

1. SDR 介绍

软件定义无线电（Software Defined Radio, SDR）的主要设计构想是将波形处理，如调制、解调等交给 CPU 实现，而将射频信号的收发，数字上下变频、**抽样**、**插值**等高速处理过程交给其硬件设备，该硬件设备也称 USRP（Universal Software Radio Peripheral，通用软件无线电外设），包含两个部分：进行高速信号处理的 FPGA 和覆盖不同频率范围 RF 前端。如图 1.1 所示。

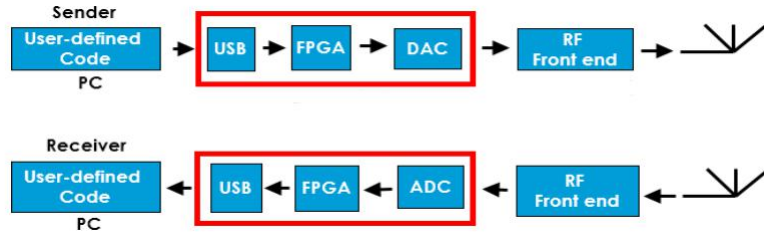


图 1.1 软件定义无线电（SDR）系统

目前的主流 SDR 系统的软件和硬件是分开实现的，各自实现各自的功能。SDR 软件部分主要是运行在计算机上的信号处理软件，硬件部分用于发射和接收物理信号。两者的接口是一个复信号 $s(t)$ 。如图 1.2 所示。



图 1.2 SDR 结构示意图

其中， $s(t)$ 为计算机产生的带限复信号序列，称为复包络信号。

$$s(t) = a(t) + ib(t)$$

$a(t)$ 和 $b(t)$ 可以是软件产生的任意实序列，分别称为 I 路（同相）和 Q 路（正交）信号。

SDR 硬件将由用户指定中心频率，带宽，增益等参数。在发射模式下，SDR 硬件将用 IQ 调制的方式将 $s(t)$ 的频谱搬移到中心频率附近，形成实连续信号 $r(t)$ ，然后通过天线发射。在接收模式下 SDR 硬件将用 IQ 解调的方式从 $r(t)$ 中解析出 IQ 两路信号。

IQ 调制是 SDR 系统采用的调制方式。在计算机中我们输出给 SDR 硬件的信号是一个复信号，而物理上我们只能传播实信号。将一个基带复信号搬移到通带上的调制方式就是 IQ 调制。记传输给 SDR 的信号为 $s(t)$

$$s(t) = a(t) + ib(t)$$

$s(t)$ 称为复包络信号。设载波频率为 f ，那么 SDR 系统实际传输信号 $r(t)$ 为

$$r(t) = a(t) \cos(2\pi f t) - b(t) \sin(2\pi f t)$$

$r(t)$ 是一个实信号，可以通过电磁波传输。 $a(t)$ 称为 I 路信号，也叫同向分量， $b(t)$ 称为 Q 路信号，也叫正交分量。

在 SDR 上应用 IQ 调制的原因是，相位调制是一种高效的调制方式，多数通信系统都采用相位调制来传递信息，IQ 调制会给相位调制系统带来很大的方便。例如，在 PSK 调制中，解调系统通过一个符号区间的载波的相位判定符号。如果采用 IQ 调制

$$r(t) = a(t) \cos(2\pi f) - b(t) \sin(2\pi f)$$

可以化成

$$r(t) = A \cos(2\pi f + \varphi)$$

其中相位 φ 由二维坐标

$$(a(t), b(t))$$

唯一决定。此时，我们只要观察某个采样点，也就是 $(a(t_0), b(t_0))$ 在星座图的位置，就可以知道传递的载波相位。同理，只要将 $(a(t_0), b(t_0))$ 的坐标表示成方波，然后直接送给 SDR 接口进行传送，就可以直接在射频端得到 PSK 已调信号。

IQ 解调是从混合的

$$r(t) = a(t) \cos(2\pi f) - b(t) \sin(2\pi f)$$

信号中解调出 $a(t)$ 和 $b(t)$ 的过程。记混合信号为 $m(t)$ ，那么在理论上，将 $m(t)$ 分别乘上 $\cos(2\pi f)$ 和 $-\sin(2\pi f)$ ，经过低通滤波就可以分别解调出 $a(t)$ 和 $b(t)$ 。但是，由于发射机和接收机的载波频率 f 可能不一致，载波的初始相位不同，解调无法完全还原 $a(t)$ 和 $b(t)$ 。需要通过另外的调制解调设计才能从解调得到的 IQ 两路信号中得到提取出信息。

目前 SDR 的主流方案将 SDR 系统分为硬件平台和软件平台。硬件平台包括射频、天线模块，负责发送和接收无线电波；也有 FPGA 模块负责信号的高速处理。在计算机上安装好一种 SDR 软件平台相应的硬件驱动，处理好接口，就可以使用这些硬件平台。通常 SDR 硬件需要连接到计算机上才能使用，但是一些 SDR 硬件在不连接计算机的情况下也能实现一些简单的功能。常见的 SDR 硬件实现有 HackRF、bladeRF、USRP、Zynq SDR、RTL-SDR、limeSDR、ADALM-PLUTO 等。

原则上任何通用程序设计语言都可以用于编写 SDR 应用。当然，在一些已有的平台上编写程序会非常方便，因为这些平台或者有强大的信号处理功能，或者内置了很多信号处理常用模块，有很多虚拟通信器件等。常用的 SDR 软件平台有 GNURadio 和 Matlab/Simulink。

2. HackRFone 介绍以及 SDR Sharp 实验

2.1 HackRFone 介绍

HackRFone 是一个完全开源的 SDR 硬件，作者是 Michael Ossmann。作者在其 GitHub 上公开了 HackRF 接口的源代码以及 PCB 设计等所有资料。图 2.1 是 HackRF 的外观图。HackRF 开源资源：<https://github.com/mossmann/hackrf>

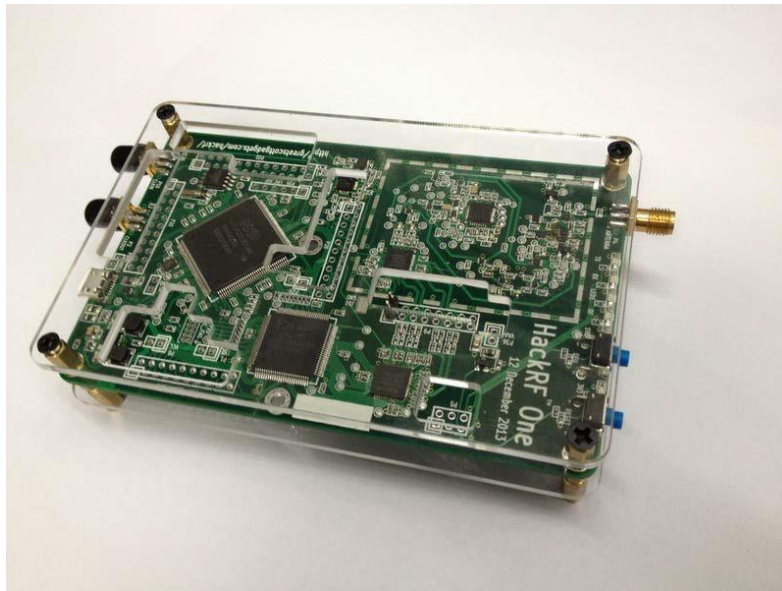


图 2.1 HackRF 外观图

如图 2.2 所示，HackRF 有 3 个 SAM 接头，分别标有 ANTENNA, CLKIN, CLKOUT。ANTENNA 口连接天线，CLKIN/CLKOUT 口作为 10MHz 时钟信号的输入输出，可以用于多个设备之间的时钟同步。一个 microUSB 接口用于连接计算机，传递信号。

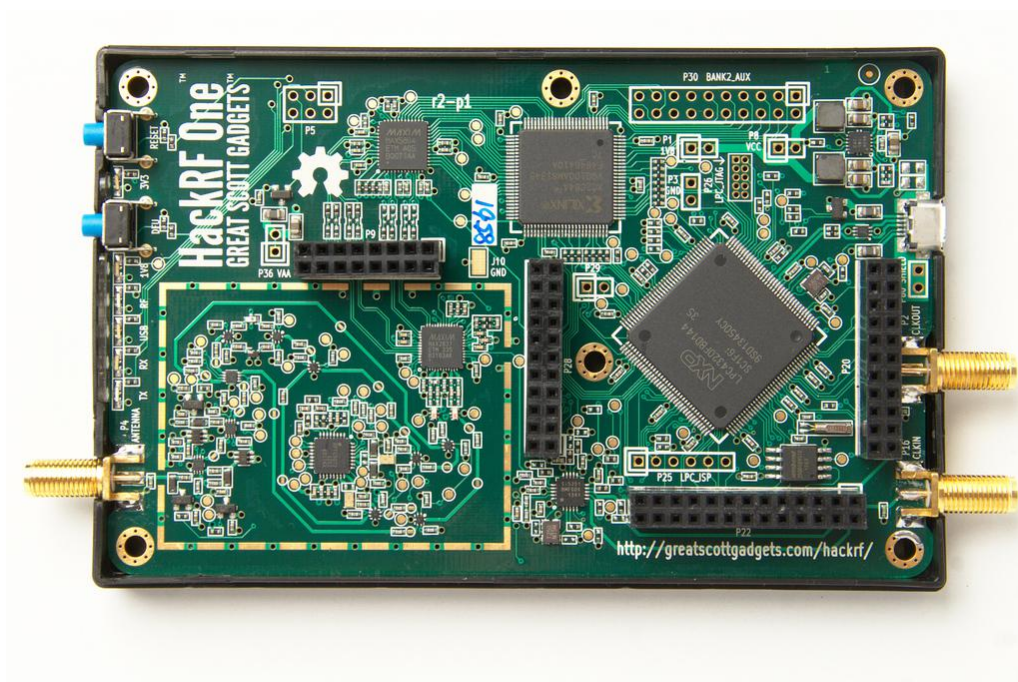


图 2.2 HackRF 电路板

2.2 HackRF 在 Windows 下驱动程序的安装

HackRF 在 Windows 下需要安装驱动程序，为了方便通常使用 `zadig` 来安装。用数据线连接 HackRF 的 microUSB 接口和计算机的 USB 接口。运行 `Zadig`，如图 2.3。单击 `Options->List All Devices`，显示所有的设备。

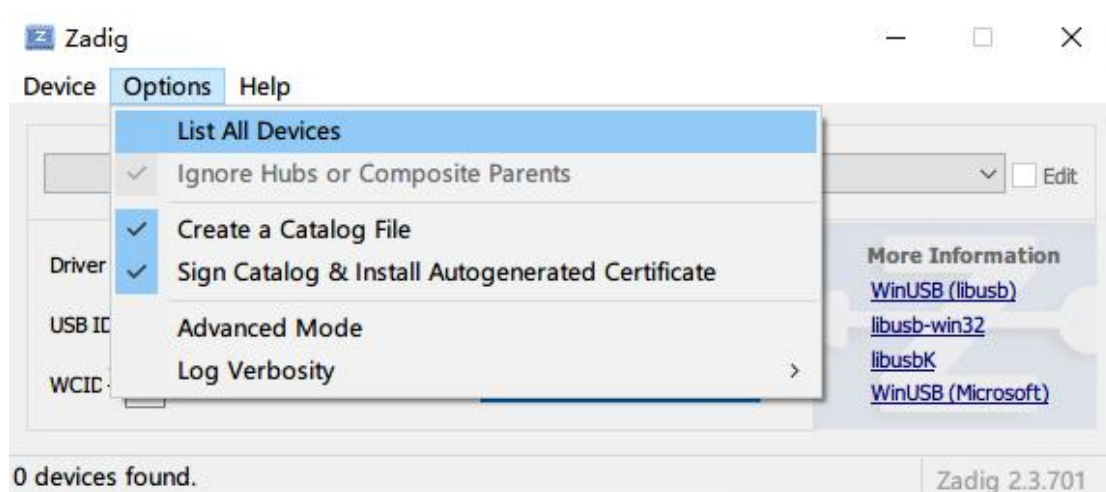


图 2.3

如图 2.4，选择设备 `HackRFOne`，在绿色箭头后的框中选择目标驱动程序 `WinUSB`，然后单击按钮 `Replace Driver`，完成设备的安装。

