Dual FM Receiver Report
ECE 447 Communications
November 1, 2024

C1C D Benjamin Cometto

Introduction

While it is common to receive one FM radio station, it is often desired to listen to multiple songs (or, more likely, advertisements) simultaneously. This need motivated the development of the dual FM receiver. This receiver can be used to monitor two FM channels at the same time, a vital feature of any radio to be used in a combat environment.

This dual FM receiver is implemented in GNU Radio, and uses the RTL-SDR software defined radio. It can listen to any any channel in the FM radio band (88 MHz to 108 MHz) simultaneously with a second channel within 1.4 MHz of the first. Each channel has individual volume controls. Additionally, when the capabilities exist, the receiver operates with stereo audio: one station is sent to the left channel and one is sent to the right channel, allowing better monitoring of disparate audio.

For access to the GNU Radio flowgraph and example audio output files, see the GitHub at https://github.com/dbcometto/ece447_dualfm.git.

1 Flowgraph Description

The GNU Radio flowgraph for the dual FM receiver can be seen in Figure 1. The first channel (station 1) is set using `freq_station1`, which can tune to any FM frequency. The second channel (station 2) is set using an offset from the first. The sample rate limits the range of this second channel to one-half of `samp_rate`, allowing an offset of ± 1.4 MHz, set in increments of 200 kHz for useability.
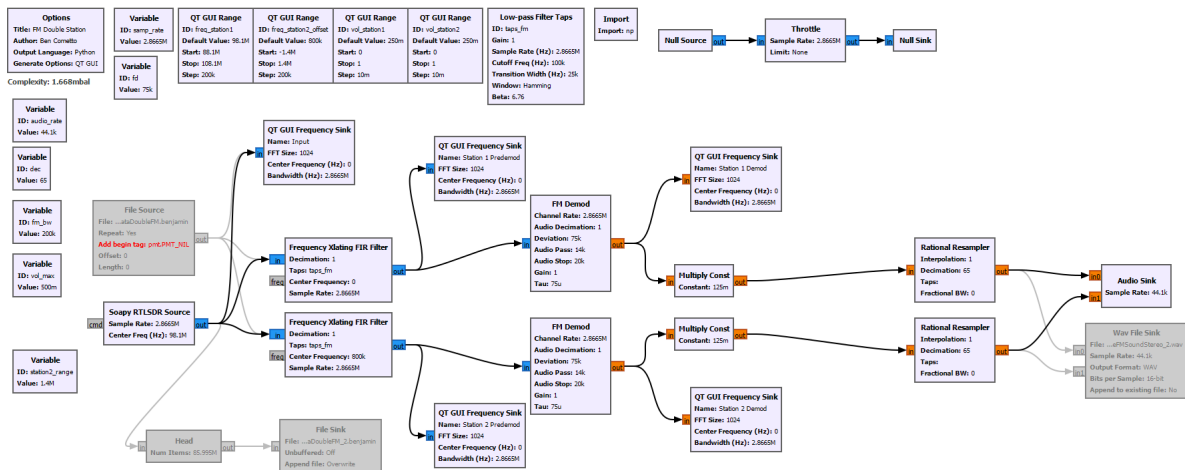

Figure 1: Dual FM Receiver Flowgraph

After receiving the raw data from the RTL-SDR, the flowgraph selects each wanted signal using a translating low pass filter. The first filter (for station 1) is centered around 0 Hz, the center frequency of the RTL-SDR source. The second filter is centered around the chosen offset.

Then, each signal is demodulated, using the standard FM frequency deviation f_d of 75 kHz. At this point, each signal is multiplied by the volume control, a constant between 0 and 0.5. Finally, the signals are decimated down to the audio rate of 44.1 kHz and played through a stereo audio sink.

Notice additionally that the flowgraph has the capability to record the raw RTL-SDR data, playback raw RTL-SDR data, and record the output audio to a `.wav` file.

2 Features

A main consideration was a desire to be able to listen to any two FM stations. However, the maximum sampling frequency f_s of the RTL-SDR is $3.2\,\text{Msps}$. This limits the ability of the second station to be at any frequency. Thus, only the first station is unconstrained, while the second is limited to be within approximately $f_s/2$ of the first station.

The system is easily controlled using the QT GUI. The GUI features a slider/entry for station 1's frequency and the offset for station 2 and volume knobs for each channel. These can be seen in Figure 2. Additionally, the GUI features several analytical graphs, including the frequency spectrums of the RTL-SDR data and each station before and after modulation.

Figure 2: The receiver's QT GUI control knobs

A problem with the current iteration of the system is the large CPU requirements. Running the RTL-SDR with $f_s = 2.8\,\text{MHz}$ requires significant CPU computing power, which can be seen in Figure 3.

Figure 3: Windows Task Manager showing GNU Radio (Python)'s CPU usage

This leads to occasional computer-wide slowdowns and audio skips/lag.

Documentation

I did my own work, using the GNU Radio wiki and a few google searches related to commercial FM radio, such as for the frequency deviation and common stations in Colorado Springs.