GRCon25

Reverse Engineering a Simple RF Remote

Learning Objectives

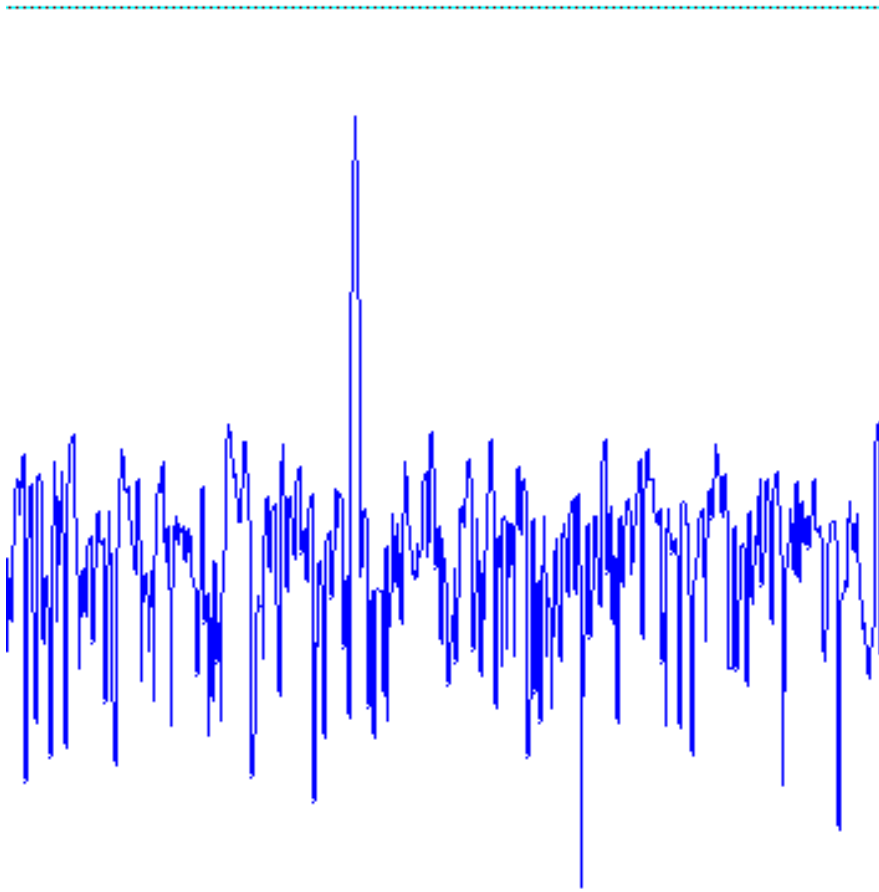
1.1 Reverse engineer a simple RF transmitter

