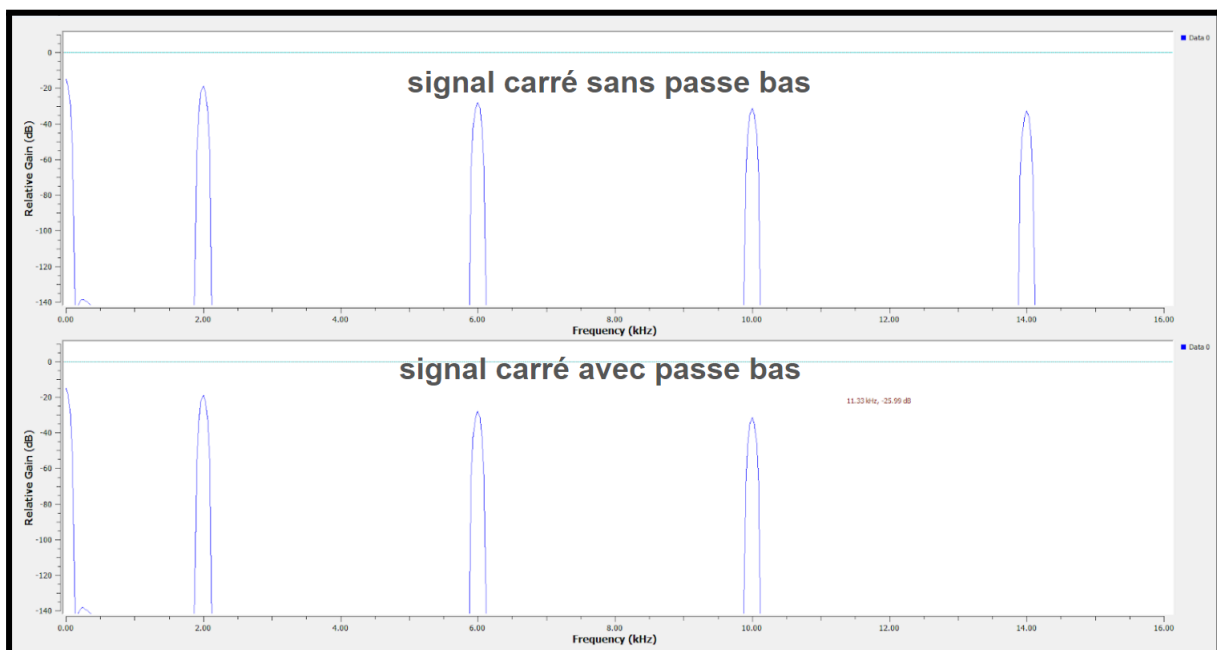


### 3<sup>ème</sup> Démonstration:

Dans cette démonstration on va essayer d'utiliser un filtre passe bas afin d'observer le spectre du signal carré. Nous avons utilisé le même signal généré dans la démonstration précédente. On applique à ce signal un filtre passe bas comme paramètres, 1 en gain, 32000 Hz en fréquence d'échantillonnage, et une fréquence de coupure à 12 kHz.



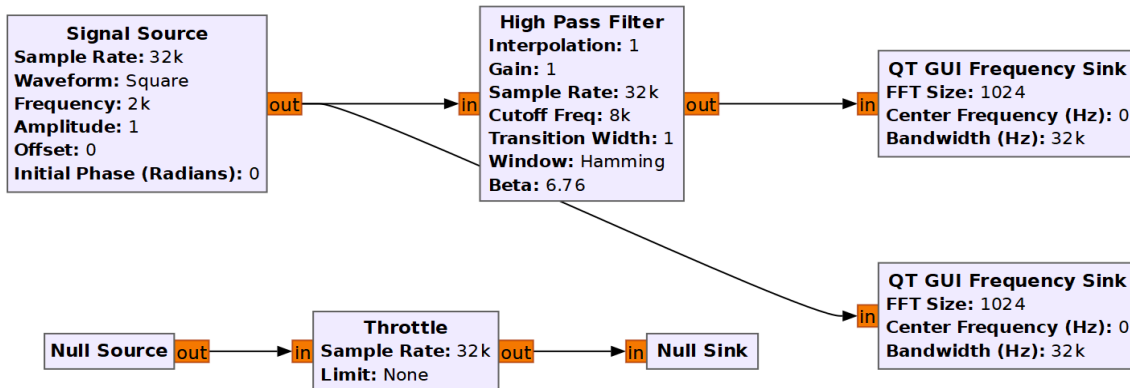
On exécute le logigramme afin de savoir comment le filtre passe bas se comporte. Sur l'image ci-dessous vous pouvez observer le signal généré avec un filtre passe bas et sans filtre.



