SDR Demonstration

James Flynn
Sharlene Katz

Overview

- Dial Tone Example
- NBFM Receiver Example
- Demonstrations
- Next Meeting
Sampling

Nyquist Rate:

\[ T < \frac{1}{2f_m} \Rightarrow \frac{1}{f_s} < \frac{1}{2f_m} \Rightarrow f_s > 2f_m \]

Example: For an audio signal with \( f_m = 5 \) KHz, we must sample at \( f_s > 10 \) KHz or \( T < 0.1 \) mS.

Dial Tone Example

- This simple example of GNU Radio code adds two sine waves together to create a dial tone
- Block Diagram:
Dial Tone Example

• Generating sine waves

src0 = gr.sig_source_f (sample_rate, gr.GR_SIN_WAVE, 350, ampl)
src1 = gr.sig_source_f (sample_rate, gr.GR_SIN_WAVE, 440, ampl)

Pre-written C++ block is called with number of samples per second, waveform, frequency, and amplitude. Outputs floating point data stream.

Dial Tone Example

• Tell program where to output or “sink” results.

dst = audio.sink (sample_rate, options.audio_output)

This is the interface to the outside world and user. Converts floating point data stream into analog signal in sound card.
Dial Tone Example

• Connect the sources and sinks together.

self.connect (src0, (dst, 0))
self.connect (src1, (dst, 1))

This is the real power of python programming.

---

Dial Tone Example

• Set up a way to pass values to program from command line

```python
parser = OptionParser(option_class=eng_option)
parser.add_option("-O", "--audio-output", type="string", default="",
    help="pcm output device name. E.g., hw:0,0 or /dev/dsp")
parser.add_option("-r", "--sample-rate", type="eng_float", default=48000,
    help="set sample rate to RATE (48000)")
parser.add_option("-a", "--ampl-entry", type="eng_float", default=0.1,
    help="set amplitude to (0.1) Must be less than 1.0")
(options, args) = parser.parse_args()
```

• If command line is empty, print the help lines to assist user

```python
if len(args) != 0:
    parser.print_help();
    raise SystemExit, 1
```

• Example:

```bash
./dial_tone -r 44100 -a .95 #sets sample rate to 44100 and amplitude to .95
```
Dial Tone Example

```python
sample_rate = int(options.sample_rate)
ampl = options.ampl_entry
```

- Define variables in the program with the values passed in command line.

Dial Tone Example

- Setting things up:

```bash
#!/usr/bin/env python
```

- First line tells shell to use python interpreter to compile file.

```python
from gnuradio import gr
from gnuradio import audio
from gnuradio.eng_option import eng_option
from optparse import OptionParser
```

- These “import” lines tell compiler what other modules to include with the program
Dial Tone Example

class my_top_block(gr.top_block):
    This line sets up the flow graph, i.e. the framework of the program using the pre-existing code gr.top_block

    def __init__(self):
        gr.top_block.__init__(self)

These two lines define the function __init__ for this class and call the parent constructor to initialize the program.

Dial Tone Example

if __name__ == '__main__':
    try:
        my_top_block().run()
    except KeyboardInterrupt:
        pass

    • These lines run the program until Control-C is pressed.
Dial Tone Example

• Python script is run once at beginning to compile code that actually performs signal processing.
• Actually a series of instructions to tell computer how to write software.
• Python script may refer to other blocks of code which use still other blocks of code which use even other blocks… etc.