1.2 Create a GNURadio flowgraph to operate fan (or at least generate simulated bitstream)

1.3 Get hands-on with GRC and other tools!

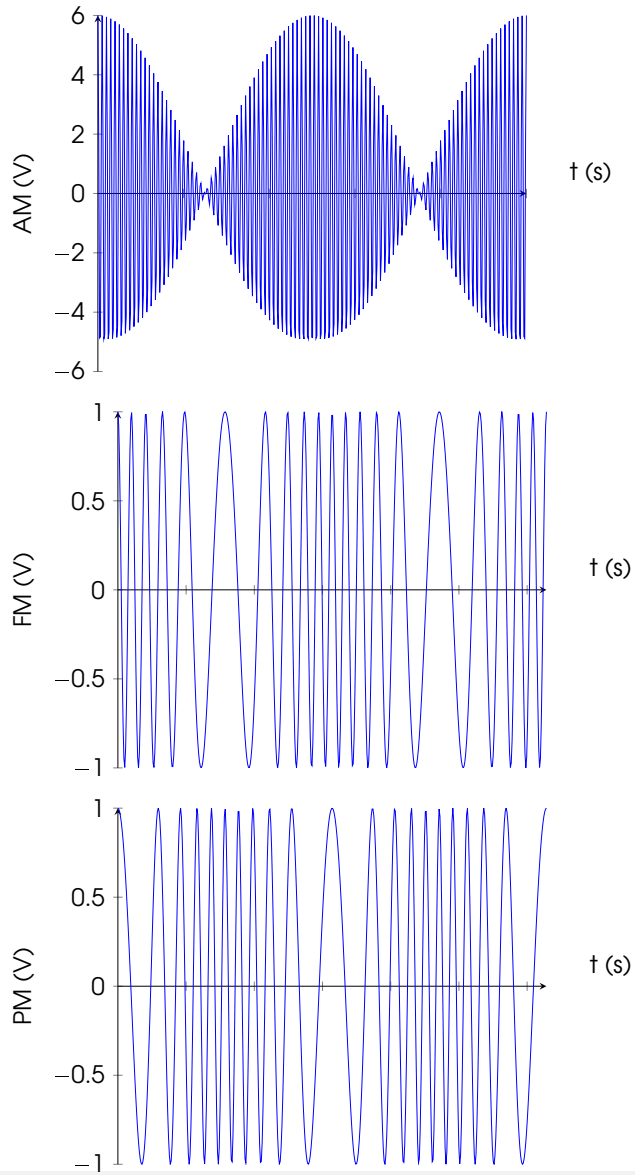
Process Overview

1. Find carrier frequency
2. Determine modulation type
3. Determine encoding scheme
   - Commands
   - Bits/bytes/nibbles
   - Bit rate/baud rate
4. Build encoder
5. Modulate and Transmit



Process Overview

1. Find carrier frequency
2. Determine modulation type
3. Determine encoding scheme
   - Commands
   - Bits/bytes/nibbles
   - Bit rate/baud rate
4. Build encoder
5. Modulate and Transmit

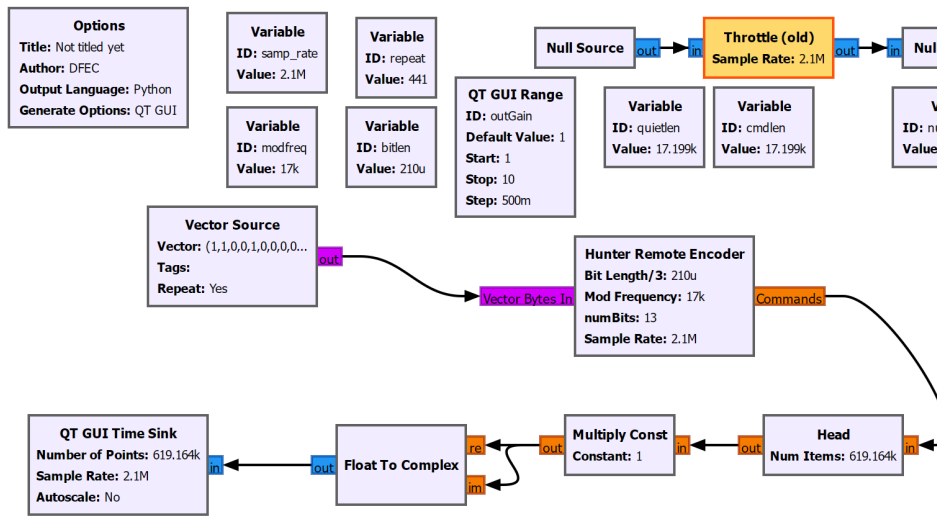


Process Overview

1. Find carrier frequency
2. Determine modulation type
3. Determine encoding scheme
   - Commands
   - Bits/bytes/nibbles
   - Bit rate/baud rate
4. Build encoder
5. Modulate and Transmit

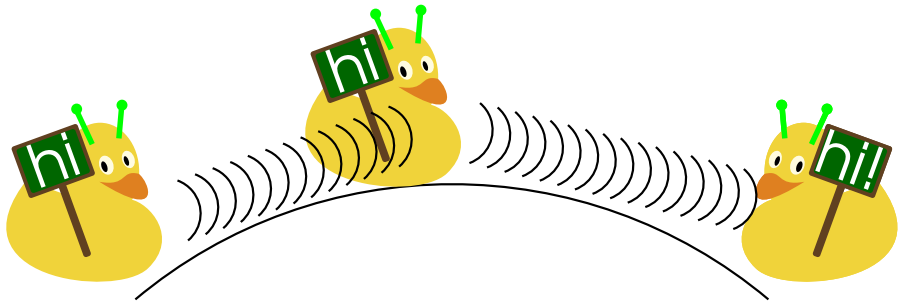
Process Overview

1. Find carrier frequency
2. Determine modulation type
3. Determine encoding scheme
   - Commands
   - Bits/bytes/nibbles
   - Bit rate/baud rate
4. Build encoder
5. Modulate and Transmit