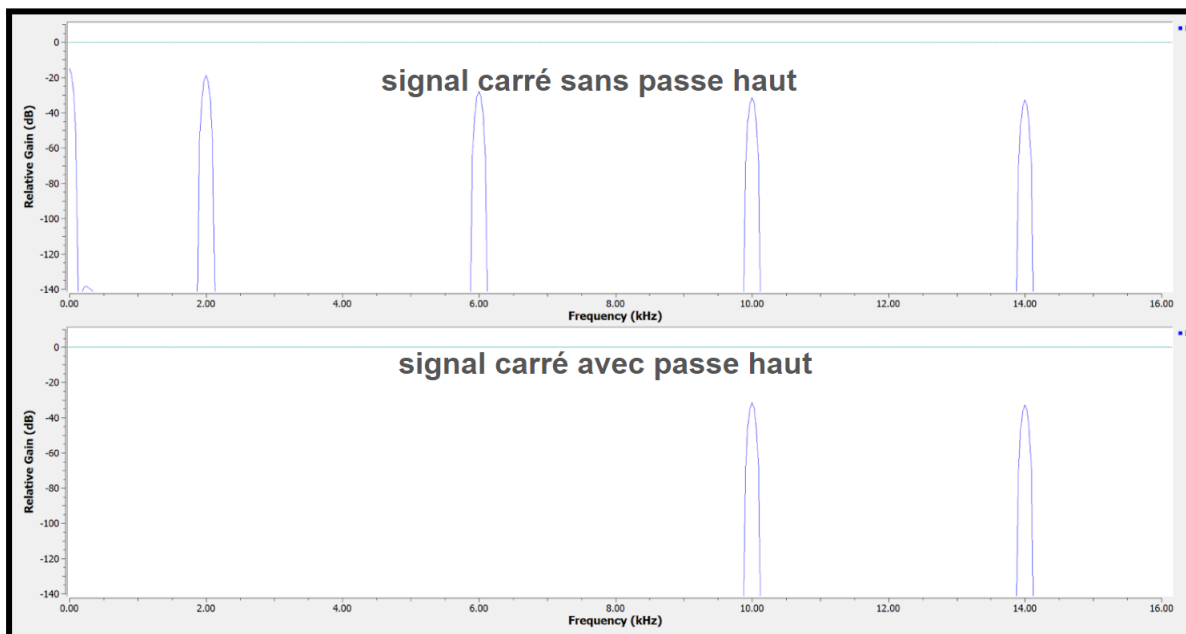
Donc on voit bien que le signal sans passe bas affiche une raie de plus que celui avec le passe bas. Pour être sûr que le spectre soit juste, on peut le vérifier par le calcul. En effet le spectre d'un signal carré possède des raies à  $1f$ ,  $3f$ ,  $5f$ ,  $7f$ . On voit bien que les 4 raies sont présentes sur le spectre du haut. De plus on peut observer que le spectre du bas à la dernière raie coupé à cause du filtre passe bas,  $7f$  étant inférieur à la fréquence de coupe du passe bas. Donc le filtre laisse bien passer les basses fréquences et il coupe les hautes fréquences ici  $7f$ .

## 4<sup>ème</sup> Démonstration:

Nous avons utilisé le même signal généré dans la deuxième démonstration. Cette démonstration a pour but de voir le comportement d'un passe haut s'appliquant à un signal carré généré à une fréquence de 2kHz dans GNU Radio. Ici on paramètre le passe haut de notre logigramme, de sorte à ne voir que les lobes : 5f et 7f. On lui applique comme paramètres, 1 en gain, 32k en fréquence d'échantillonnage, et une fréquence de coupure à 8 kHz.

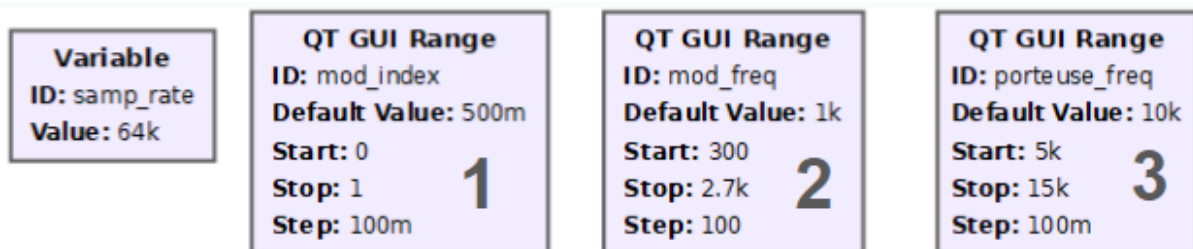


En haut on voit bien les 4 raies à 1f, 3f, 5f, 7f, notre signal n'est donc pas affecté par le passe haut. En revanche, en bas on voit bien que les raies à 1f et 3f ont bien été supprimées car elles sont inférieures à la fréquence de coupure qui est de 8 kHz. Donc on observe bien que le filtre passe haut laisse passer les hautes fréquences, 5f et 7f et il coupe bien les basses fréquences, 1f et 3f. Que ce soit pour un filtre passe bas ou un filtre passe haut il est possible de régler la fréquence de coupure afin de couper les fréquences que l'on souhaite.



