OOK Modulation with GNURadio

https://github.com/afonsonf/ook-modulation
Afonso das Neves Fernandes
201606852

***Abstract*—Signal modulation is a important topic in communications because it allows to encode information in a signal and transmit the signal wireless. In this work we will study how to use GNURadio to create a software defined radio capable of modulate and demodulate a signal using On-Off Keying modulation. First we will present the modulation and encoding used. Then we will present the flow graphs of GNURadio to modulate and demodulate a signal and finally go through a example of modulating the string ”hello world”.**

***Index Terms*—Software Defined Radio, GNURadio, OOK**

I. Introduction

In this chapter we will talk about what is signal modulation and the main ways to encode information in a signal. We will also present the modulation and the software used in this work.

A. Signal Modulation

Signal modulation is the process of encoding information in a signal, normally a sinusoidal wave. This wave is then transmitted through space as an electromagnetic wave.

The main ways to encode information in a signal are [1]:

- Frequency Modulation (FM), where the information is encoded by varying the frequency of the wave.
- Phase Modulation (PM), where symbols used for encoding correspond to different initial phases of the wave.
- Amplitude Modulation (AM), where the bits of the information are encoded in levels of amplitude.

B. On-Off Keying Modulation

In this work we will focus on a form of Amplitude Modulation, On-Off Keying (OOK).

In OOK there are two symbols to encode information, the presence or absence of a carrier signal. The encoding that was used is the following:

- The bit 1 is encoded with $3/4$ of the period high and $1/4$ low.
- The bit 0 is encoded with $1/4$ of the period high and $3/4$ low.

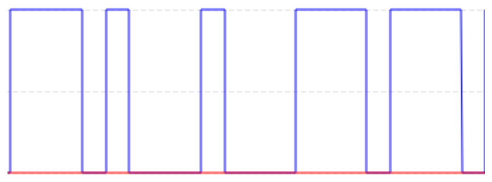


Fig. 1: Modulation of 10011 using OOK.

This is the same as encoding a 1 with 1110 and a 0 with 1000 and is used in a variety of remotes (Figure 1).

This encoding is used because is more resistant to noise interference.

C. GNURadio

GNURadio is a free and open source software that can be use to create software defined radios and analyze signals. GNURadio can be used with external RF hardware to receive and transmit signals [2].

The main way you can use the software is by creating flow graphs using blocks for both signal source and sink as well as manipulating the signal between source and sink. You can also create custom blocks in python.

In this work we created two flow graphs (Figure 2 and 3), one for modulation and other for demodulation, both using OOK modulation, that will be explained in the next chapters.

II. Signal Modulation

In this chapter we will explain the flow graph in Figure 2 used for modulation of a signal, given a list of bits. The flow graph is based on the work of ”Impersonating a remote using SDR and GNURadio” by Foo-Manroot [3].

The flow graph accepts a file with a binary string but with a simple python script (Listing 1) is possible to modulate from a plain text file.

Listing 1: Python script to encode plain text to binary.