Process Overview

1. Find carrier frequency
2. Determine modulation type
3. Determine encoding scheme
   - Commands
   - Bits/bytes/nibbles
   - Bit rate/baud rate
4. Build encoder
5. Modulate and Transmit



Analysis Tools

- GNURadio
- GQRX
- Universal Radio Hacker (URH)
- SDR Angel

Analysis Tools

- GNURadio
- GQRX
- Universal Radio Hacker (URH)
- SDR Angel

Analysis Tools

- GNURadio
- GQRX
- Universal Radio Hacker (URH)
- SDR Angel

Analysis Tools

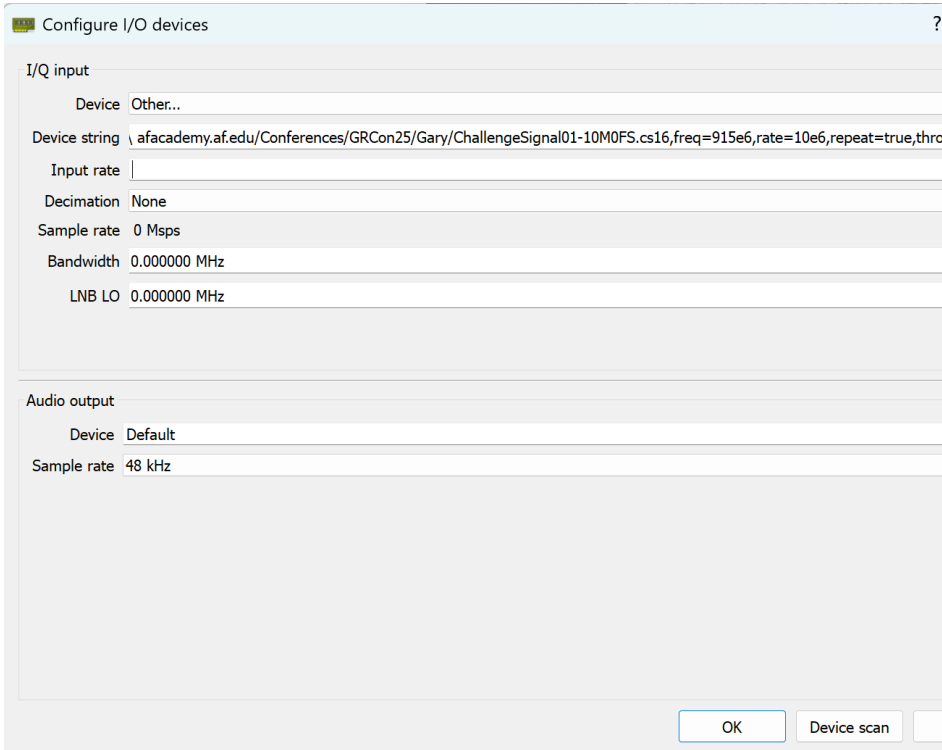
- GNURadio
- GQRX
- Universal Radio Hacker (URH)
- SDR Angel

HARDWARE

- PlutoSDR
- HackRF
- BladeRF
- USRP (B20X, etc.)