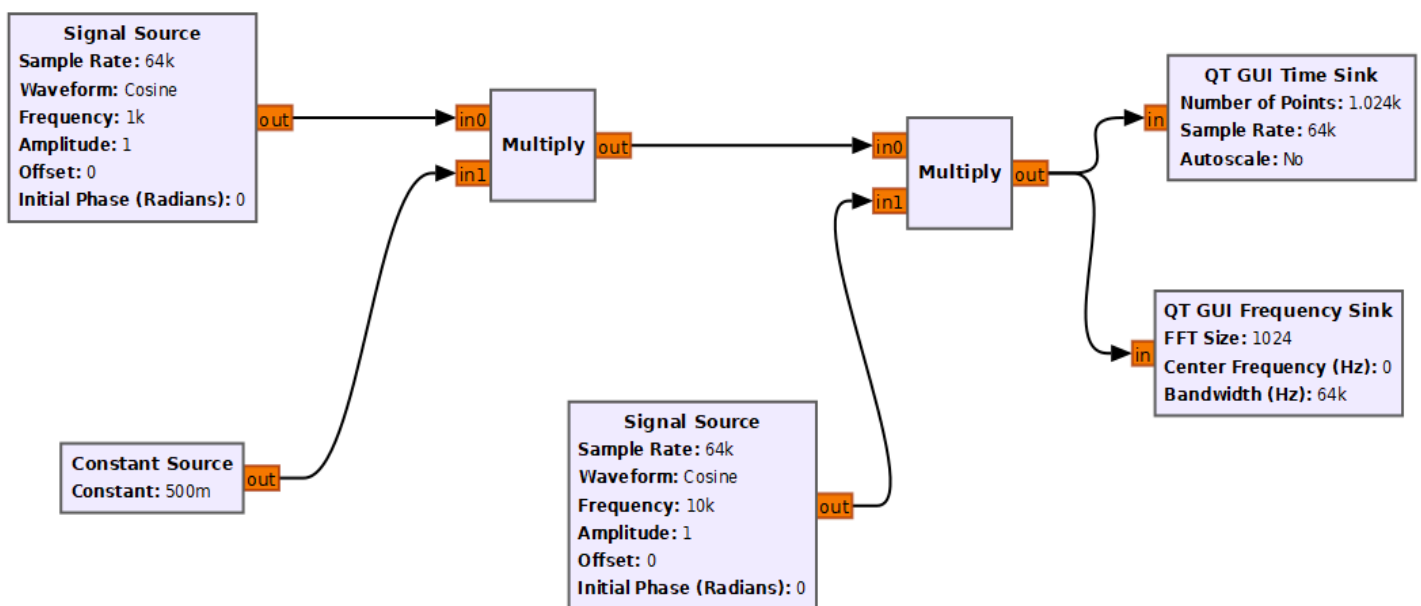
## 5<sup>ème</sup> Démonstration:

Dans cette démonstration nous allons voir comment faire de la modulation avec une fréquence porteuse qui sera définie à l'aide d'un variable curseur. Le but de la démonstration est de montrer le fonctionnement des variables à curseur mais aussi de montrer le fonctionnement de la modulation sur GNU Radio.



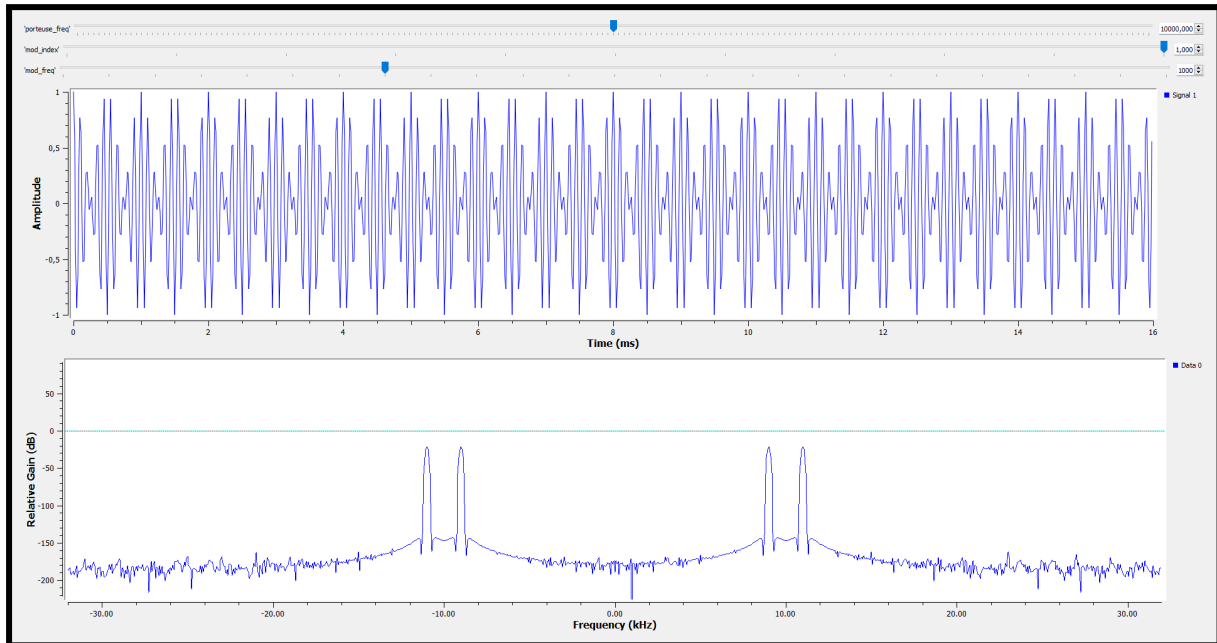
Nous rappelons que le bloc QT GUI Range permet de créer un curseur qui est associé à une variable qui fera varier un paramètre choisi. En effet dans notre cas les blocs QT GUI Range permettent de:

- 1) modifie la valeur de l'index (changera la constante et donc modifie l'amplitude)
- 2) modifie la valeur de la fréquence du signal
- 3) modifie la valeur de la fréquence porteuse

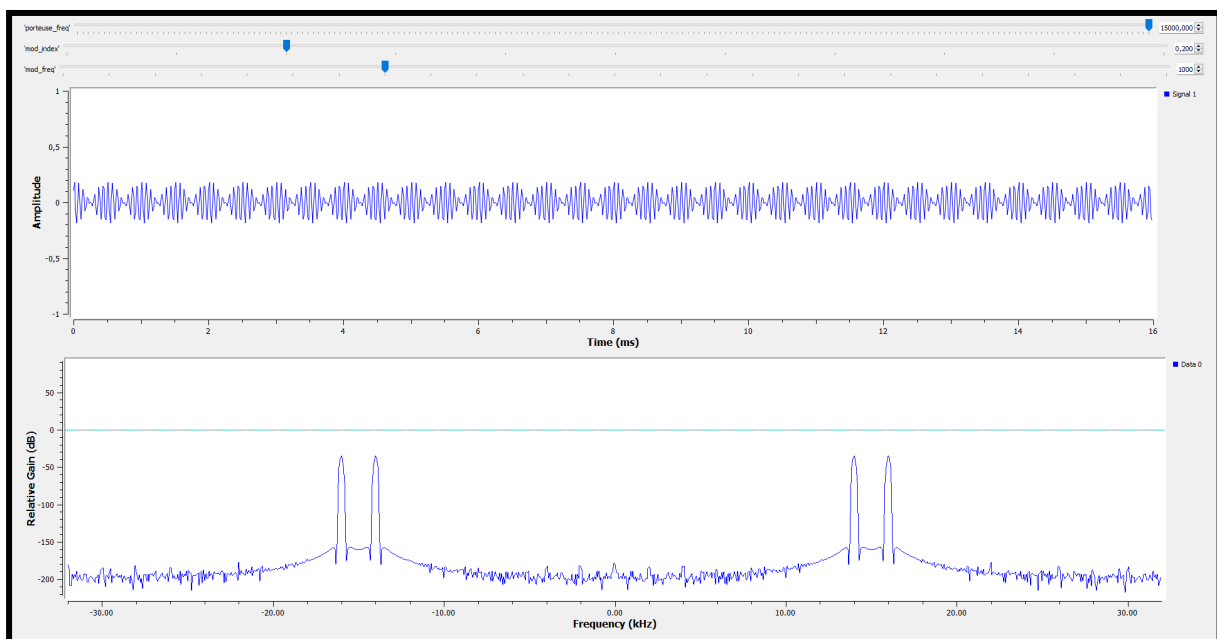


Ici de nouveaux blocs sont utilisés comme "Constant Source" qui est simplement une constante que l'on modifiera avec une variable. Ici "constant source" est multipliée au signal source et lui permet de faire varier l'amplitude. Add et Multiply sont des [opérations mathématiques](#) qui permettent de faire des calculs simples, comme ajouter deux signaux. Dans le cadre de cette démonstration, le bloc premier multiply permet de faire varier l'amplitude. Ensuite on associe au signal de base une fréquence porteuse afin de pouvoir réaliser un signal modulé.

Grâce aux blocs nous avons pu réaliser la modulation d'un signal. Le haut de l'image nous montre l'allure de notre signal modulé tandis que le bas de l'image représente le spectre modulé de notre signal. En effet, la fréquence porteuse nous a permis de moduler le signal. Nous pouvons faire varier le curseur de l'amplitude (mod\_index) afin d'obtenir une meilleure visualisation du signal. Nous pouvons également faire varier le curseur de la fréquence du signal qui va être modulé (mod\_freq) mais il sera également possible de faire varier la valeur de la fréquence porteuse (porteuse\_freq).