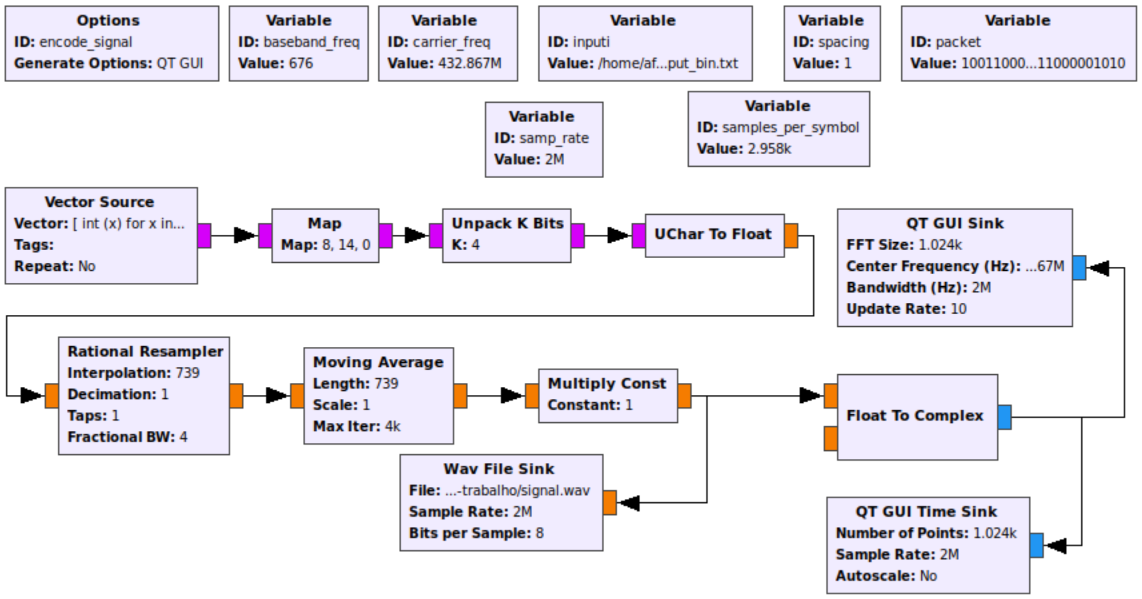
```
from binascii import hexlify

file = open('input_text.txt')
lines = [l for l in file]
s = ''.join(lines)
print 'encoding: \n', s, '\n'

sb = bin(int(hexlify(s), 16))[2:]
file = open('input_bin.txt','w')
file.write(sb)
file.close()
print 'output: \n', sb, '\n'
```

The modulation proceeds as follows:

- First we encode the text in ”input_text.txt” to binary using the python script in Listing 1.
- In the flow graph the binary file is loaded to variable $packet$ and then used as a Vector Source with a 2 in the end that will represent the end of the message.
- The 0 is then mapped to 8 (1000 in decimal), the 1 to 14 (1110 in decimal) and the 2 to 0.



Options
ID: encode_signal
Generate Options: QT GUI

Variable
ID: baseband_freq
Value: 676

Variable
ID: carrier_freq
Value: 432.867M

Variable
ID: inputi
Value: /home/af...put_bin.txt

Variable
ID: spacing
Value: 1

Variable
ID: packet
Value: 10011000...11000001010

Variable
ID: samp_rate
Value: 2M

Variable
ID: samples_per_symbol
Value: 2.958k

Vector Source
Vector: [int (x) for x in...
Tags:
Repeat: No

Map
Map: 8, 14, 0

Unpack K Bits
K: 4

UChar To Float

QT GUI Sink
FFT Size: 1.024k
Center Frequency (Hz): ...67M
Bandwidth (Hz): 2M
Update Rate: 10

Rational Resampler
Interpolation: 739
Decimation: 1
Taps: 1
Fractional BW: 4

Moving Average
Length: 739
Scale: 1
Max Iter: 4k

Multiply Const
Constant: 1

Float To Complex

Wav File Sink
File: ...-trabalho/signal.wav
Sample Rate: 2M
Bits per Sample: 8

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 2M
Autoscale: No

Fig. 2: GNURadio flow graph for modulation of a signal.

- The numbers are separated into bits (e.g. $[8]$ into $[1, 0, 0, 0]$)
- Then, each bit is re-sampled from 1 to 739 samples (sample rate divided by baseband freq and by 4) and the moving average is computed on those samples
- Finally the result is written to a wav file.

III. Signal Demodulation

In this chapter we will explain the flow graph in Figure 3 used for demodulation of a signal, given a wav file. The flow graph was originally made, not based in any one.

The flow graph outputs a binary file that can be decoded to plain text using a simple python script like the one presented in Listing 2.

Listing 2: Python script to decode binary to plain text.

```
from binascii import unhexlify

file = open('out_bin.txt')
lines = [l.strip() for l in file]
print 'decoding: \n', lines[0], '\n'

n = int(lines[0].encode('ascii'),2)
print 'output: \n', unhexlify( '%x' % n ), '\n'
```

The demodulation proceeds as follows:

- First the wav file is loaded by a wav file source block.
- The signal goes through a throttle block in order to have the correct sample rate.
- Then the signal goes through a Symbol Sync Block. This block will identify each bit in the signal by measuring the average value in each 739 samples (samples per bit).
- Each four bits are packed into a number (e.g. 1000 into a 8, 1110 into a 14 and 0000 into 0)
- Each number is mapped into the original value, 8 is mapped to 48 (order of 0 in ascii table), 14 into 49 and 0 into a 10 (newline) to separate repetitions of the original text.
- Finally we can run the script in Listing 2 to decode the binary output of the flow graph to plain text.