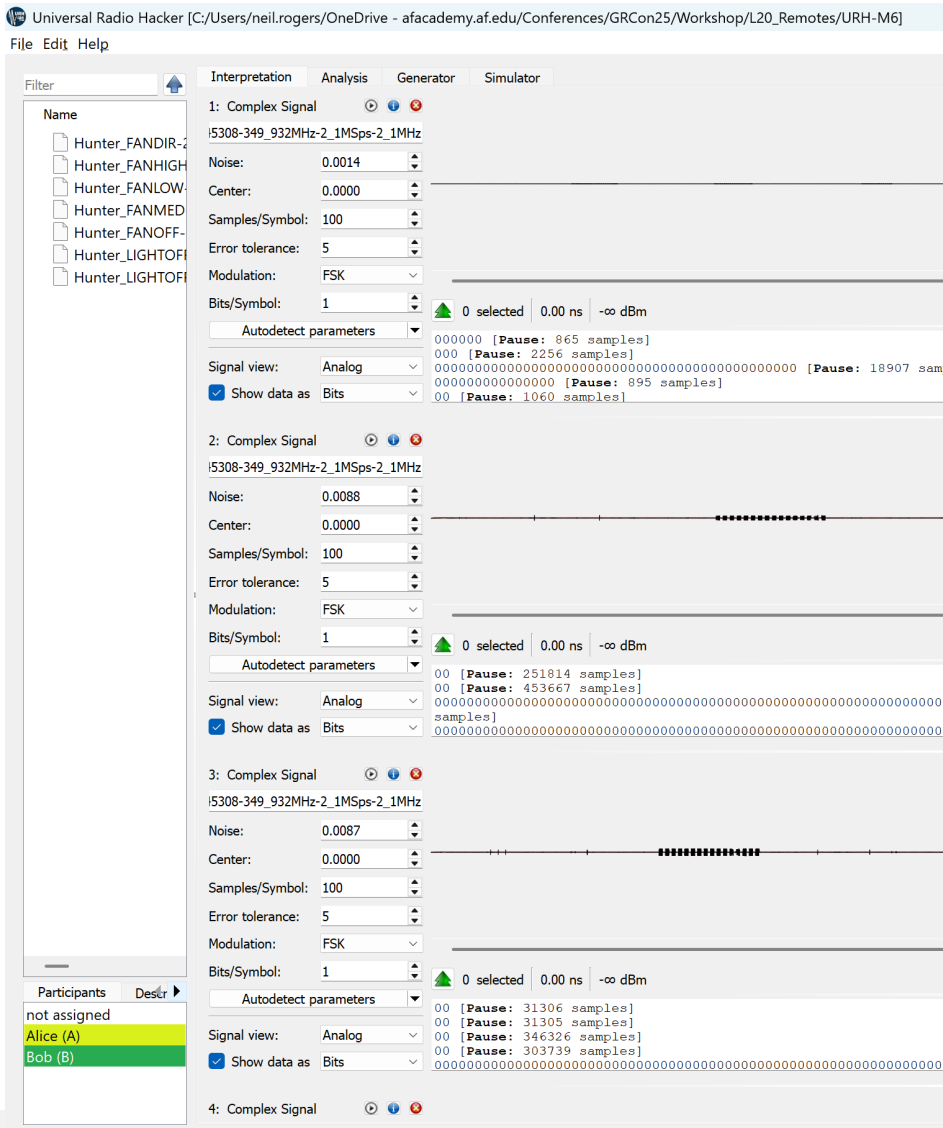
Signal Capture: First Find it

- Let's use GQRX to find it
- Any guesses as to carrier freq?
- Set up GQRX to use your radio (anything but a USRP!)



Signal Capture: URH

- Capture multiple buttons
- Open Project
- Spectrum Analyzer to fine tune carrier acquisition
- Collect data for 3 fan speed buttons