En effet, on peut modifier la valeur des curseurs afin d'obtenir de nouveaux résultats. Ici le curseur: "porteuse\_freq" a été mis au maximum soit 15kHz alors que sur l'image du dessus la fréquence porteuse était à 10kHz. En bas de l'image on observe bien que les deux signaux sont modulés à l'aide d'une fréquence porteuse. On observe bien que le signal modulé à changer car la fréquence porteuse est passé de 10kHz à 15kHz. De plus on peut observer que l'amplitude du signal à était modifiée car la valeur du curseur est passé de 1 à 0.200.



## **Présentation des catégories:**

Ici vous trouverez la description et l'explication de chaque catégorie regroupant des blocs spécifiques. Chaque catégorie sera accompagnée d'un des blocs présents dans cette catégorie à titre d'exemple pour illustrer la catégorie.

**Audio:** permet différentes interactions avec les entrées et sorties de son. On peut capter un son diffusé dans un périphérique externe (microphone) ou bien en peut directement lui donner un signal à traiter.

Source: est le bloc permettant au logiciel de traiter un son/signal mis en entrée.

Sink: Fait le contraire de "source", il permet de passer du logiciel à un périphérique ou à l'affichage du spectre par exemple.

**Boolean Operator:** permet au logiciel d'utiliser différentes opérations booléennes (and, or, xor) et de les appliquer aux différentes entrées.

And: permet d'utiliser l'opération AND (et), on peut l'utiliser sur deux signaux pour leur appliquer l'opération AND et aussi observer ce que cela fait.

Or: même chose que pour le bloc AND mais avec l'opération OR.

**Channelizers:** permet de diviser un spectre en plusieurs canaux, c'est-à-dire que les différentes bandes obtenues avec CHANNELIZERS peuvent être étudiées une à une et donc traiter par exemples que les signaux sur une certaine bande de fréquence.

Polyphase Channelizers: utilise le spectre mis en entrée afin de le découper en plusieurs canaux (parties) égales.

Polyphase synthesizers: A l'inverse de POLYPHASE CHANNELIZERS il permet de partir de plusieurs morceaux de spectre pour en reconstruire un seul.

**Bytes Operators:** a l'instar de "Booleans operators", Bytes operator remplissent les mêmes fonctions cependant avec des opérations de "Bytes" c'est-à-dire de la compilation et décompilation et aussi de la conversion.

Pack K bits: regroupe les bits d'un flux d'entrée en octets

Packed to Unpacked: Opération inverse et divise le résultat en plus petits paquets. (!\ il existe aussi "unpacked to packed")

**Channels Models:** simule des effets sur l'entrée comme du bruit, une altération ou atténuation.

Fading: Simule les variations du signal en fonction de l'environnement que ce soit les variations d'environnement ou même les obstacles entre l'émission et la réception.

Dynamic channel model: Pour un signal sans fil simule les variations au fil du temps (car un canal de communication sans fil peut rencontrer des changements fréquents en raison du mouvement de l'environnement, des obstacles, des conditions météorologiques, etc.)

**Coding:** permet au logiciel de produire du pseudo-aléatoire (car l'aléatoire n'existe pas en informatique) afin de modifier des signaux par exemple.

Scrambler: Ajoute un facteur aléatoire à un signal désigné, il le modifie pour ne plus être "prévisible".

Descrambler: inverse le processus de SCRAMBLER

**Control Port:** permet de choisir l'entrée (port) par laquelle on récupère le signal on peut donc avoir deux port et deux signaux différents. Il aide aussi au débogage car.

Control port monitor: donne les différentes valeurs d'un signal souhaitées?

Control Probe: permet la récupération de valeurs à certaines fréquences

**Debug tools:** permet le débogage des logigrammes et aussi aide à la compréhension des différents blocs (car on peut faire plus de tests).

Tags strobe: ajoute une balise(tags) a un instant qui "remplace" un signal pour effectuer des test sur ce dernier. Aide aussi à la synchronisation de signaux.

**Deprecated:** (obsolète) est une catégorie contenant différents blocs déconseillé d'utilisation car d'autres sont mieux ou avec moins de risques de problèmes.:

PSK mod: module un signal en PSK (dans cette catégorie car il existe un bloc plus sur BPSK mod)

PSK demod: inverse le processus PSK mod

**Digital Television:** Modulation et démodulation de télévision numérique. DVB et DVB-T sont deux moyens différents de faire la modulation et démodulation de signal de Télévision numérique.

**Equalizer:** permet de compenser ou corriger les distorsions qui ont eu lieu dans le canal de communication. L'égaliseur est utilisé pour atténuer les effets indésirables.

Égaliseur linéaire: (linear equalizer) est utilisé dans les communications sans fil. Il cherche à inverser les distorsions en utilisant des filtres linéaires pour ajuster l'amplitude et la phase du signal reçu. L'objectif est de rétablir la forme de l'onde du signal à son état initial.