Design of a Receiver

• USRP: Set frequency of local oscillator (receive frequency), gain of amplifier, decimation factor
• GNU Radio application: use Python to specify and connect blocks that perform demodulation and decoding
Example: 400 - 500 MHz NBFM Receiver

- Problem: Receive an audio signal (up to 4 KHz) transmitted at 446 MHz using narrowband FM (NBFM) with a 16 KHz transmission bandwidth

Design Procedure

1. Plan the block diagram of system components
2. Determine block parameters
3. Determine decimation rates
4. Write Python script to specify the blocks and connect them together
Determining the Decimation Factors

Total Decimation factor = 8000 = D_1 D_2 D_3

64Msamp/sec 8Ksamp/sec
FPGA Decimation Factor, $D_1$

- Total Decimation factor = $8000 = D_1D_2D_3$
- Maximize the decimation in FPGA
- Maximum decimation factor in FPGA = 256
- Select $D_1 = 250$ (factor of 8000)
- Output sample rate = $64\text{Ms/s} / 250 = 256\text{Ks/s}$

Channel Filter Specification

- Maximum frequency = 16 KHz $\rightarrow$ Reduce sample rate to 32 Ks/s
- $256\text{Ks/s} / 32\text{Ks/s} \rightarrow D_2 = 8$
FM Demodulator

- Maximum frequency = 4 KHz → Reduce sample rate to 8 Ks/s
- 32Ks/s / 8Ks/s → $D_3 = 4$
- FM Demodulator block “extracts” audio signal from FM waveform by operating on I and Q

Complete Application Design

- Total decimation ratio = 250*8*4 = 8000
- Problem: The audio card requires an input sample rate ≥ 44.1 Ks/s
- Solution: Use a Resampler to increase the output sample rate
Final Application Design

- Audio Card requires a sample rate ≥ 44.1 Ks/sec. Use 48 Ks/sec.
- Modify FM Demodulator to have a decimation factor of 1 (no change)
- Increase the sample rate to 48 Ks/sec with Resampler (x 3/2)

Implementing the Design

- Create a Python script to specify and connect the various GNU radio blocks
- Blocks are already written in C++
- USRP parameters are set within Python script
- # indicates that the line is a comment
- Refer to nbfm.py script
Setting the USRP Parameters

• The following code sets the USRP Parameters:

```python
# Create USRP data source
u = usrp.source_c(decim_rate=250)

# Tell USRP what daughter board to use and displays name
# Find the board on side A
rx_subdev_spec=(0,0)
rx_subdev_spec=usrp.pick_rx_subdevice(u)
u.set_mux(usrp.determine_rx_mux_value(u,rx_subdev_spec))
subdev = usrp.selected_subdev(u,rx_subdev_spec)

print "Using RX d’board %s" % (subdev.side_and_name(),)

# Tune it to supplied frequency
u.tune(0,subdev,frequency)
```

Channel Filter Design

• The following code specifies the channel filter and computes the coefficients:

```python
# Create channel filter coefficients
chan_taps = optfir.low_pass(1.0, 256e3, 8000, 9000, 0.1, 60)
```

![Channel Filter Design Diagram]
Channel Filter Creation

- The following code creates the channel filter using the coefficients computed:

```python
# creates the channel filter with the coef found above
chan = gr.freq_xlating_fir_filter_ccf(8, # Decimation rate
    chan_taps, # coefficients
    0.0, # Offset frequency - could be used to shift
    256e3) # incoming sample rate
```

FM Demodulator

- The following code creates the FM demodulator.
- The demodulator block also includes a low pass filter.

```python
demod = fm_demod_cf(32e3, # Sample Rate at input
    1, # Decimation
    5000, # Deviation
    3000, # edge of audio passband
    4000) # edge of audio stopband
```
Resampler

- The following code creates the resampler.
- The resampler decimates and/or interpolates the data to adjust the sample rate.

```
#insert resampler to increase sample rate of 32K to 48K
# mult by 3 and divide by 2
rsamp = blks2.rational_resampler_fff(3,2)
```

Connecting the Blocks

- The following code connects the blocks:

```
self.connect(u,chan)
sel.connect(chan,demod)
sel.connect(demod,rsamp)
spk = audio.sink(48000)
sel.connect(rsamp,spkr)
```

Or, a single connect statement:

```
self.connect(u,chan,demod,rsamp,spkr)
```
Final Thoughts

• Demonstrations
  – NBFM Receiver
  – Spectrum Analyzer
  – Oscilloscope

• Questions

• Next Meeting