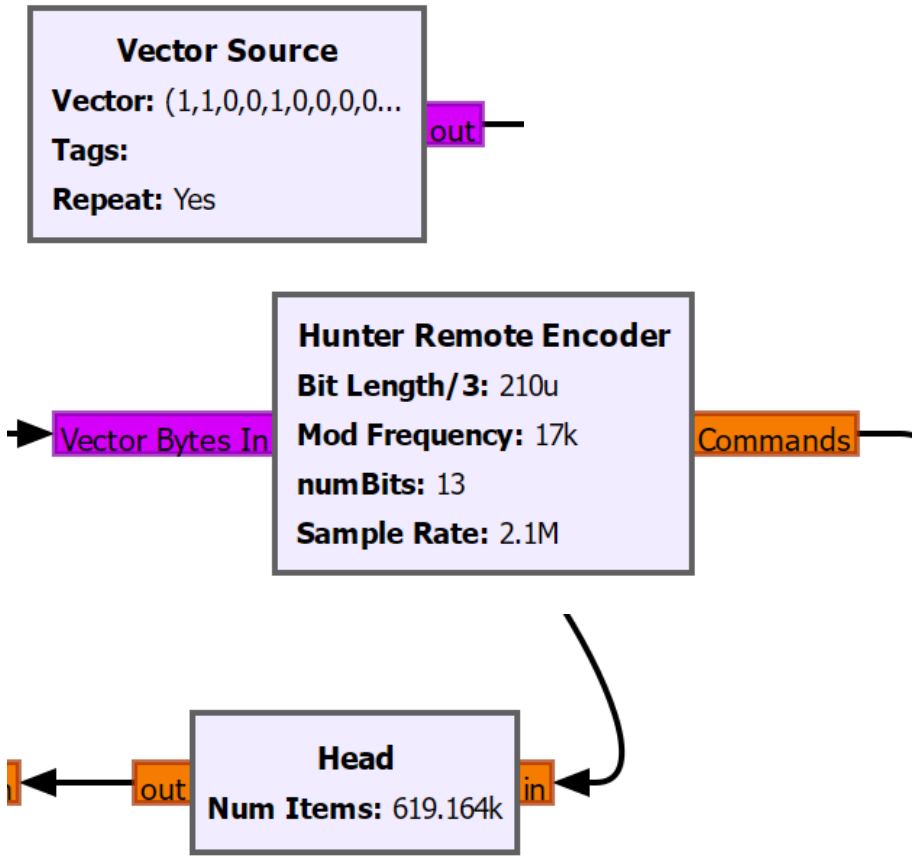


Encoding Analysis

- What fields would we expect to see in a simple message?
- How is information (1's and 0's) encoded?
- Can we identify parts of the message based on collects?
- Is a command sent just once?
- What else do you notice?

Building an Encoder

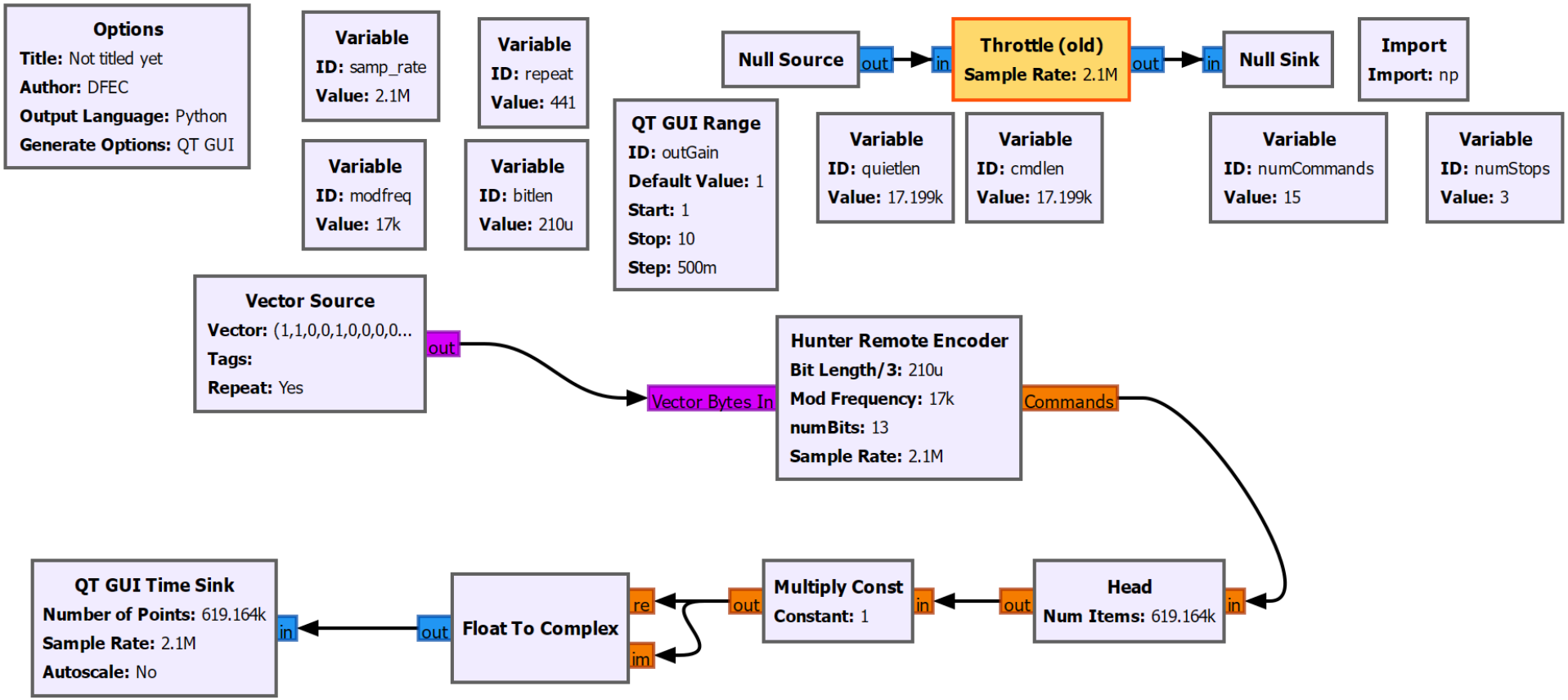
- Source: vector
- Hunter fan encoder block: hier block to generate OOK
  - Mapping
  - Resampling/repeating
  - Adding silence between commands
- Head block to limit transmission
- Amplify output
- Output sink