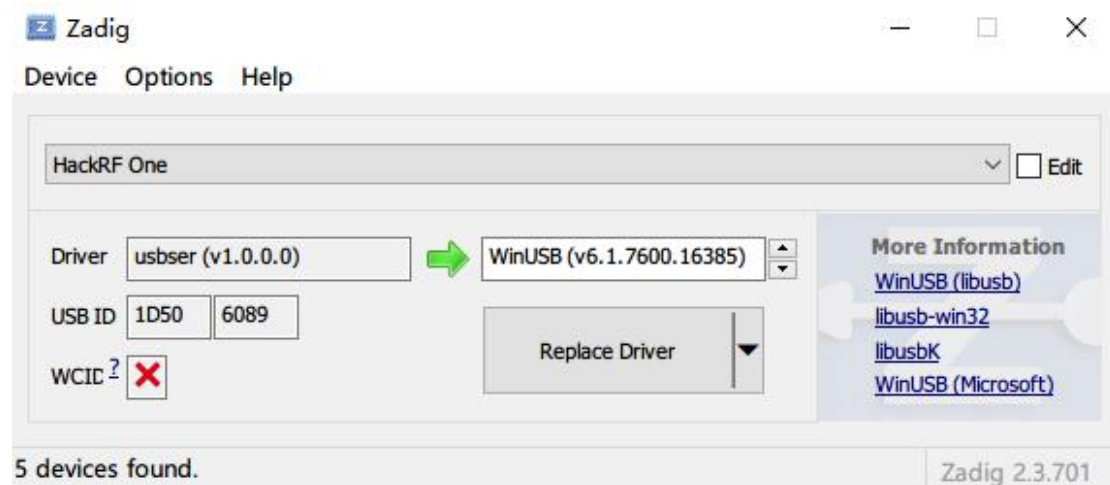


图 2.4

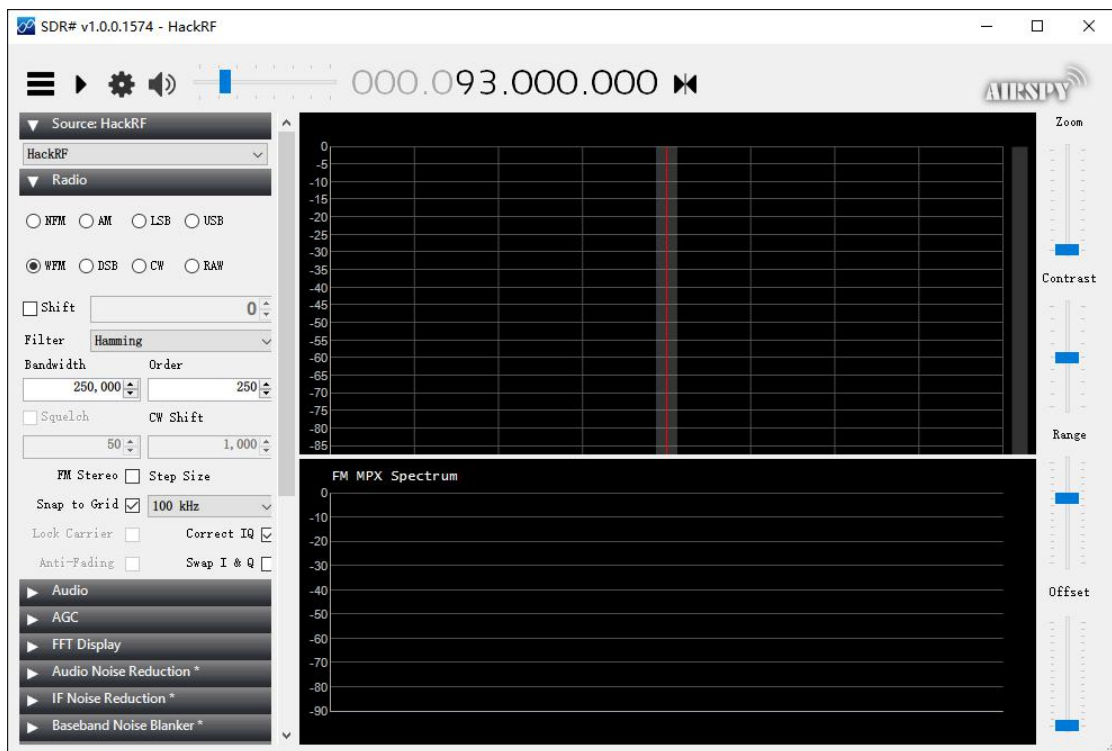
提示：当有多个 HackRF 设备时，如果已经为一个设备安装好了驱动，那么当更换一个一样的 HackRF 设备时，这个新设备可能无法工作。可以通过为新设备重复上述安装步骤使其工作。

2.3 在 SDRSharp 软件平台下的几个实验

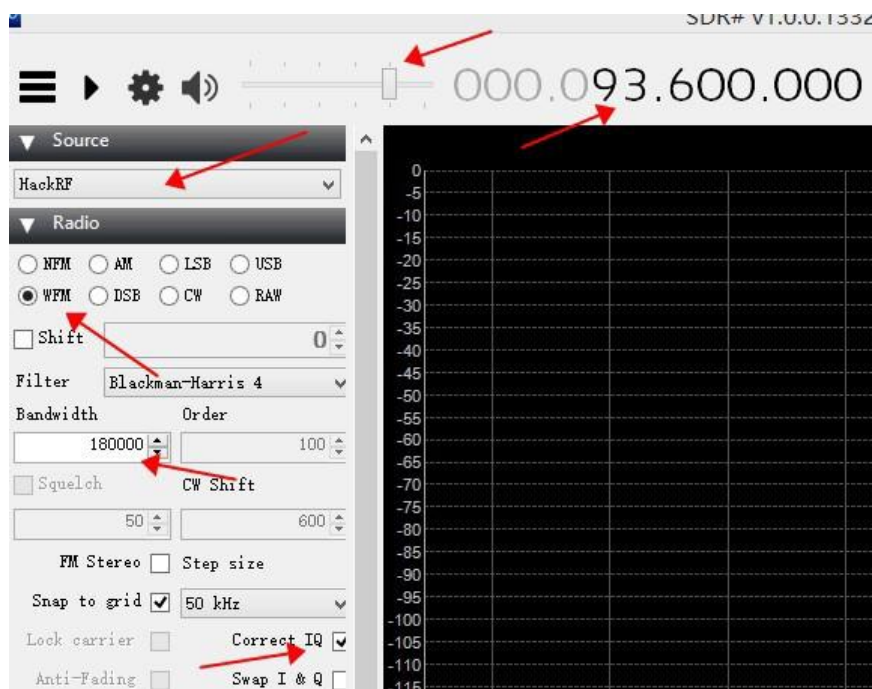
2.3.1 收听 FM 广播

在 SDRSharp 安装目录下运行 `SDRSharp.exe`，打开 SDRSharp 的主界面

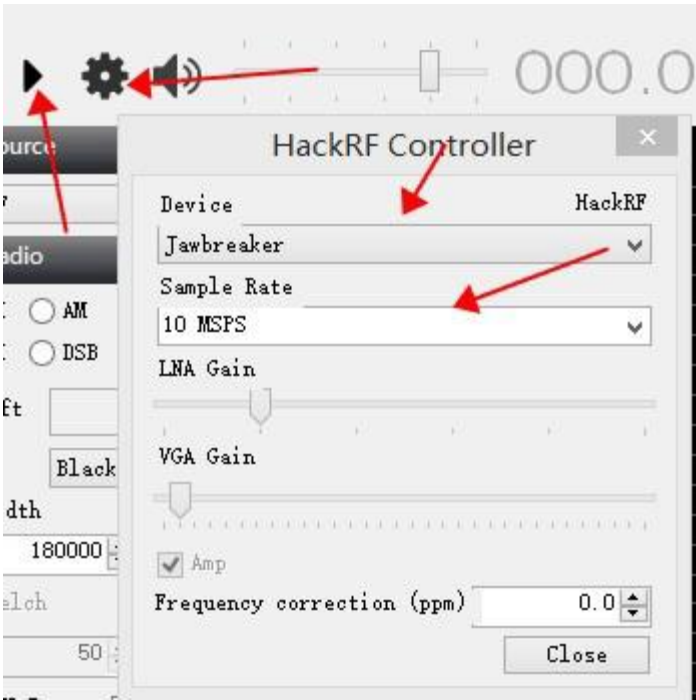
SDRSharp.Common.dll	2017/5/28 2:07	应用程序扩展	13 KB
SDRSharp.DNR.dll	2017/5/28 2:07	应用程序扩展	26 KB
SDRSharp.exe	2017/5/28 2:07	应用程序	268 KB
SDRSharp.exe.Config	2019/7/31 5:54	XML Configurati...	8 KB
SDRSharp.FrequencyEdit.dll	2017/5/28 2:07	应用程序扩展	23 KB
SDRSharp.FrequencyManager.dll	2017/5/28 2:07	应用程序扩展	38 KB



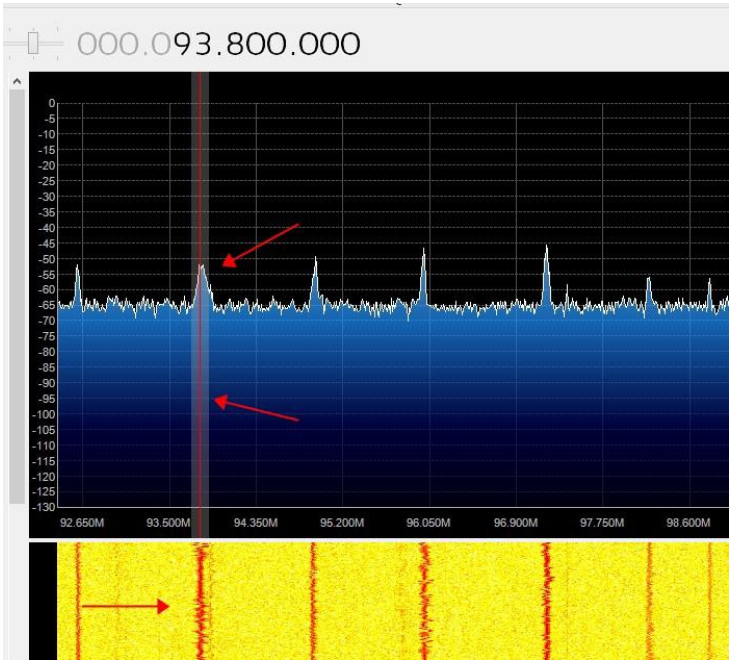
打开 SDRSharp 后将设置调整到下图所示。箭头指向从上到下分别指向：音量，频率（Hz），设备源，调制方式（选择宽带 FM），调制带宽，Correct IQ。



下图箭头指示为启动停止，配置按钮。HackRF Controller 为配置面板，DEVICE 下为设备名称，Sample Rate 为采样速率，LNA Gain 与 VGA Gain 为增益控制，Frequency correction 频率偏差校准。

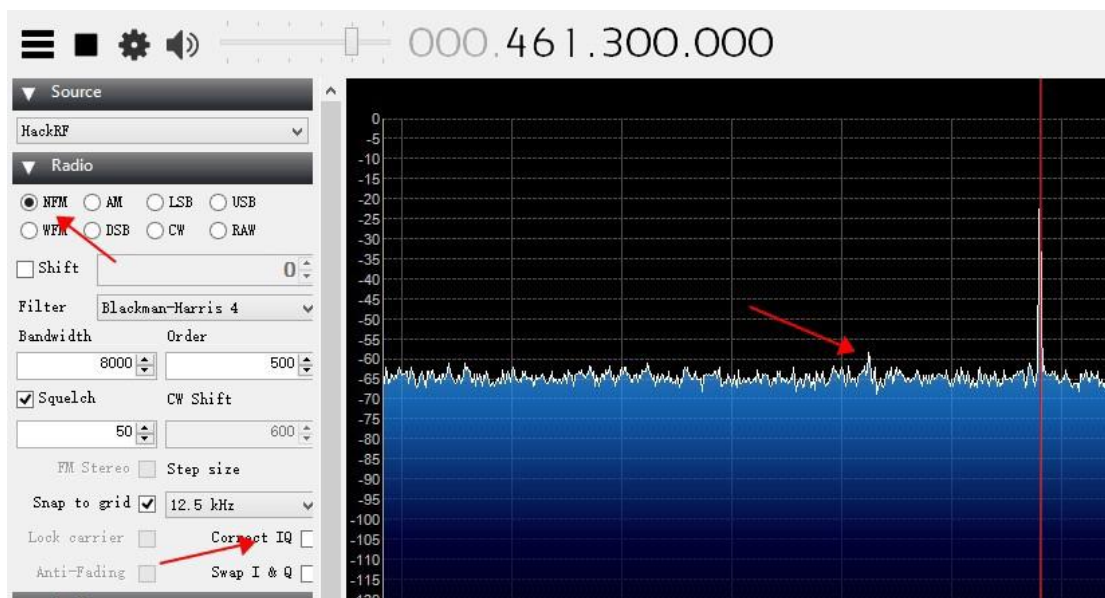


通常，在 FM 接收时，白天设置 LNA 为 16，晚上设置为 8 可正常工作，VGA 设置为 10，AMP 勾选。设置完成后单击 CLOSE。单击启动按钮，结果如下图所示。从信号波峰中选择一个收听。用鼠标单击想要设为中心频率的位置，即可收听广播。