IV. Example

In this chapter we will show how the modulation of the string ”hello world” proceeds.

First the string is encoded to binary using the script in Listing 1. The output binary is:

1101000011001010110110001101100011011110010000000
1110111011011110111001001101100011001000000001010

Then each bit is encoded into 1110 and 0001 conformed what was explained in previous chapters. The result of the encoding with the separator symbol at the end is the following:

1110111010001110100010001000100011101110100010000
11101000111010001110111010001110111010001000010001
11011101000111011101000100010001110111010001110111
10111011101000100011101000100010001000100010001111
01110111010001110111011101000111011101000111011100
11101110100011101110111010001000111010001000111011
11010001110111010001000100011101110100010001110101
00010001000100010001000111010001110100000000

The resulting signal in the wav file is showed in Figure 4. The image is a screenshot of the wav file opened with the software Sonic Visualizer [4].

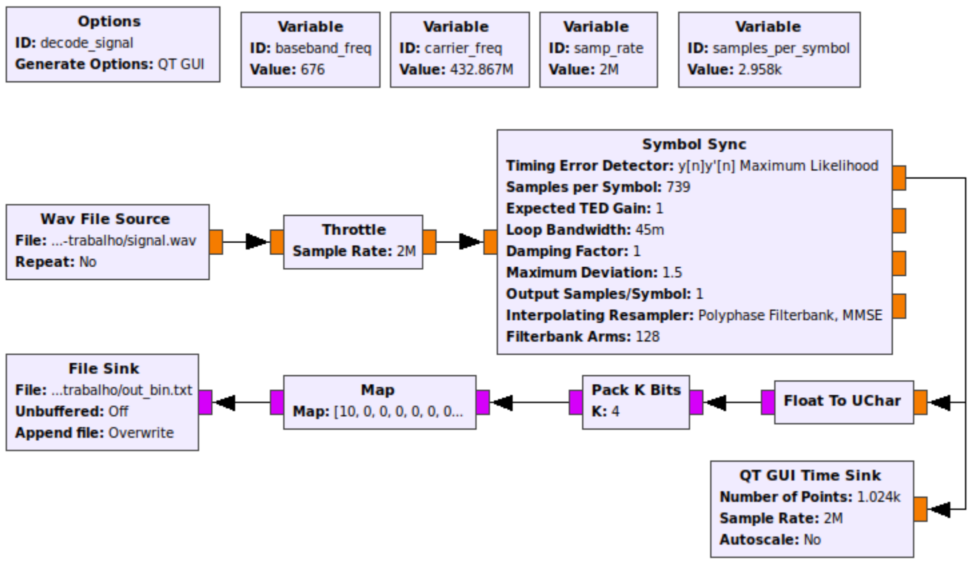


Fig. 3: GNURadio flow graph for demodulation of a signal.

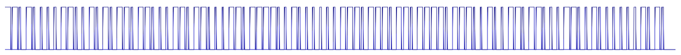


Fig. 4: Modulated signal of the string ”hello world” using OOK.

To recover the message we can use the demodulation flow graph with the wav file that will output the following binary:

```
11010000110010101101100011011000110111100100000
01110111011011110111001001101100011001000000001010
```

That can be finally decoded to the original string ”hello world” with the script in Listing 2.

V. Conclusion and Future Work

Signal Modeling is one of main aspects in communication and a important field of research. In this work we showed how to use GNURadio to test a simple modeling scheme. This work can be iterated by creating and using more efficient and simple blocks. Finally this works shows how one can use GNURadio to test and create new modeling schemes.

References

[1] H. Roder, “Amplitude, phase, and frequency modulation,” *Proceedings of the Institute of Radio Engineers*, vol. 19, pp. 2145–2176, Dec 1931.
[2] GNURadio. https://github.com/gnuradio/gnuradio.
[3] Foo-Manroot, “Impersonating a remote using sdr and gnuradio.” https://foo-manroot.github.io/post/gnuradio/sdr/2018/01/15/gnuradio-ook-transmit.html.
[4] Sonic Visualiser. https://www.sonicvisualiser.org.