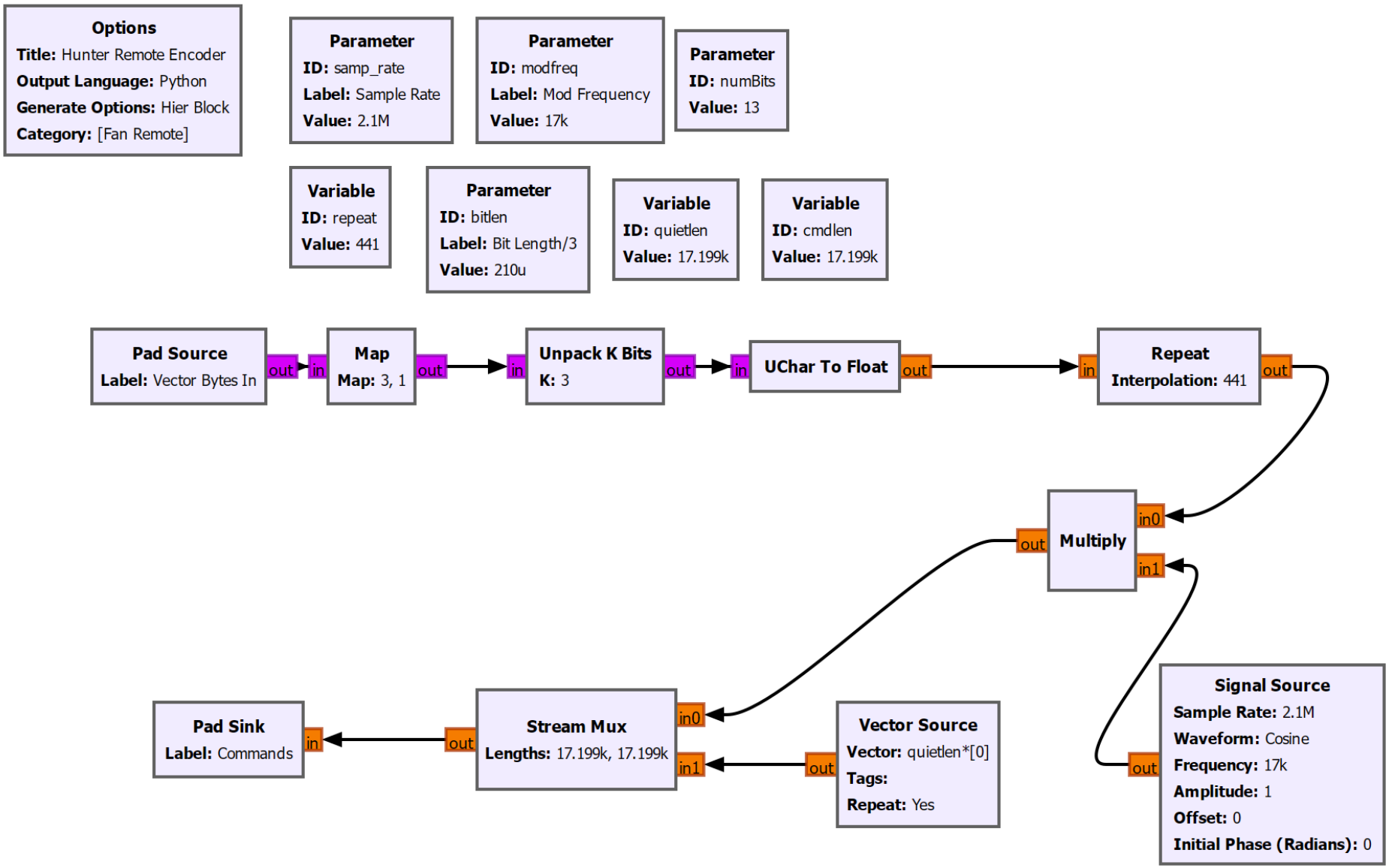
GNURadio-isms and Nuances

- Can we turn the fan on?
- What's the right radio?
- Data format for command input
- Stop command?
- Samples vs Symbols
- Double modulation?
- How many command repeats?
- How to produce on-demand transmit?

Fan Encoder


Options
Title: Not titled yet
Author: DFEC
Output Language: Python
Generate Options: QT GUI
Variable
ID: samp_rate
Value: 2.1M
Variable
ID: repeat
Value: 441
Variable
ID: modfreq
Value: 17k
Variable
ID: bitlen
Value: 210u
QT GUI Range
ID: outGain
Default Value: 1
Start: 1
Stop: 10
Step: 500m
Null Source
out
in
Throttle (old)
Sample Rate: 2.1M
out
in
Null Sink
Import
Import: np
Variable
ID: quietlen
Value: 17.199k
Variable
ID: cmdlen
Value: 17.199k
Variable
ID: numCommands
Value: 15
Variable
ID: numStops
Value: 3
Vector Source
Vector: (1,1,0,0,1,0,0,0,0...
Tags:
Repeat: Yes
out
Vector Bytes In
Hunter Remote Encoder
Bit Length/3: 210u
Mod Frequency: 17k
numBits: 13
Sample Rate: 2.1M