2.3.2 监听对讲机

选择窄带 FM (NFM)，去除 Correct IQ 选项，以免听到不断的微波背景噪声，在波形窗口中偶尔出现的波峰即是对讲机发出的信号。

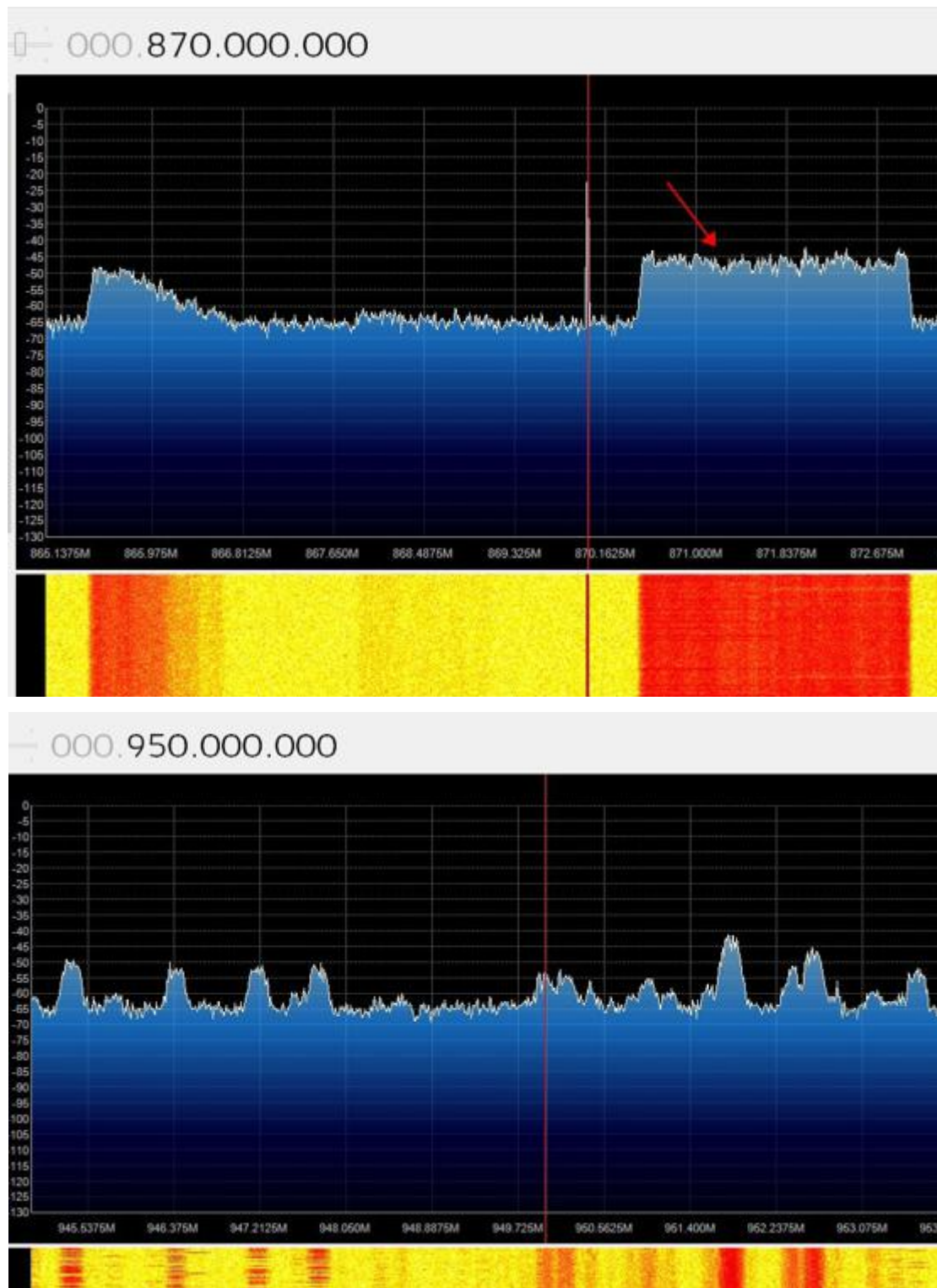


2.3.3 监听家用无线遥控

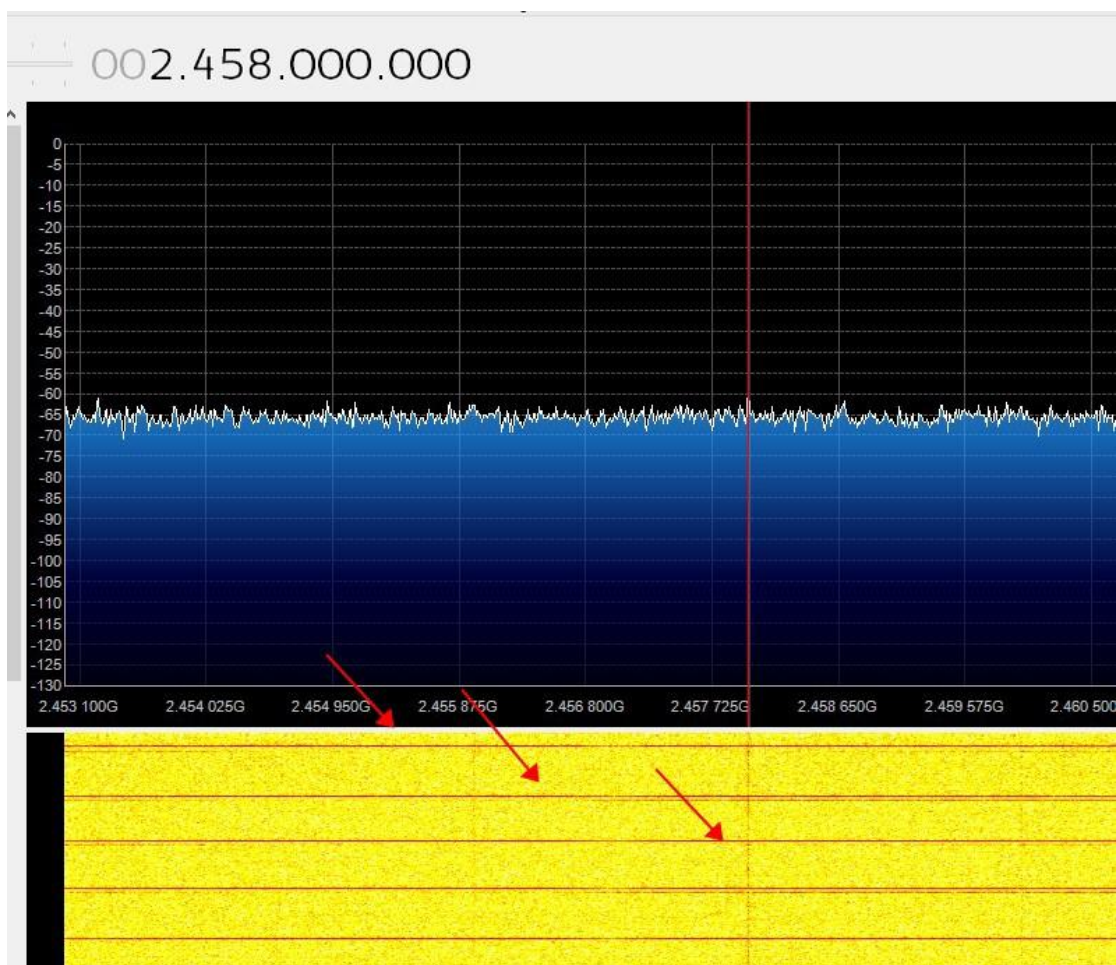
家用无线遥控的频率通常是 315MHz 和 433.92MHz，将频率直接移动到这个中心频率即可。

2.3.4 观察 GSM 信号

GSM 信号有两个频率 870MHZ，950MHZ。下图演示了两个频率下的 GSM BUSH 情况采样均为 10MSPS。



2.3.5 观察 WIFI 信号



上图箭头指示了 WIFI 信号在通讯时的水波情况，2.457GHz 为 10 信道 WIFI 信号所在的频段。信号所在的具体频率取决于附近的 WIFI 信号所用的信道。

下面给出了 2.4GHzWIFI 信号频率表，以供参考。

信道	中心频率	信道	中心频率
1	2412MHz	8	2447MHz
2	2417MHz	9	2452MHz
3	2422MHz	10	2457MHz
4	2427MHz	11	2462MHz
5	2432MHz	12	2467MHz
6	2437MHz	13	2472MHz

3. GNURadio

3.1 GNURadio 介绍

GNURadio 是著名的开源 SDR 软件平台，是目前广泛使用的一种平台。GNURadio 可以选择是否连接硬件。在不连接硬件的情况下，GNURadio 可以作为一个信号处理模拟环境。可以用这种方式理解：GNURadio 本身是作为一个信号处理系统开发的，在搭载上硬件 SDR 平台后，得到了发射和接收信号的功能。

GNURadio 的平台设计基于两点：一是模块化，二是模块间的数据流动。一个信号处理单元，或者叫做模块，由一个信号处理过程及其接口组成。信号的处理程序因为要求高速处理，需要用 C++ 编写。GNURadio 内置了大量的信号处理模块。在一个 GNURadio 的信号处理程序中，我们需要调用内置的或者自己编写的模块，然后将它们连接起来，接着运行该流图。模块的连接是 GNURadio 平台提供的一个重要功能，用户只需要简单调用函数将模块连接起来即可，GNURadio 平台的设计将保证信号在模块之间高速，有效地流动。

一个可运行的流图必须包含信源模块、信宿模块，两者之间可以加上需要的信号处理模块。信源模块可以是正弦波、方波等波形，可以是文件、音频波形、符号向量等；信宿模块可以文件、音频、示波器、频谱仪等；信号处理类模块大多都能在实际通信系统中找到对应的模块，如基础运算类、调制解调类、滤波器类模块等。

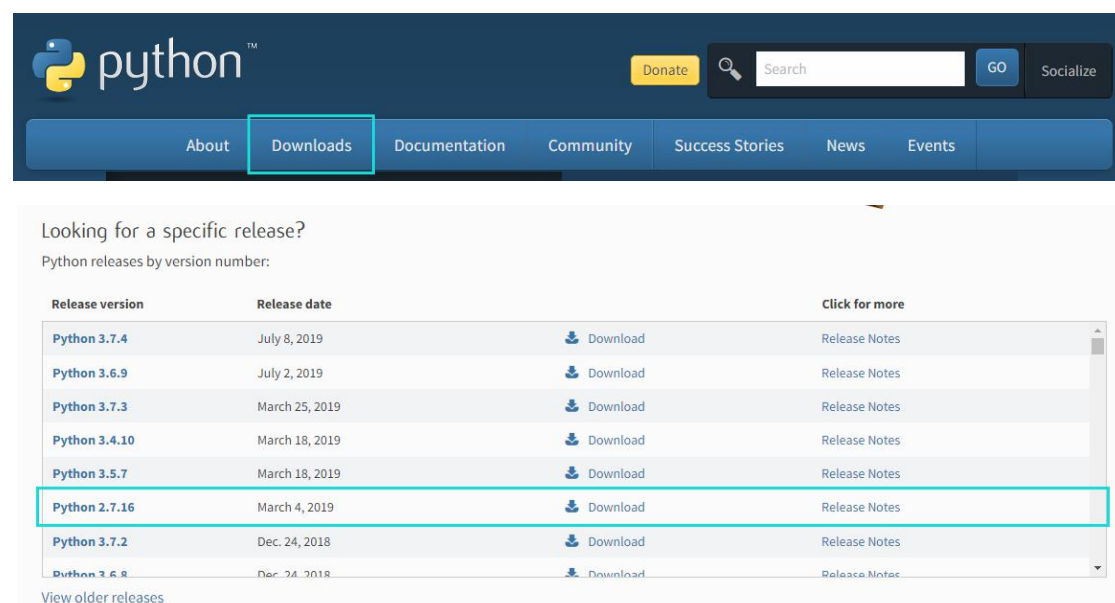
SDR 硬件连接到计算机上后，在 GNURadio 中也是通过模块调用的。SDR 硬件可以作为流图中的信源或信宿。

GNURadio 有 C++ 和 Python 的 API，模块算法为了执行速度使用 C++ 编写，然后将模块的应用和连接脚本化，使用 Python 编写。通常，写 GNURadio 程序在自定义模块时使用 C++，其它时候使用 Python。

3.2 GNURadio 的安装 (Windows)

3.2.1 安装 python2.7

在 Python 官网 <https://www.python.org/> 上下载 Python2 的最新版本。以当前时间为例，下载 Python2.7.16。



Release version	Release date	Download	Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes
Python 3.6.8	Dec. 24, 2018	Download	Release Notes

[View older releases](#)

Files					
Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		f1a2ace631068444831d01485466ece0	17431748	SIG
XZ compressed source tarball	Source release		30157d85a2c0479c09ea2cbe61f2aaf5	12752104	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	70b0f58eba7b78b174056369b076c085	30252432	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	a3af70c13c654276d66c3c1cb1772dc7	23743901	SIG
Windows debug information files	Windows		f94690edbbf58b10bfd718badc08b1f8	25088166	SIG
Windows debug information files for 64-bit binaries	Windows		4292c4db30c27fedbbe8544967b6452	25899174	SIG
Windows help file	Windows		3bbf29b6712b231d2dff9211fc7b21e2	6263118	SIG
Windows x86-64 MSI installer	Windows	for AMD64/EM64T/x64	2fe86194bb4027be75b29852027f1a79	20361216	SIG
Windows x86 MSI installer	Windows		912428345b7e0428544ec4edcd70286	19419136	SIG