**Error coding:** est un système qui permet de détecter et de corriger les erreurs qui se sont produites lors de la transmission d'informations sur le canal de communication à cause du bruit, d'interférences, ou d'autres perturbations.

Le codeur: (encoder) va convertir les données avec une certaine forme de redondance pour que le décodeur (decoder) puisse effectuer la même action dans l'autre sens, tout en corrigeant les erreurs selon le type de code utilisé.

**File operator:** permet de lire et d'écrire des données à partir de fichiers, ce qui peut être utile lors de la lecture de signaux enregistrés à partir d'un fichier ou l'enregistrement de signaux pour une analyse ultérieure.

File source: permet de lire les données stockées dans fichier pour s'en servir comme source du signal.

**Filter:** est un dispositif ou un algorithme qui permet de modifier les propriétés d'un signal. Un filtre est souvent utilisé pour rejeter certaines fréquences non voulues ou pour atténuer le bruit. Il existe deux types de filtres: les filtres analogiques et les filtres numériques.



Passe bas: (low pass) il laisse passer les basses fréquences et atténue les hautes fréquences

Passe haut: (high pass) il laisse passer les hautes fréquences et atténue les basses fréquences

Passe bande: (band pass) il laisse passer une certaine bande de fréquence et atténue tout ce qui est en dehors de cette intervalle

Coupe bande: (band stop) il atténue une certaines bande de fréquences et laisse passer toutes les autres fréquences

**Fourier analysis:** est une opération mathématiques qui permet de décomposer une fonction très complexe en une série de fonctions bien plus simple. Cette décomposition est effectuée grâce à la transformée de Fourier pour un signal continu ou discret.

La transformée de Fourier rapide: (Fast Fourier Transform = FFT) est un algorithme qui permet de calculer la transformée de Fourier discrète. Elle est utilisée pour effectuer l'analyse d'un spectre spectrale d'un signal discret.

**GUI widgets:** sont des éléments visuels qui permettent à l'utilisateur d'interagir avec l'élément voulu. Elles permettent en général de modifier les paramètres d'un signal.

GUI toggle button: permet à l'utilisateur d'interagir avec le logiciel. Il peut être utilisé par exemple pour activer ou désactiver une partie spécifique du traitement du signal.

**Impairment models:** permet de simuler et comprendre comment différents facteurs peuvent affecter la qualité des signaux ou la performance d'un système.

Phase balance: se réfère généralement à l'équilibrage des phases dans un signal.

La phase est une mesure du décalage temporel entre deux signaux. Elle permet de minimiser les variations indésirables.

**Instrumentation:** permet l'ajout d'éléments ou d'outils permettant de surveiller, mesurer ou déboguer le fonctionnement du système de traitement du signal. Il peut inclure la surveillance de paramètre tels que la qualité du signal, la consommation de ressources, les erreurs de transmission.

Qt GUI time sink: permet de visualiser graphiquement des signaux en fonction du temps. Il est possible de personnaliser divers aspects de l'affichage pour répondre à des besoins spécifiques. Il est possible d'ajuster les paramètres de visualisation en temps réel.

**Level controllers:** est associé à la gestion de la puissance du signal à différents points du flux de traitement.

AGC: correspond à Automatic Gain Control, c'est une technique de contrôle automatique du gain qui ajuste automatiquement le gain de l'amplificateur pour maintenir le niveau de sortie à un niveau constant malgré les variations du niveau d'entrée.

**Math operator:** sont des blocs qui incluent la plupart des opérations mathématiques que nous utilisons régulièrement.

Max: qui va enregistrer la valeur la plus grande

multiplication: (multiply) permet de réaliser une multiplication

**Measurement:** regroupe généralement des blocs de traitement signal qui sont utilisés pour effectuer des mesures sur ces signaux.

Probe Signal: fait référence à des blocs qui permettent d'extraire des informations à partir d'un signal (on va par la suite, pouvoir effectuer des mesures)

**Message Tools:** fonctionnalités liées à la manipulation et au transport de messages entre différents blocs de traitement du signal.

Message Debug: permet d'afficher les messages dans la console de débogage, facilitant le suivi.

**Misc:** contient des blocs ou des outils qui ne sont pas liés à une application ou à une fonctionnalité particulière, mais qui peuvent être utiles dans divers contextes.

Random source: génère des signaux aléatoires

**Modulators:** regroupe différents types de modulateurs, offrant aux utilisateurs de GNU Radio un ensemble d'outils pour réaliser diverses tâches de modulation en fonction de leurs besoins spécifiques.

Modulation ASK: modification de l'amplitude d'un signal

**Networking tools:** fonctionnalités permettant de créer des systèmes de traitement du signal distribués, où différentes parties d'un traitement peuvent être exécutées sur des machines distinctes connectées par un réseau.