Commands
QT GUI Time Sink
Number of Points: 619.164k
Sample Rate: 2.1M
Autoscale: No
in
out
Float To Complex
re
im
out
Multiply Const
Constant: 1
in
out
Head
Num Items: 619.164k
in

Hier Block


Options
Title: Hunter Remote Encoder
Output Language: Python
Generate Options: Hier Block
Category: [Fan Remote]
Parameter
ID: samp_rate
Label: Sample Rate
Value: 2.1M
Parameter
ID: modfreq
Label: Mod Frequency
Value: 17k
Parameter
ID: numBits
Value: 13
Variable
ID: repeat
Value: 441
Parameter
ID: bitlen
Label: Bit Length/3
Value: 210u
Variable
ID: quietlen
Value: 17.199k
Variable
ID: cmdlen
Value: 17.199k
Pad Source
Label: Vector Bytes In
out
Map
Map: 3, 1
Unpack K Bits
K: 3
UChar To Float
Repeat
Interpolation: 441
Multiply
in0
in1
Signal Source
Sample Rate: 2.1M
Waveform: Cosine
Frequency: 17k
Amplitude: 1
Offset: 0
Initial Phase (Radians): 0
Vector Source
Vector: quietlen*[0]
Tags:
Repeat: Yes
Stream Mux
Lengths: 17.199k, 17.199k
Pad Sink
Label: Commands
in
out