图 3.1 下载 Python 2.7.16

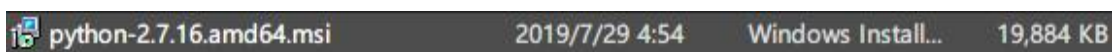


图 3.2 下载后的安装程序

双击运行安装程序，按照提示安装。注意在选择组件这一步选择将 Python 添加到环境变量中。

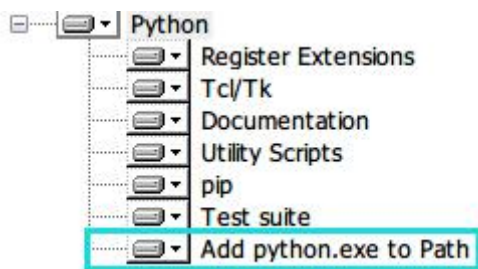


图 3.3

完成安装后查看环境变量。打开控制面板->系统与安全->系统->高级->环境变量->系统变量->Path，查看 Python 的安装路径是否被添加到了环境变量中。



图 3.4 环境变量 Path

打开 cmd，输入 python，如果出现以下文字，则说明 python 已被正确安装。输入 exit() 退出 python 环境。

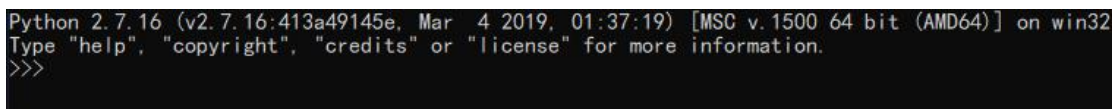


图 3.5

3.2.2 安装 GNURadio

GCNDevelopment 将 GNURadio 移植到了 Windows 上。GNURadioWin64 版的安装包可以在 GCNDevelopment 的网站上下载到。下载完毕后按照官网文档中的指引，安装主程序和矢量计算优化库 VOLK。

在 <http://www.gcndevelopment.com/gnuradio/> 上下载最新的 GNURadio 安装包。

GNURadio Win64 Binaries

[Download Latest Installer](#)

[Download All Other Binaries](#)

[Documentation](#), [Known Issues](#) / [Change Log](#)

Background:

GNU Radio is a powerful SDR tool, however it's biggest limitation has always been that it is a challenge to install, even on Linux systems, because of the large number of shared dependencies. If you are a daily Windows user there is an even larger problem because most of these dependencies are also very Linux focused and difficult to build/install on windows for the average Windows users who is accustomed to simply double-clicking "setup.exe". Using a Linux VM is problematic with SDR because of the high demands that are placed on the USB interface. This leaves the user with dual booting, limiting them to one environment or the other. So the goal of this project was to build an optimized GNU Radio binary using the most recent compiler for all dependencies, and packaging the result in an easy-to-use .msi package.

图 3.6

下载后得到的文件：



图 3.7

双击安装程序，按照提示安装。

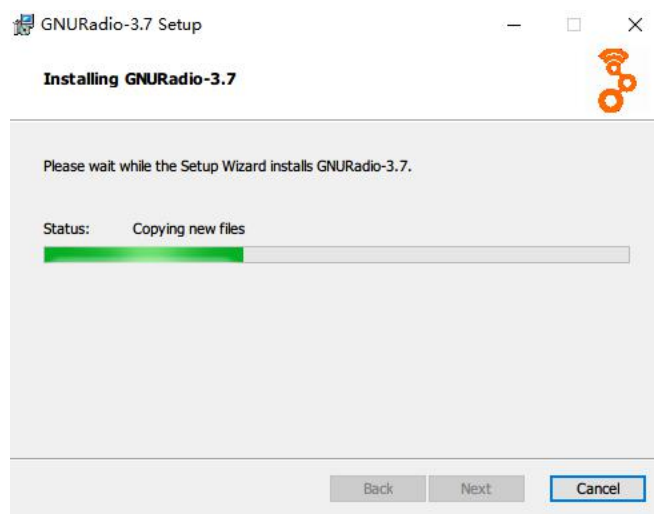


图 3.8 安装过程

安装完成后尽量关闭当前所运行的程序，然后进入安装目录的 bin 目录中（例如 C:\Program Files\GNURadio-3.7\bin），双击 volk_profile.exe，运行矢量计算优化库安装程序。安装中请不要用计算机做其它事，以免影响运算速度。静待程序退出，这代表安装完成。至此 GNURadio 环境安装完成。



```
C:\Program Files\GNURadio-3.7\bin\volk_profile.exe
a_avx2 completed in 51ms
Best aligned arch: a_avx2
Best unaligned arch: u_avx2
RUN_VOLK_TESTS: volk_32u_byteswappuppet_32u(131071, 1987)
generic completed in 211ms
u_sse2 completed in 207ms
a_sse2 completed in 199ms
u_avx2 completed in 132ms
a_avx2 completed in 138ms
Best aligned arch: u_avx2
Best unaligned arch: u_avx2
RUN_VOLK_TESTS: volk_32u_popcntpuppet_32u(131071, 1987)
generic completed in 201ms
a_sse4_2 completed in 283ms
Best aligned arch: generic
Best unaligned arch: generic
RUN_VOLK_TESTS: volk_64u_byteswappuppet_64u(131071, 1987)
generic completed in 481ms
u_sse2 completed in 391ms
a_sse2 completed in 415ms
u_ssse3 completed in 312ms
a_ssse3 completed in 293ms
u_avx2 completed in 292ms
a_avx2 completed in 268ms
Best aligned arch: a_avx2
Best unaligned arch: u_avx2
RUN_VOLK_TESTS: volk_32fc_s32fc_rotatorpuppet_32fc(131071, 1987)
generic completed in 1263ms
a_sse4_1 completed in 425ms
```

图 3.9 安装矢量运算优化库

GNURadioWin64 软件包在设计上与计算机中的其它程序独立，这意味着安装程序不修改系统的环境变量。因为 Python 需要依靠环境变量来解析依赖项，所以要运行基于 GNURadio 的 Python 程序，需要运行开始菜单/GNURadio 中的 GNURadio Command Prompt，然后在这个 shell 环境中运行需要的程序。GNURadio Companion 是图形化编程工具，Gqrx 是基于 GNURadio 的一个 SDR 实用工具集，Spectrum Scanner 是频谱仪。



图 3.10 GNURadio 的开始菜单

现在运行 GNURadio Command Prompt，将会设定好一个独立的环境，然后等待输入命令。



图 3.11

输入 `cd C:\Program Files\GNURadio-3.7\share\gnuradio\examples\audio` (视安装路径决定)

输入 `dial_tone.py`

这将会运行一个 GNURadio 程序。命令行将如下显示，并播放一段正弦波音频。按 **Ctrl+C** 组合键中止程序。



图 3.12

提示：如果这一步成功，说明之前的安装步骤成功。如果需要输入 `python dial_tone.py`，才能运行程序，那可能是 `python` 环境变量设置的问题。其它问题请参照之前的安装步骤查错。

运行开始菜单中的 GNURadioCompanion，(GRC) 将打开一个图形界面，如图 3.13。在这个界面中可以放置信号处理模块，设置模块属性，连接模块，保存成 `grc` 流图文件以供修改。运行时 GRC 会将 `grc` 流图生成一个 `python` 程序，然后运行该程序。GRC 的使用下面会详细说明。

GNURadio 安装目录下有多个文件夹。`/bin` 文件夹下有可执行程序（这里附带了一个 Zadig）。`/gr-python27` 文件夹下为 GNURadio 自带的一个 `python` 环境，因此如果做好相应设置，那么不安装 `python` 也可以。

`/share/gnuradio/examples` 目录下有大量示例 `python` 程序或 `grc` 流图，可以用于学习和测试，对学习 GNURadio 很有帮助。`dial_tone.py` 就属于这些示例程序。

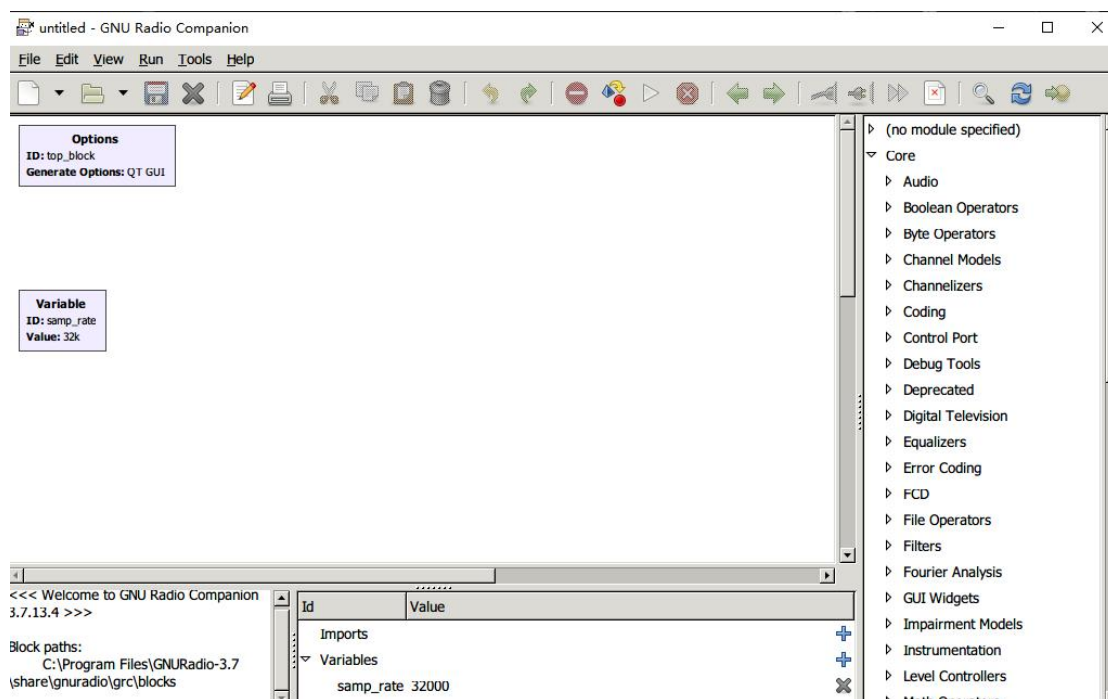


图 3.13 GRC 界面

3.3 GNURadio Companion 流图编程介绍

GNURadio 提供了一个非常简单易用的图形化界面工具 GRC (GNURadio Companion)。安装好 GNURadio 后，在开始菜单/GNURadio 中可以找到这个工具。基于 GRC 流图的编程包括：选取模块，设置参数，连接模块，执行流程。

如图 3.13，界面的左下角为状态栏，显示流图运行的相关信息和错误信息；中下为变量表，显示在当前流图中定义的变量；右侧为模块工具箱，用户可以在分类的模块列表中选择需要的模块加入流图；中部为模块设计界面，处理模块的摆放、参数和模块间的连接关系。

现在我们用 GRC 创建一个流图，用于输出音频，左、右声道各输出 350Hz 和 440Hz 的正弦波。

点击右侧工具箱中的 SignalSource 后，设计区将会出现一个 SignalSource 模块(图 3.14)。双击模块打开参数配置对话框，配置模块参数。配置好的参数会显示在模块上，并展示模块的接口。

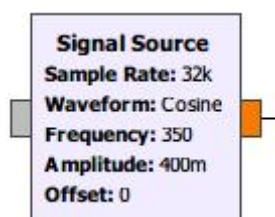


图 3.14 信号源模块

从工具箱中取出更多模块，用鼠标连接接口，得到如图 3.15 的流图：

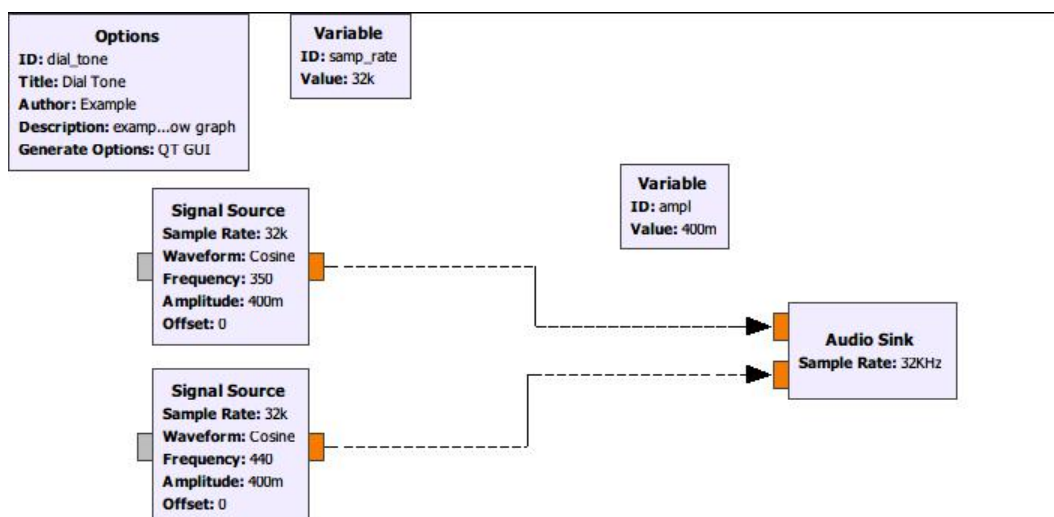


图 3.15 音频输出程序的流图

与代码编程相同，grc 流图也需要信源，信宿，信号处理过程这三个部分。搭建完成后，运行流图，GRC 将会根据流图，在与.grc 文件相同的目录下生成一个 Python 程序，然后执行。设计流图过程中的定义模块，定义参数，连接操作在代码体现上与代码编程是一致的。GRC 将此过程简化，从而使开发变得更加方便。