UDP source et UDP sink: permettent la transmission et la réception de données via le protocole UDP (User Datagram Protocol)

**Orthogonal Frequency-Division Multiplexing (OFDM):** regroupe des blocs de traitement du signal spécifiquement conçus pour la modulation et la démodulation sur des systèmes de communication sans fil

ASK, FSK, PSK applicable sur ces systèmes de communication sans fil

**Packet Operators:** regroupe des blocs de traitement du signal qui sont utilisés pour manipuler des paquets de données dans un format particulier.

Packet header: gestion des en-têtes de paquets

**Peak Detectors:** permet le traitement du signal qui sont utilisés pour détecter et extraire des pics ou des valeurs maximales dans un signal.

Find Max: utilisé pour trouver l'indice et la valeur du maximum dans un signal

**Resamplers:** regroupe des blocs de traitement du signal qui sont utilisés pour changer le taux d'échantillonnage d'un signal.

Fractional resampler: permet un rééchantillonnage avec un facteur de rééchantillonnage fractionnaire.

**Stream Operators:** regroupe des blocs de traitement du signal qui sont utilisés pour effectuer des opérations sur un flux continu de données.  
Interleaves: utilisés pour réorganiser l'ordre des échantillons dans un flux de données.

**Stream tag tools:** collecte toutes les balises qui lui sont envoyées sur les ports d'entrées et les affiche sur la sortie standard.  
Tag debug: Lors de la connexion d'un bloc à ce récepteur de débogage, un nom approprié lui sera attribué. Ce bloc agit comme un déboguer.

**Symbol coding:** exploite une boucle, généralement une longueur d'échantillonnage d'un symbole, puis ajuste cette boucle jusqu'à ce qu'elle soit correctement alignée avec chaque symbole de la série.  
Digital\_diff\_phasor\_cc: décodage différentiel basé sur le changement de phase. Il permet de calculer la différence de phase entre deux symboles pour déterminer le symbole de sortie

**Synchronizers:** consiste à obtenir une estimation de décalage du temps et de phase. Ces estimations sont transmises en aval sous forme de balise de flux pour être utilisées par les blocs de synchronisation et de suivi.  
Symbol\_sync\_cc: Bloc de synchronisation de symboles avec entrée complexe, sortie complexe. Cela implémente un synchroniseur de suivi des erreurs à temps discret.

**Trellis Coding:** est un encodeur convolutif, un encodeur convolutif est fréquemment utilisé dans la conception des codes correcteurs d'erreurs, notamment dans le domaine des communications numériques.  
Trellis Metrics: Génère les métriques requises pour les algorithmes Viterbi ou SISO.

**Type Converters:** consiste à convertir des données binaires. Il faut savoir que la plupart du temps nous utilisons: float 64, float 32 et bits.  
Random Source: génère aléatoirement des valeurs de 0 et 1.

**UHD:** fournit une interface pour configurer et contrôler les paramètres du matériel USRP, tels que la fréquence d'échantillonnage, la fréquence de la porteuse, la puissance de transmission, etc. Il permet également de recevoir et de transmettre des données entre GNU Radio et le matériel USRP (Universal Software Radio Peripheral).  
RFNoC TX Radio: utilisation du framework RFNoC (Radio Frequency Network on Chip) pour la transmission (TX) radio dans des applications de radio logicielle.  
RFNoC est un cadre de développement qui permet d'implémenter des fonctions de traitement du signal radio sur des FPGA (Field-Programmable Gate Arrays) pour des plateformes matérielles spécifiques, notamment les périphériques USRP.

**Variables:** lie une valeur à une variable unique. On peut utiliser la variable dans le champ paramètre d'un bloc simplement en utilisant l'ID du bloc variable. Le bloc variable est un moyen pratique d'avoir une valeur utilisée à plusieurs endroits différents tout en pouvant la modifier rapidement.

Variable Config: ce bloc représente une variable qui peut être lue à partir d'un fichier de configuration. Pour enregistrer la valeur dans le fichier de configuration : entrez le nom d'une autre variable dans le paramètre de réécriture. Lorsque l'autre variable est modifiée au moment de l'exécution, le fichier de configuration sera réécrit.

**Vidéo:** est utilisé pour afficher des données vidéo générées dans un flux GNU Radio. Il peut être intégré dans un diagramme de flux GNU Radio pour visualiser des signaux vidéo en temps réel ou pour déboguer des composants liés à la vidéo. Video sdl sink: lorsqu'on utilise ce bloc dans un diagramme de flux GNU Radio, vous pouvez le configurer en spécifiant des paramètres tels que la taille de la fenêtre d'affichage, la résolution vidéo, le nombre d'images par seconde, etc. Ce bloc peut être intégré avec d'autres blocs.

**WaveForm:** permet de générer des signaux.

Random Source: Génère de nombreux échantillons de nombres aléatoires de [min, max). Répétez les échantillons si spécifiés.

Ex: Avec min=0 et max=2, la séquence 01110101... sera générée.

Si on souhaite un générateur de nombres aléatoires traditionnel, on utilisera plutôt Random Uniform Source.

