GNU Radio: Signal Intelligence Toolbox

GSoC 2016 proposal

Sebastian Müller
Karlsruhe Institute of Technology
`gsenpo@gmail.com`

March 23, 2016

1 Introduction

Signal intelligence describes the gathering of information out of intercepted radio signals with unknown origin and unknown parameters.

It is typically used by intelligence agencies to get information about various radio sources where normally a receiver has knowledge about the sender's signal parameters like symbol rate, modulation and frequency band. The purpose of signal intelligence is extracting the transmitted information and to enable receiving of the source signal without any of this knowledge.

GNU Radio has already the ability to do signal intelligence, but it is only possible through some detours. A typical signal intelligence workflow could at this time be recording a broadband spectrum and afterwards searching for signals in the spectrum plot. Then, one signal can be mixed to DC and low-pass filtered. After that, one is able to begin analyzing the time/spectrum/waterfall/constellation plot to make a guess for the modulation. As the signal is recorded, it can be manually synchronized by estimating the symbol rate while examining the time signal. Samples can be moved or thrown away to realize manual synchronization or a synchronization block from GNU Radio can be used. Finally, one can choose the correct demodulation block for the guessed modulation scheme to get the transmitted bits. Real-time signal intelligence is hardly possible now. The tool `gqrx` has some of this ability (real-time AM/FM demodulation), but there is no final solution for this problem at this time. The target of this project is to develop an easily accessible and extendible solution for this workflow.

2 About me

As a graduate student of electrical engineering with focus on communication engineering in my 10th semester and a B.Sc. degree, I have advanced knowledge of signal processing and radio technology. I have chosen so far various lectures that will become useful during this project, like *Communication Engineering*, *Signal Processing in Communication Systems*, *Methods of Signal Processing* and *Modern Radio Systems Engineering*. I have also done multiple projects involving Matlab, C++ and Python during my studies and internships.

During my six months bachelor's thesis 2014 at the Communications Engineering Lab (CEL) of the Karlsruhe Institute of Technology I worked together with Stefan Wunsch at the `gr-radar` OOT module and implemented a radar cross section estimation block [1].

Additionally, I won the 1st price of the IEEE Signal Intelligence Challenge, hosted at the CEL, in 2014 and 2015. During these competitions, I gathered much practial and theoretical knowledge about signal intelligence and signal processing. I used GNU Radio to solve several challenges during these competitions. As a student of the CEL I enjoy access to a big hardware pool in addition to my own Ettus USRP B210.

I have been watching the GNU Radio community for some time now and want to take GSoC as the opportunity to begin with my participation in the project. GNU Radio fits perfectly to my study focus and I am confident that I can contribute valuably to that project.

I have acknowledged and understood the rules of the GNU Radio community.

3 Benefits

The GNU Radio project can benefit from my work in the following ways.

The idea behind software defined radio is a multi purpose solution which can be easily adapted to different radio standards by changing the software. This process can be made significantly easier by automated signal intelligence algorithms. In the ideal case, the user does not have to know any properties of a signal to receive it and extract information from it. The signal intelligence toolkit would make a huge step in that direction.

GNU Radio is predestined to be used by practical beginners in the field of radio engineering. Everyone has access due to the free software status of the project and hardware can easily be bought (e.g. DVB-T sticks). The understanding of radio technology can be improved significantly by examining existing radio signals and finding out **why** and **how** they work. With `gr-sigint`, any beginner can examine radio signals without needing expert knowledge in this field.

4 Deliverables

The explicit components that I plan to develop during my GSoC program are listed below. My focus will lay on developing a practical solution that will work as out-of-the-box as possible. Methods will be prototyped in Python and finally implemented in C++ afterwards. Since signal intelligence is a vast topic, the deliverables are separated in core deliverables, which I will definitely implement during GSoC, and additional deliverables, which I will try to implement if there is enough time. Also, they are a good starting point for the post-GSoC period.