3.4 GNURadio Companion 的常用模块介绍

3.4.1 信源模块

Signal Source 用于产生正弦波、方波、三角波等简单波形。File Source 用于输出文件数据。Wav File Source 用于音频文件输出。Vector Source 用于输出向量，可以用于生成如 (1,0,1,1……) 之类的向量序列。Constant Source 用于生成常量信号。Audio Source 用于输出从话筒接收的信息。Random Source 输出范围内的随机数字序列，例如按图 3.16 中，令 Minimum=0，Maximum=2，则会输出二进制随机数字序列。

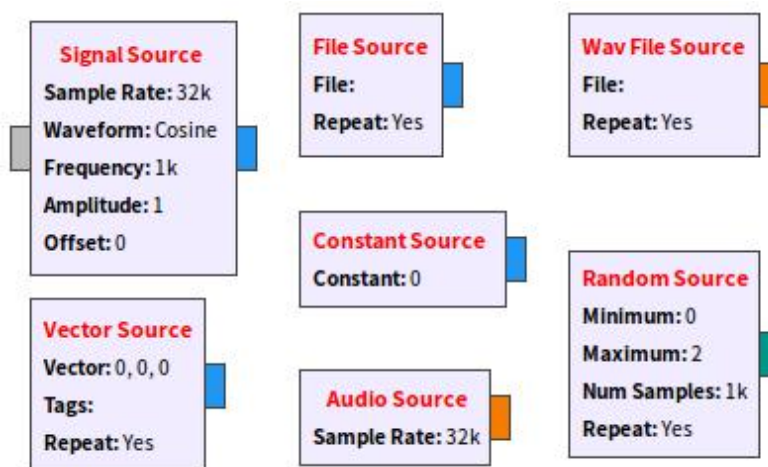


图 3.16 GRC 中的信源模块

3.4.2 运算模块

包含加减乘除、积分、求极值，布尔运算等。

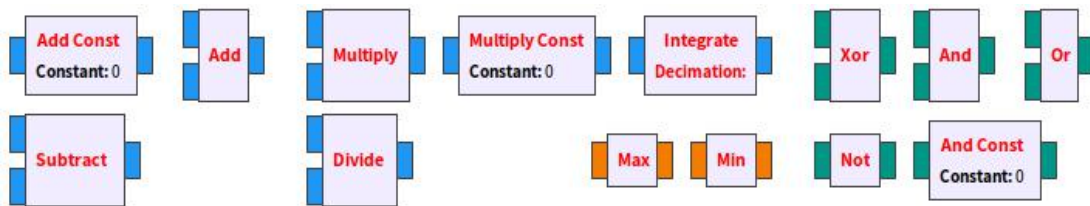


图 3.17 GRC 中的运算模块

3.4.3 Rational Resampler 模块



图 3.18 Rational Resampler 模块

该模块通过多相 FIR 滤波器实现抽样速率变换，相当于将一个抽取器和一个内插器级联起来，从而实现对抽样速率的调整，即重采样。Rational Resampler 模块输入抽样速率和输出抽样速率应满足：

$$\text{输出抽样速率} = \text{输入抽样速率} * \frac{\text{Interpolation}}{\text{Decimation}}$$

3.4.4 Throttle 模块



图 3.19 Throttle 模块

该模块为节流模块，限制了通过该模块的数据速率，避免 CPU 将所有资源都耗费在数据传输和处理上。在不连接 SDR 硬件，仅用 GNURadio 环境做信号处理实验时，需要加上此模块，以免 CPU 资源被耗尽。在连接 SDR 硬件时，不要加此模块。

3.4.5 UHD:USRP Source/Sink 模块

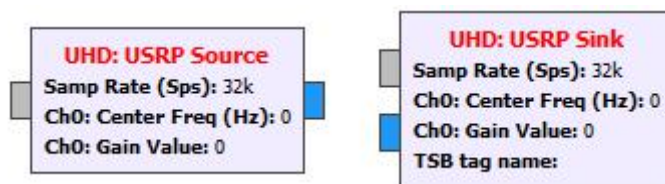


图 3.20 UHD:USRP Source/Sink 模块

这两个模块是 USRP 硬件的接口。Source 负责输入 SDR 硬件的信号，Sink 负责将信号输出给 SDR 硬件。这两个模块中参数的含义是：

Sample Rate (sps): 采样速率。

Center Freq (Hz): 射频信号发送/接收的中心频率。

Gain Value: 增益。

3.4.6 Packet Encoder 模块

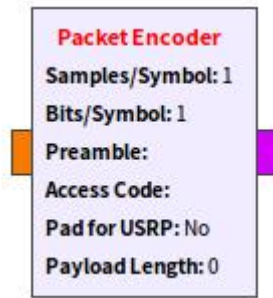


图 3.21 Packet Encoder 模块

该模块为分组编码模块，能将数字信号序列分组并编码，通常配合调制解调模块使用。其中 Samples/Symbol 为每符号采样数。Bits/Symbol 参数的设置根据调制方式来确定，确定关系如图 3.22 所示。

Bits/Symbol should be set accordingly:
 gmsk -> 1
 dbpsk -> 1
 dqpsk -> 2
 d8psk -> 3
 qam8 -> 3
 qam16 -> 4
 qam64 -> 6
 qam256 -> 8

图 3.22 Packet Encoder 模块的 Bits/Symbol 参数设置

3.4.7 类型转换模块、调制解调模块和仪器模块（频谱、时域波形、星座图等等）

▼ Type Converters	▼ Modulators	▼ Instrumentation
Char To Float	AM Demod	▼ QT
Char To Short	FM Demod	QT GUI Bercurve Sink
Complex to Arg	Frequency Mod	QT GUI Constellation Sink
Complex To Float	Phase Mod	QT GUI Frequency Sink
Complex To IChar	Quadrature Demod	QT GUI Histogram Sink
Complex to Imag	Constellation Modulator	QT GUI Number Sink
Complex To IShort	Continuous Phase Modulation	QT GUI Sink
Complex to Mag	DPSK Demod	QT GUI Time Raster Sink
Complex to Mag^2	DPSK Mod	QT GUI Time Sink
Complex To Mag Phase	GFSK Demod	QT GUI Vector Sink
Complex To Real	GFSK Mod	QT GUI Waterfall Sink
Float To Char	GMSK Demod	▼ WX
Float To Complex	GMSK Mod	WX GUI Constellation Sink
Float To Int	GMSK Modulator	WX GUI FFT Sink
Float To Short	Modulate Vector	WX GUI Histo Sink
Float To UChar	PSK Demod	WX GUI Number Sink
IChar To Complex	PSK Mod	WX GUI Scope Sink
Int To Float	QAM Demod	WX GUI Terminal Sink
IShort To Complex	QAM Mod	WX GUI Waterfall Sink
Magnitude and Phase To Complex		
Short To Char		
Short To Float		
UChar To Float		

图 3.23 GRC 中的类型转换模块、调制解调模块和仪器模块

在图 3.23 右侧列出的仪器模块中，有 QtGUI 和 wx GUI 两种。Qt 和 wxWidgets (wx) 都是跨平台的 C++ GUI 工具库。在 GNU Radio 中，Qt 指基于 Qt 的 Python GUI 图形库 PyQt，而 wx 指基于 wxWidgets 的 Python GUI 图形库 wxPython。一个 GRC 流程图程序只能在 Qt 和 wx 中选择一种 GUI 库使用。可以从实际需求出发来选择合适的库。

4. 基于 USRP 硬件平台和 GNURadio 的实验

4.1 USRP 介绍

USRP 可以支持两路并行的发送或者接收。RF 前端, 完成射频信号和不同频带信号之间的转换, 涵盖从直流到 6GHz 的整个范围, 这包括了从调幅广播到超过 Wi-Fi 的所有频率。USRP-B210 是一款应用较广的 SDR 设备, 其系统架构如图 4-1 所示。

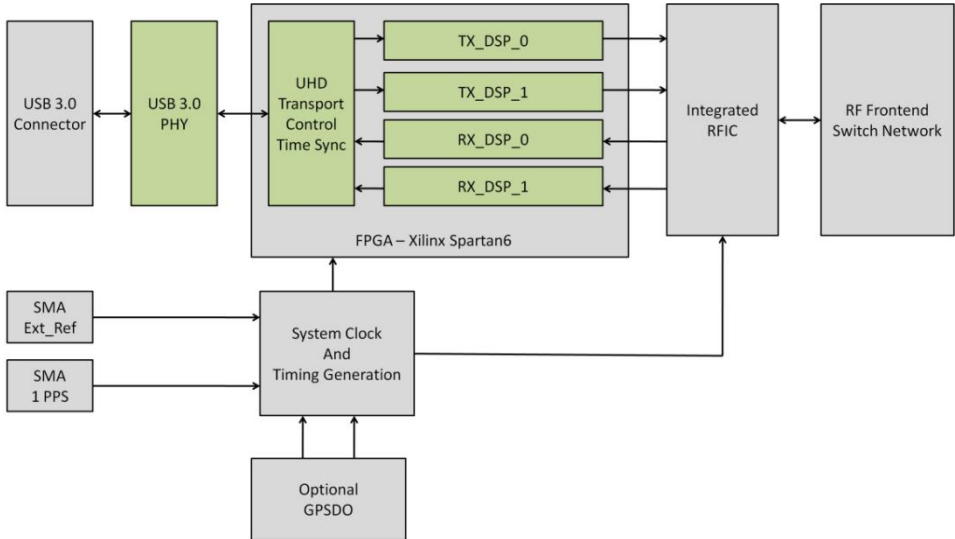


图 4-1 USRP-B210 设备系统架构

USRP 的真正价值是它能使工程师和设计师以低预算和最少的精力进行创造。为数不少的开发者和用户贡献了大量的代码库, 并为软件和硬件提供了许多实际应用。灵活的硬件、开源的软件和拥有经验丰富用户社区群的强强联合, 使它成为您软件无线电开发与应用的理想外设。

USRP-B210 主要特性:

- 高度集成, 两路收发, 射频范围从 70 MHz-6 GHz;
- 支持高速的 USB 3.0 连接;
- 开源的 UHD 支持多种框架;
- 用户可编写的 FPGA, 一个完全集成的混合信号基带直接转换收发器, 可提供到 56MHz 的带宽。

USRP- B210 技术参数

参数类别	数值	单位	参数类别	数值	单位
输入\输出			单通道射频性能参数		
直流电压输入	6	V	单边带信号/镜像 抑制	-35/50	dBc
转换模块参数			3.5GHz	1.0	degRMS
ADC 采样速率(最大)	61.44	MS/s	6GHz	1.5	degRMS
ADC 分辨率	12	bits	输出功率	>10	dBm
ADC SFDR	78	dBc	输入三阶截取点	-20	dBm

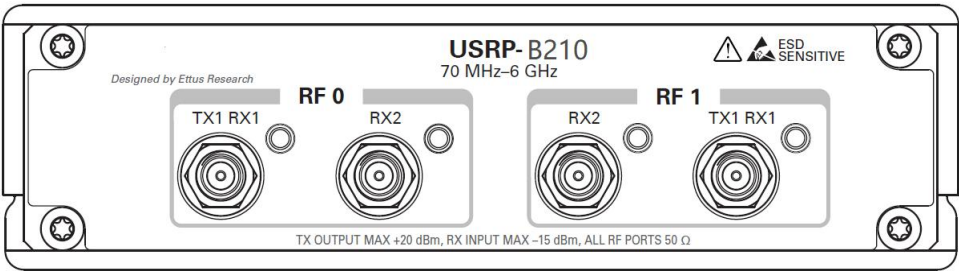
DAC 采样速率	61.44	MS/s
DAC 分辨率	12	bits
与主机最大速率(16b)	61.44	MS/s
本振精度	± 2.0	ppm
GPS 未锁定 TCXO 精度	± 75	ppb
GPS 锁定 TCXO 精度	< 1	ppb
噪声系数	< 8	dB
物理属性		
尺寸	9.7x15.5x1.5	cm
重量	350	g

B210 采用了射频前端集成芯片 AD9361 设计，AD9361 支持射频端直接上下变频，可实时提供 56MHz 的带宽。采用 AD9361 的两路信号链，提供相干 MIMO 能力。主机通过 USB 3.0 对型号为 Spartan6XC6SLX150 的 FPGA 一系列操作来实现对 AD9361 中信号的处理和控制。B210 最大提供 61.44MS/s 的吞吐量，利用 UHD 的 API 与 PC 相连。