**ZéroMQ:** est une bibliothèque asynchrone qui fournit des primitives de communication entre processus. Ce bloc permet de créer des connexions pour échanger des données entre différents composants.

Zeromq\_pub\_sink: Ce bloc fonctionne comme un récepteur de streaming pour un organigramme GNU Radio et écrit son contenu vers un socket ZMQ PUB. Une socket PUB peut avoir des abonnés et transmettra toutes les données de flux entrantes à chaque abonné avec une clé correspondante. Si la clé de l'éditeur est défini sur "GNURadio", les exemples de clés d'abonné suivants correspondront: "G", "GN", ..., "GNURadio"

En d'autres termes, l'abonné doit contenir le premier ensemble de caractères de la clé de l'éditeur. Si l'abonné définit une clé vide "", il sera accepté par tous les messages d'entrée de l'éditeur (y compris la clé elle-même si elle est définie).





## **Vidéo:**

Cliquez directement sur l'image ou bien sur le lien ci-dessous.

[https://youtu.be/gi-LgKrT9\\_o?si=ZFy8PN-tWDTJ\\_eUf](https://youtu.be/gi-LgKrT9_o?si=ZFy8PN-tWDTJ_eUf)



## GNU Radio organisation:

Louis Gensou	Kylian Ollivier	Yann Esteves	Alexandre Parnaudeau
<ul style="list-style-type: none"> <li>- Présentation des 10 dernières catégories, de 30 à 40 (du 15/11/2023 au 1/12/2023)</li> <li>- Vidéo d'installation et de démonstration sur GNU Radio (du 19/12/2023 au 22/12/2023)</li> </ul> 	<ul style="list-style-type: none"> <li>- Présentation des catégories de 10 à 20 (du 15/11/2023 au 1/12/2023)</li> <li>- Présentation de GNU Radio (du 1/12/2023 au 4/12/2023)</li> <li>- Présentation de l'interface utilisateur (du 4/12/2023 au 05/12/2023)</li> <li>- Les éléments à savoir (du 6/12/2023 au 14/12/2023)</li> </ul> 	<ul style="list-style-type: none"> <li>- Présentation des catégories de 20 à 30 (du 15/11/2023 au 1/12/2023)</li> <li>- Installation de GNU Radio sous Ubuntu, Windows (du 1/12/2023 au 4/12/2023)</li> <li>- Principaux blocs (du 04/12/2023 au 08/12/2023)</li> <li>- Prise en main de la partie python (du 8/12/2023 au 15/12/2023)</li> </ul> 	<ul style="list-style-type: none"> <li>- Présentation des catégories de 0 à 10 (du 15/11/2023 au 1/12/2023)</li> <li>- Interfaces affichages des signaux, spectres (du 01/12/2023 au 06/12/2023)</li> <li>- Démonstrations (du 6/12/2023 au 18/12/2023)</li> </ul> 

Début du brouillon: 15/11/2023

Fin du brouillon: 01/12/2023

Début de la rédaction rapport: 01/12/2023

Fin de la rédaction du rapport: 22/12/2023

Début de la réalisation du diaporama: 22/12/2023

Fin de la réalisation du diaporama: 8/01/2024



## Sources:

About GNU Radio. (s. d.). GNU Radio. <https://www.gnuradio.org/about/>

PEZET, B. (s. d.). Introduction à GNU-Radio [Base de données].  
<https://ref31.r-e-f.org/Gnuradio-demo.pdf>

Risset, T. (s. d.). First French GNU Radio days Tutorial [Base de données].  
[https://gnuradio-fr-18.sciencesconf.org/data/pages/OOT\\_bloc\\_lab.pdf](https://gnuradio-fr-18.sciencesconf.org/data/pages/OOT_bloc_lab.pdf)

Free online gantt chart software. (s. d.). <https://www.onlinegantt.com/#/gantt>

Posts, V. A. O. J. C. (2022, 18 mai). AM Modulation on GNU Radio – Telecommunications, Navigation & Electronics. <https://jeremyclark.ca/wp/telecom/am-modulation-on-gnu-radio/>

Introduction aux radios logicielles avec GNU Radio/Prise en main — Wikilivres.  
(2014).[https://fr.wikibooks.org/wiki/Introduction\\_aux\\_radios\\_logicielles\\_avec\\_GNU\\_Radio/Prise\\_en\\_main](https://fr.wikibooks.org/wiki/Introduction_aux_radios_logicielles_avec_GNU_Radio/Prise_en_main)

HB9AFO : GNU Radio. (s. d.). <https://www.hb9afo.ch/articles/GNU%20Radio/default.htm>

Jeremy Clark. (2022, 18 mai). AM modulation on GNU Radio [Vidéo]. YouTube.  
<https://www.youtube.com/watch?v=FxB8kpYE5k>

Frédéric Daigle. (2017, 20 février). Présentation GNU Radio [Vidéo]. YouTube.  
<https://www.youtube.com/watch?v=njEsnGvJQtw>