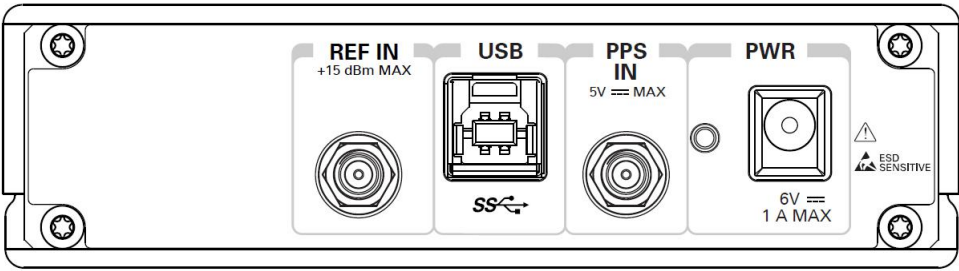
利用 B210 可以实现十分广泛的应用，包括广播，手机，GPS，WiFi，ISM，FM 和 TV 信号等等。用户可以利用 GNU Radio，LabVIEW 等软件很快的进行一些无线电的开发，或者参加一些开源的软件无线电项目。基于 UHD 的支持，允许用户重用现有的设计，例如 HSDR 和 OpenBTS 两个开源的应用均支持。

USRP-B210 前后面板布局，如图 2-3 所示。前面板中，有 RF0 和 RF1 两个通道。每通道中的 TX1/RX1 为射频输入/输出接口，负载阻抗 50 欧姆，单端口；RX2 为射频输入接口，负载阻抗 50 欧姆，单端口。相应的指示灯可指示各端口的工作状态。

后面板中的 REF IN 是本振参考信号的输入端，USB 接口用于连接电脑，PPS IN 接口可连接外部的秒脉冲（PPS）参考信号，PWR 接口在需要时可连接 6V/3A 的电源适配器（功耗较小时，直接由 USB 供电）。



前面板



后面板

图 4-2 USRP 面板结构图

4.2 FM 信号的发送和接收实验

下面介绍在 GNURadio Companion 软件平台下，基于 USRP B210 硬件设备的 FM 发送和接收实验。GNURadio 内置了许多信号处理模块，FM 信号的发送和接收可以使用内置的模块实现。

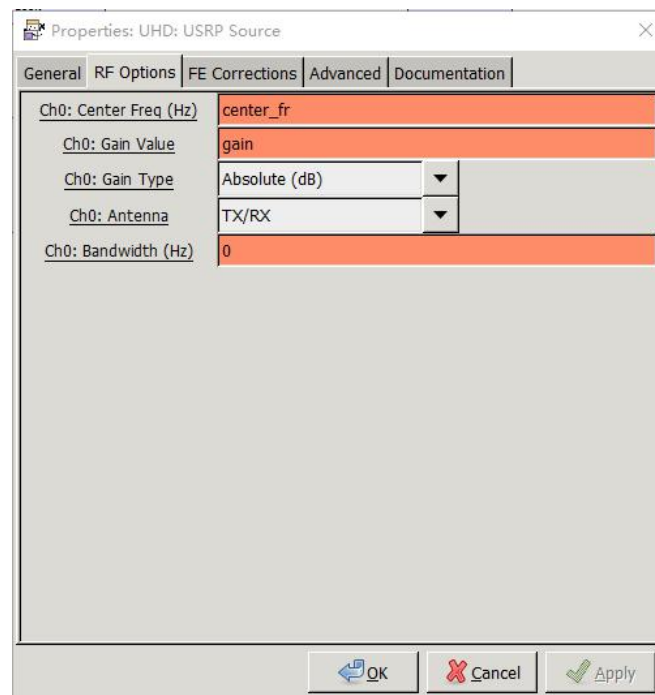
4.2.1 FM 接收实验

在这个实验中，我们使用 USRP 设备做 FM 收音机收听 FM 广播。实验步骤如下：

(1) 打开 GNURadio Companion 软件

(2) 新建一个流程图页面，将 Options 属性的 ID 改为 LAB1，将 Variable 的 Value 改为 250e3。Options 块定义该 grc 流图的属性。ID 为标题，也决定了生成的 py 文件的文件名。生成选项 Generate Options 可以选择 QTGUI/WXGUI/NoGUI，对应使用的 GUI 库的类型。在这里选择某个 GUI 库后，在流图中的 GUI 模块就只能使用该库中的模块。

(3) 添加 UHD USRP Source 到画布上，其参数设置如下：



(4) 添加两个 QT GUI Range 模块，并将其属性改为如下所示。QTGUIRange 模块是一个可在运行时调整的变量，ID 是变量名。程序运行时将会出现一个滑动条，滑动游标或者直接设置数值可以在运行时实时改变变量值。

Properties: QT GUI Range

General	Advanced	Documentation
ID	gain	
Label		
Type	Float	
Default Value	40	
Start	0	
Stop	100	
Step	1	
Widget	Counter + Slider	
Minimum Length	200	
GUI Hint	0,0	

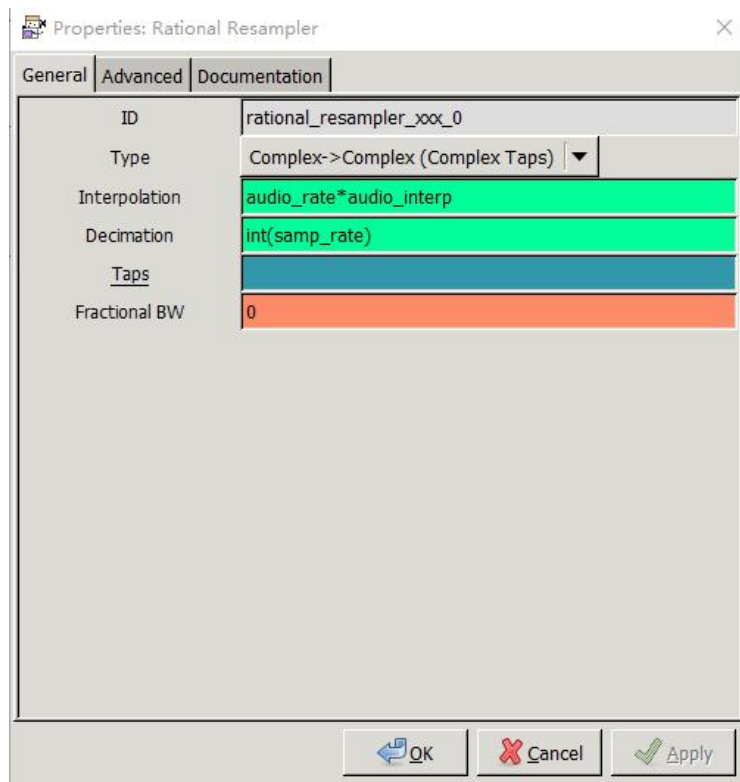
OK Cancel Apply

Properties: QT GUI Range

General	Advanced	Documentation
ID	center_fr	
Label		
Type	Float	
Default Value	98e6	
Start	0	
Stop	108e6	
Step	100000	
Widget	Counter + Slider	
Minimum Length	200	
GUI Hint	1,0	

OK Cancel Apply

- (5) 添加一个 QT GUI Sink 模块，用于观测接收信号的信息。
- (6) 添加一个 Rational Resampler 模块，用于采样率变换。Rational Resampler 模块的 Interpolation 属性为差值倍数。其插值倍数为 $\text{audio_rate} \times \text{audio_interp}$ 。



- (7) 添加一个 Variable 模块，其 ID 设置为 audio_interp，其 Value 值设置为 4。
- (8) 添加一个 Variable 模块，其 ID 设置为 audio_rate，其 Value 值设置为 48000。
- (9) 添加一个 WBFM Receive 模块，用来实现 FM 接收。其参数设置参考图 4-3。
- (10) 添加一个 Audio Sink 模块，用来输出声音。将其 Sample Rate 设置为 audio_rate。
- (11) 按照图 4-3 FM 接收整体程序流程图连接好后，保存为 LAB1.grc。
- (12) 将 USRP-B210 设备连接到 PC 端，然后运行程序。通过调节 center_fr 的值来调节频道，接收 FM 信号。图 4-4 是流程结果图。

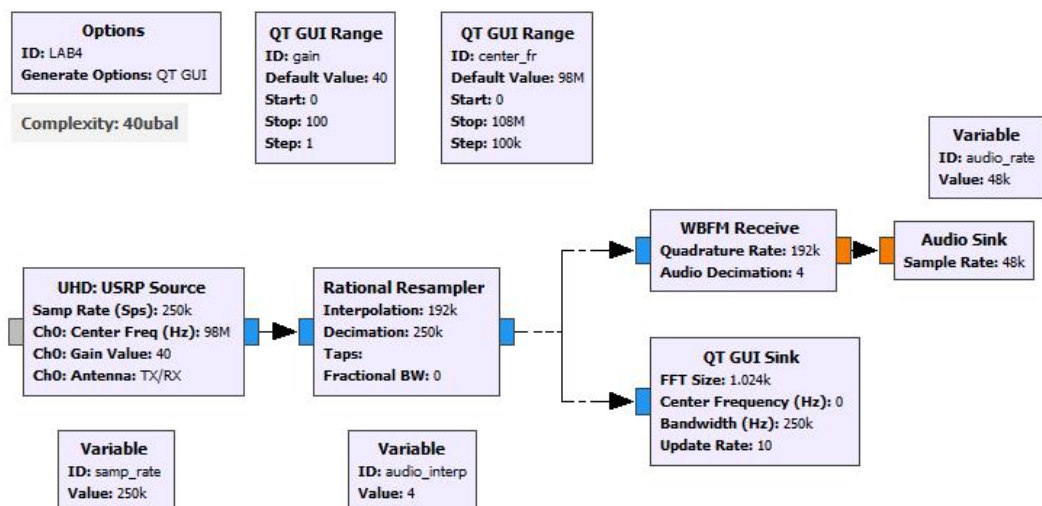


图 4-3 FM 接收信号流程图

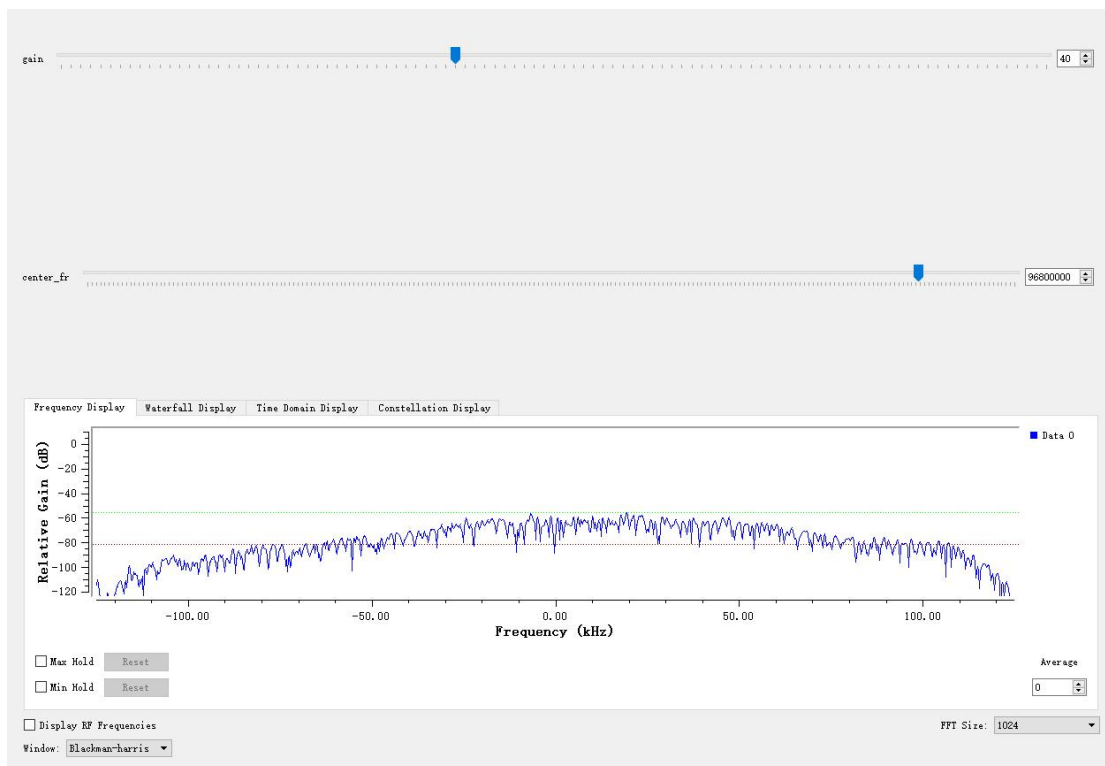


图 4-4 流程结果图

4.2.2 FM 发送实验

图 4-5 是 FM 发射信号流图。先搭建好 FM 发射信号流图，然后用搭建的 FM 发射机发送音频信号，在普通 FM 收音机中收听到音乐，并且用另一台 SDR 硬件接收，通过 FM 接收机流图在计算机上听到音乐。

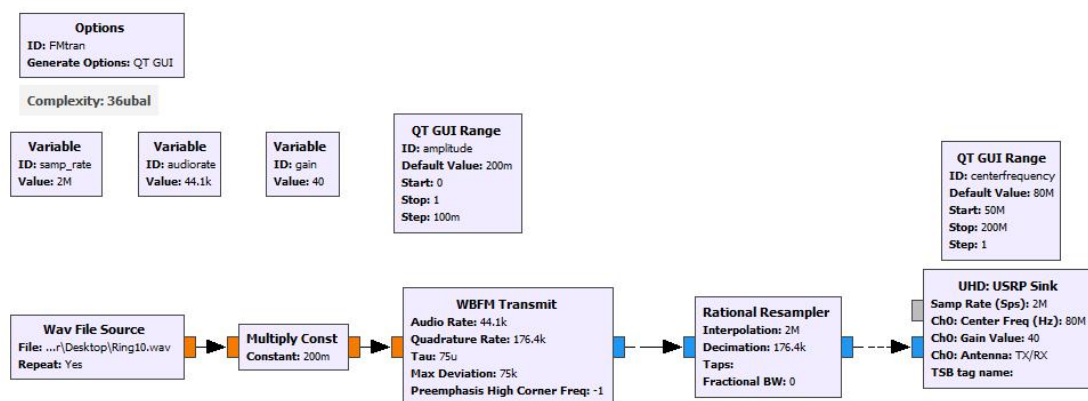


图 4-5 FM 发射信号流图

一个完整的 grc 流图需要至少一个信源和信宿。这里的信源采用 wav 文件，该模块会将声音内容转换为信号流（如 wav 文件的采样率为 44.1kHz）。信宿 UHD:USRP Sink 为 USRP 硬件接口，可以设置中心频率和增益。设置的中心频率即为 FM 广播的频段，可以自由调整。

FM 调制方式通过 WBFM Transmit 模块实现。该模块共有 5 个参数：Audio Rate 表示音频输入流的采样速率；Quadrature Rate 表示输出流的采样速率，与 Audio Rate 之间呈整数倍关系；Tau 代表 pre-emphasis time constant，即预加重时间常数；Max Deviation 代表最大频偏；Pre-emphasis High Corner Freq 代表预加重拐角频率，当该参数小于 0 时，默认值被设置

为 $0.925 \times \text{quad_rate}/2.0$ 。

通过 WBFM Transmit 模块后，数据的采样速率为 176.4kHz，通过 Rational Resampler 重采样模块调整采样速率至 2MHz。

流图绘制完毕后，如果流图符合要求，GRC 界面上的运行按钮就会处于可用状态。连接 HackRF，单击运行按钮，流图就开始运行。此时可以用收音机收听广播了。

4.3 QPSK 发射机和接收机实验

本实验要求按照给出的步骤搭建 QPSK 发射机和接收机的流图。使用两台 HackRF，一台作为发射机，一台作为接收机。在发射端发送一个特定的信号比特串，观察接收端是否解调得到了正确的比特串。记录信号经过各个模块的星座图。

4.3.1 实验原理

PSK 在一个符号区间内用载波的相位来表示一个符号。设符号区间为 T ，脉冲宽度为 T 的信号为 $g(t)$ ，符号用相位表示为 ϕ ，那么已调信号

$$s(t) = \sum_n \cos(\omega t + \phi_n) g(t - nT)$$

在一个符号区间内，将 $\cos(\omega t + \phi_n)$ 展开成 IQ 调制的形式

$$\cos(\omega t + \phi_n) = a(n) \cos(\omega t) - b(n) \sin(\omega t)$$

载波的相位完全由 $a(n)$ 和 $b(n)$ 决定。在 SDR 系统的 IQ 调制的框架下，计算机只要输出给 SDR 硬件一个 $a(n) + ib(n)$ 的复信号即可。

可以在复平面上表示复包络信号 $a(n) + ib(n)$ 。这样的表示方式称为星座图。星座图上点的位置决定了 PSK 波形在各个符号区间内的相位。按照星座点的分配不同，PSK 也有多种调制方式。图 4-6 展示了 2PSK、QPSK、 $\pi/4$ -QPSK 三种 PSK 调制的星座图。

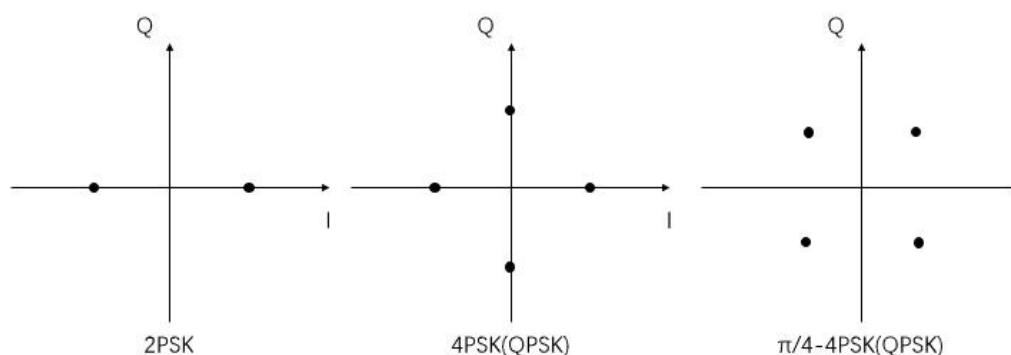


图 4-6 三种 PSK 调制方式的星座图

4.3.2 QPSK 发射机实验

我们选用 PSK 调制中常用的 QPSK ($\pi/4$ -QPSK) 调制进行实验。图 4-7 为 QPSK 发射端流图。图中采用 GNURadio 内置的星座调制器进行 QPSK 调制。

首先，信号源使用符号源。每一个符号为 8 位比特，从 0x00 到 0xFF，也就是 0~255。图中用到两个信号源，执行流图时可以启动其中一个，禁用另一个。随机源 RandomSource 设定为随机产生 0~255 的符号，向量源 VectorSource 设定为一串不断重复的符号。例如，将向量设定为 [1, 8, 3, 6, 5, 4, 7, 2]，那么输出比特串为：

00000001_00001000_00000011_00000110_00000101_00000100_00000111_00000010

比特串经过内置的星座调制器调制后，显示到星座图输出中，同时输出到 SDR 硬件中，以设定的载波频率发射（例如设为 153MHz）。

星座调制器的使用需要输入一个符号映射对象，该对象定义了符号映射到哪个星座上。

该对象必须在流图中定义。在星座调制器中，符号映射对象由 Constellation Rect. Object 定义，并将对象名设为 qpsk。那么在星座调制器中，星座一栏就可以填写 qpsk，也就是应用对象 qpsk 中定义的规则。星座调制器可以在 Differential Encoding 选项设定为开启/关闭差分编码；在 Samples/Symbol 选项设置每个符号的采样数。

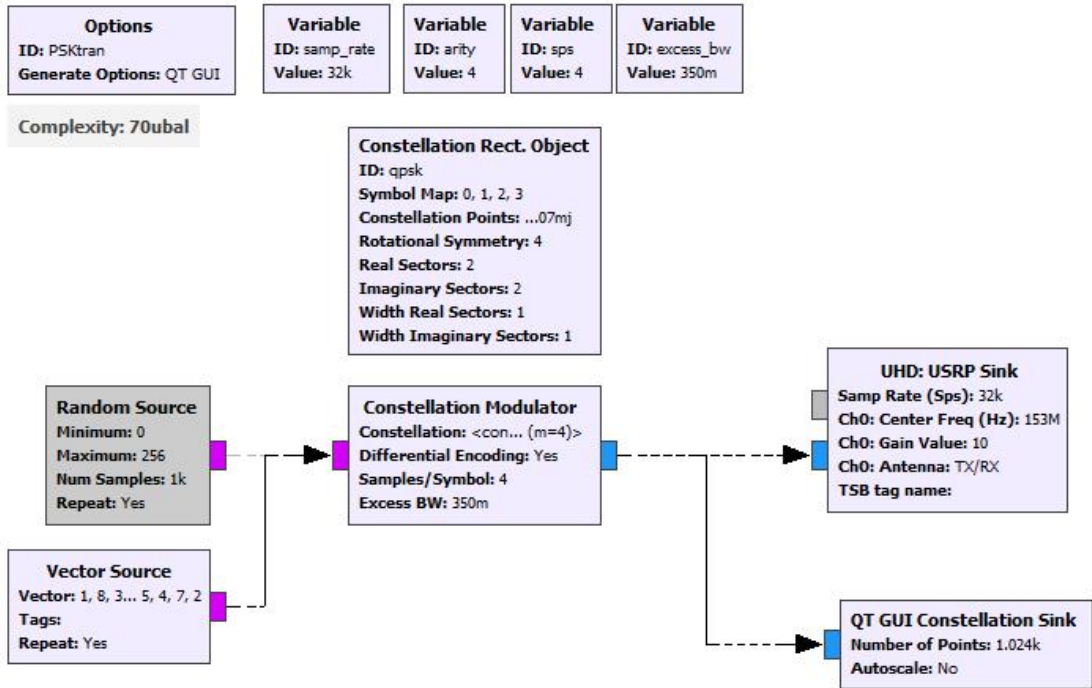
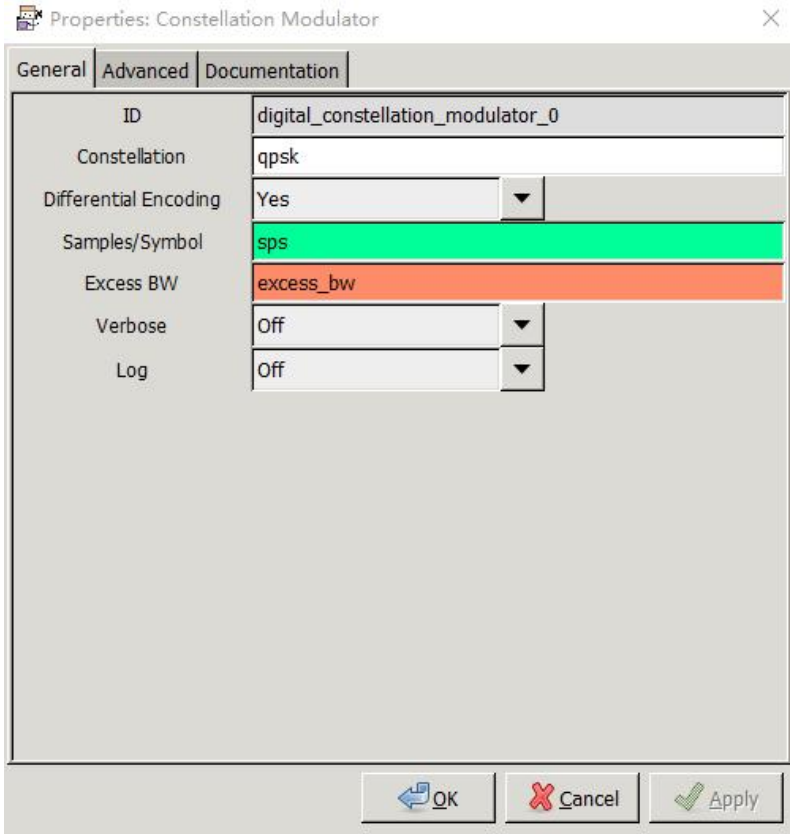


图 4-7 QPSK 发射机流图

以下是一些模块的参数设置：



Properties: Vector Source

General | Advanced | Documentation

ID	blocks_vector_source_x_0
Output Type	Byte
Vector	(1,8,3,6,5,4,7,2)
Tags	
Repeat	Yes
Vec Length	1

OK Cancel Apply

Properties: Constellation Rect. Object

General | Advanced | Documentation

ID	qpsk
Symbol Map	[0, 1, 2, 3]
Constellation Points	[0.707+0.707j, -0.707+0.707j, 0.707-0.707j, -0.707-0.707j]
Rotational Symmetry	4
Real Sectors	2
Imaginary Sectors	2
Width Real Sectors	1
Width Imaginary Sectors	1
Soft bits precision	8
Soft Decisions LUT	None

OK Cancel Apply

Symbol Map 和 Constellation Points 定义映射关系，其值都是一个向量。Symbol Map

定义要映射的符号，Constellation Points 需要一个与 Symbol Map 长度相同的向量，向量中的值用复数表示，两者共同定义了相应的符号映射在哪个星座点。

这样定义完以后，信号流的行为是这样的：星座调制器将输入的比特流分为两个比特一组，分别用[0, 1, 2, 3]表示，然后按照给定的映射关系，映射为相应的复数值，也就是 IQ 两路信号。星座调制器的输出时域波形如图 4-8 所示：

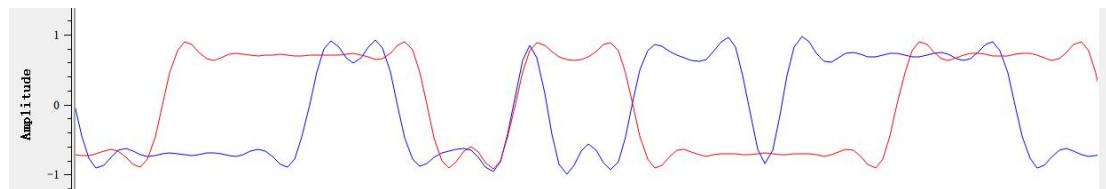


图 4-8 星座调制器的输出的时域波形

例如，当符号为 01 的时候，星座调制器将会将其映射为 $-0.707 + 0.707j$ ，I 路输出会保持在 -0.707 一段时间，Q 路输出会保持在 $+0.707$ 一段时间。因此，输出的基带信号在理想情况下是一个方波。但是，因为输出的是带限信号，输出的结果不是理想的方波，而是如图 4-8 的波形。星座图作为时域波形在每一时刻的采样，然后将 I、Q 两路作为坐标轴描绘的图像，也会受到带限的影响，从理想的 4 个点变为如图 4-9 中的样子。

如之前介绍 IQ 调制的部分所说，最后 SDR 硬件的 IQ 调制体系将会把 IQ 两路的信号幅度转变为相位来传递信息，因此在空中发射的是 PSK 信号。

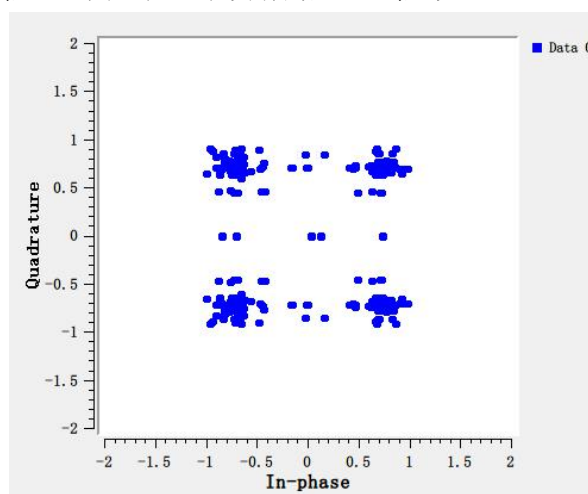


图 4-9 QPSK 星座图

4.3.3 QPSK 接收机实验

图 4-10 为 QPSK 接收机端流程图。

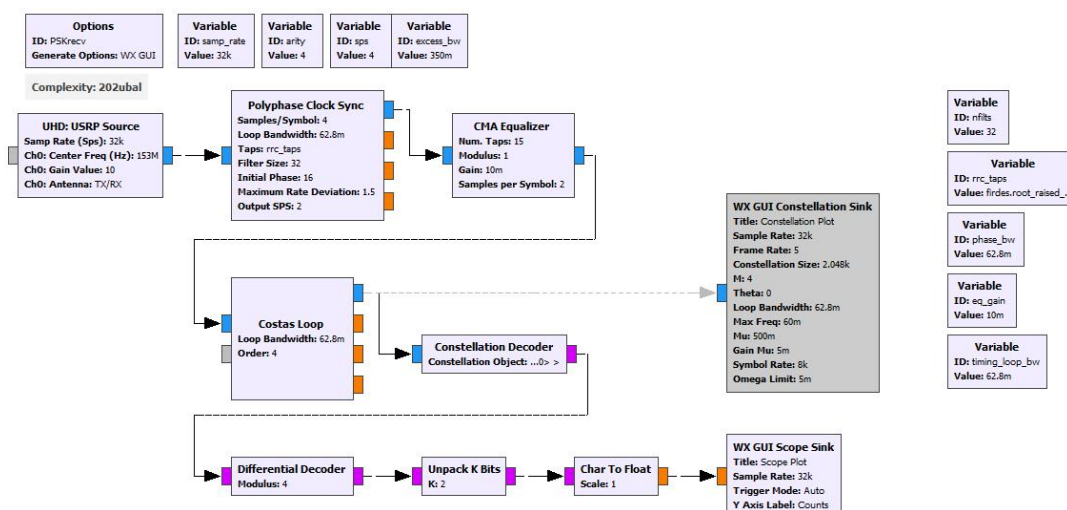
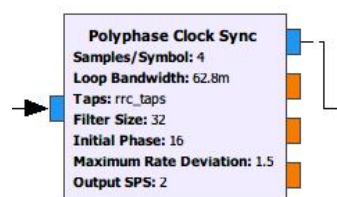


图 4-10 QPSK 接收机流程图

接收端信号从 USRP-B210 接入后，经过 Polyphase Clock Sync 进行时钟同步，矫正接收机与发射机采样频率偏差和采样时间偏差造成的码间干扰。Polyphase Clock Sync 做三件事：时钟恢复，然后用匹配的滤波器消除码间干扰，最后对信号下采样，减少 Samples/Symbol。该模块的参数如下所示：

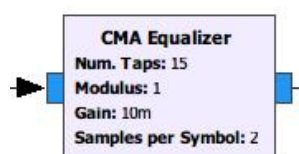


在发射机中我们设置的 Sample/Symbol 为 4，这里用一样的设置，并设输出的 SPS 值为 2。rrc_taps 定义一个根升余弦滤波器，其内容设置为：

`firdes.root_raised_cosine(nfilts, nfilts, 1.0/float(sps), 0.35, 11*sps*nfilts)`

其中 nfilts=32, sps=4。

CMA Equalizer 是一个盲均衡器，用于对抗信道对信号传输的影响，例如多径效应。并处理 SPS 为 2 的信号，式输出的每一个采样都表示一个符号。其参数设置如下：



星座调制器的输出星座图为图 4-11。图 4-12~4-14 分别列出了信号经过信道，时钟同步，均衡器后的星座图。

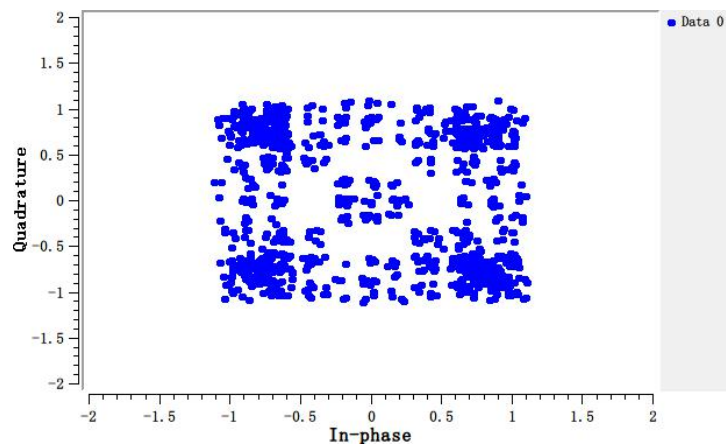


图 4-11 QPSK 发射端星座图

经过信道:

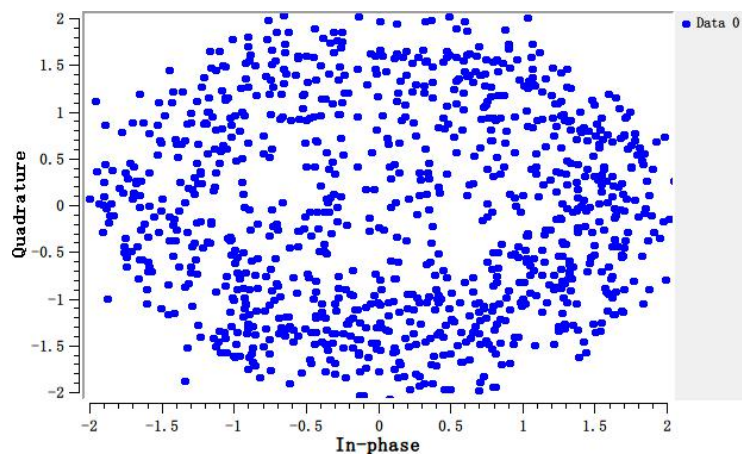


图 4-12 经过信道后的星座图

经过时钟同步:

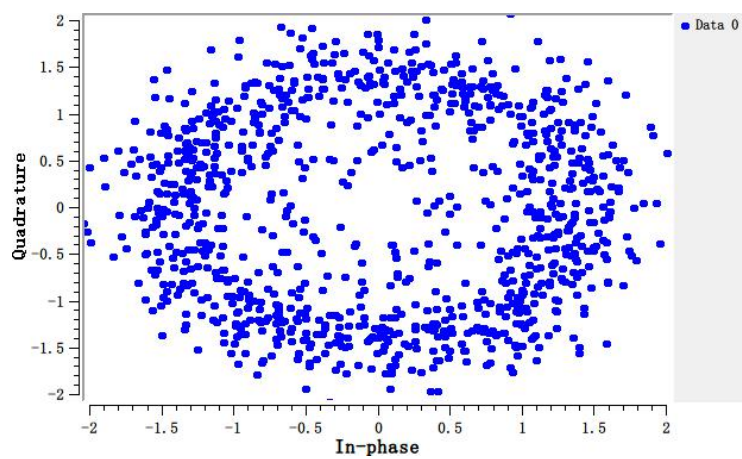


图 4-13 经过时钟同步后的星座图

经过均衡器:

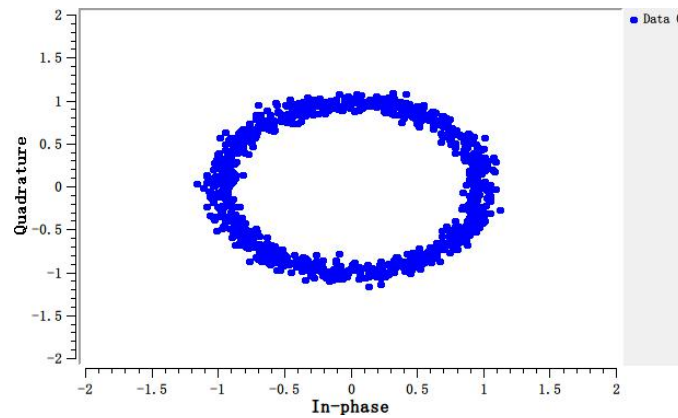


图 4-14 经过均衡器后的星座图

如图 4-14。我们希望的 QPSK 星座图是 4 个点，而这里的星座图是一个环。根据 IQ 调制解调的原理，IQ 解调时需要乘上一个和接收机相位、频率完全一致的载波。因为接收机和发射机的相位，频率是不同步的，不可能完全一致，总会有一些微小的误差，这就导致了星座图变为了图 4-14 的样子。

如果 IQ 解调仅有相位偏差，那么对于 QPSK 的 4 个星座点，每一个星座点都会旋转一个固定的角度，星座图就会旋转；如果频率存在偏差，那么这意味着星座图上的每一个星座点都会随着时间产生一个不同的相位偏差，这意味着每一个星座点都会旋转一个不同的角度，导致星座图变成一个环。

这是通信过程中常见的一种同步问题：载波同步。Costas 锁相环是用来解决载波同步问题的常用手段，可以对载波的相位和频率进行矫正。通常发射机的载波和接收机的载波频率无法保证完全一致，Costas 环可以在一定频率偏差范围内捕捉正确的载波频率。

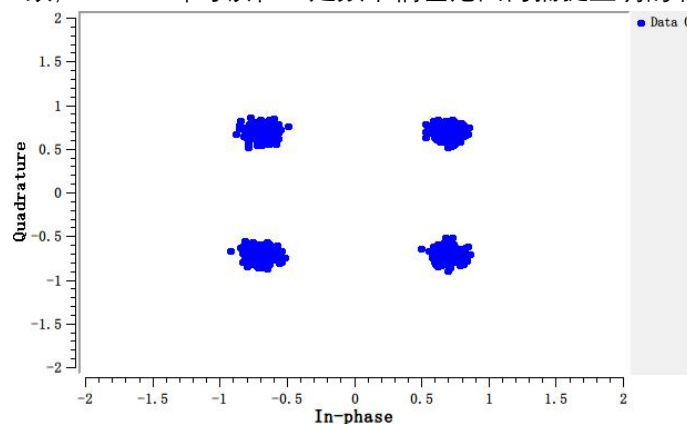


图 4-15 经过 Costas 环后的星座图

接下来需要对星座图中的每一个点进行判决，确定其所处的位置，得到其代表的符号。

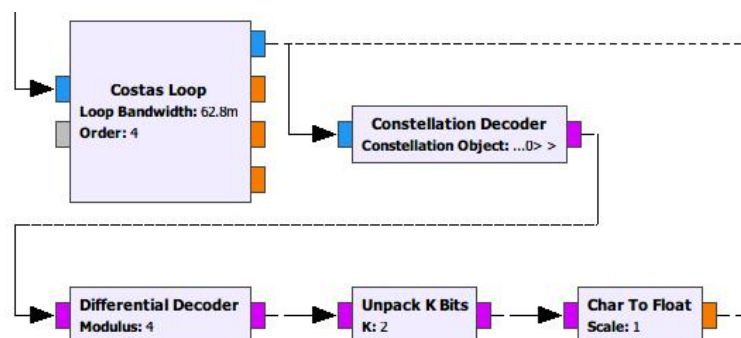


图 4-16 判决，解包部分

Constellation Decode 模块用于对每一个样本（星座点）进行判决。判决的依据在参数 ConstellationObject 参数中。参数值为

digital.constellation_calcdist([1+1j, -1+1j, -1-1j, 1-1j]), ([0,1,3,2]), 4, 1).base()

这意味着将样本与向量[1+1j, -1+1j, -1-1j, 1-1j]中的 4 个值的距离比较，然后判决为最近的值。将对应的判决结果映射为[0,1,3,2]。

最后经过差分解码，将两位的符号解包，输出比特流。