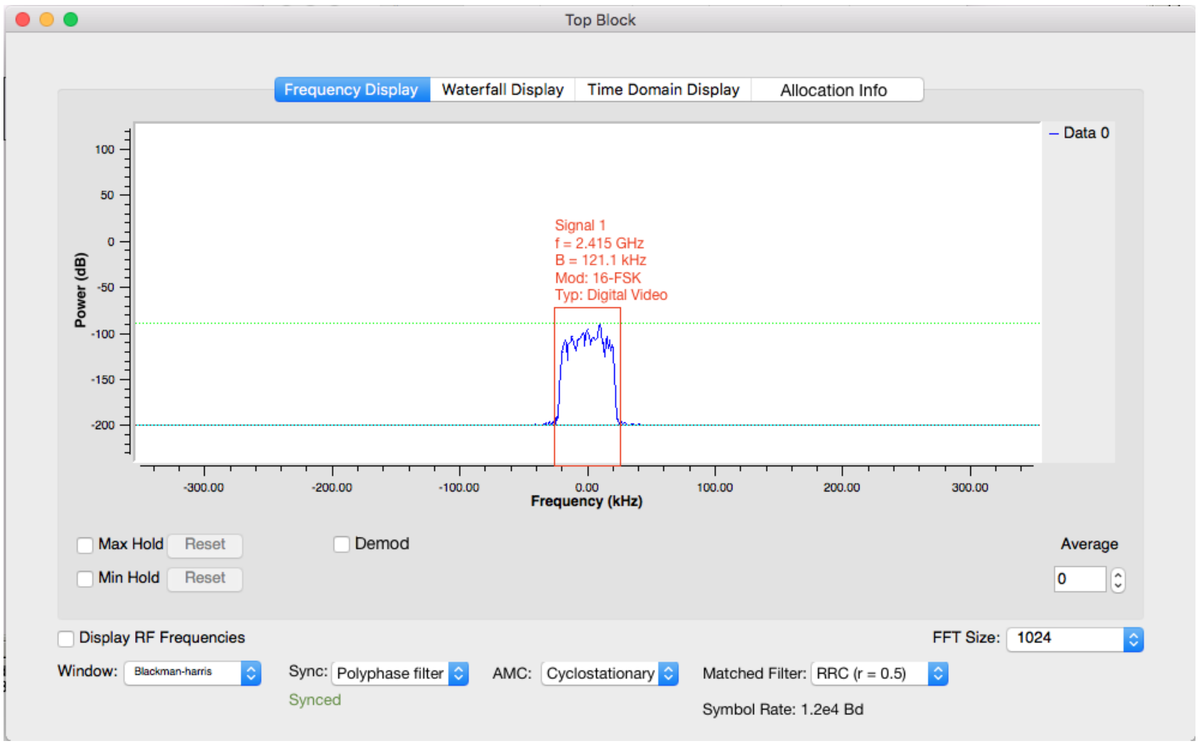
4.1 Core deliverables

Out-of-tree module `gr-sigint` Create new OOT module `gr-sigint` with `gr_modtool`. All of my work shall be part of this module. This module will be in the center of my participation in GNU Radio after the GSoC program. It shall be easily extendible and comprehensible.

Signal detection block When receiving a wideband signal, it is useful to automatically extract possible signals from the broadband spectrum. A block capable of detecting signals in the spectrum shall be developed. Also, the message API could be used to pass on information about the frequency band to look up possible radio services in following blocks. The implementation can be done by a simple energy detection followed by a RF map of the spectrum as described in [2]. Another approach is using a filter bank to split the input spectrum in sub channels. The sub channels can be examined with an energy detection and afterwards, the sub channels which contain a signal can be reconstructed with the work of [3].

Modulation classification block A seperate block shall be developed, capable of automatic modulation classification (AMC). Image comparison algorithms [6] can be applied to the spectrum, time signal, waterfall and constellation plots or analysis of cyclostationary features can be used for modulation detection. The results of [4] and [5] can be used and adapted here. As an additional target, machine or deep learning frameworks like *scikit-learn* or *Tensorflow* can be implemented [7]. The problem here for the typical user would be to generate enough training data for deep learning. Deploying a pre-trained version can be a solution. Approaches on this where made in [2]. The block shall be able to recognize FM, AM analog modulations as well as lower PSK, FSK and QAM. An easy extension of the modulation schemes will be possible.

Visualization Implementing a GUI will make the module very user friendly and easy accessible. The received spectrum and the detected signal shall be displayed. Also, information about the detected signals with ability to demodulate in real-time are possible in the GUI. The tool gqrx [8] can serve as an example for this purpose. The GUI will be implemented in QT4/QT5 with pyqtgraph. A simple concept can be seen in Figure 1.



Signal 1
f = 2.415 GHz
B = 121.1 kHz
Mod: 16-FSK
Typ: Digital Video

Average: 0

FFT Size: 1024

Window: Blackman-harris

Sync: Polyphase filter

Synced

AMC: Cyclostationary

Matched Filter: RRC (r = 0.5)

Symbol Rate: 1.2e4 Bd

Figure 1: Concept of GUI information (derivated from QT GUI Sink)

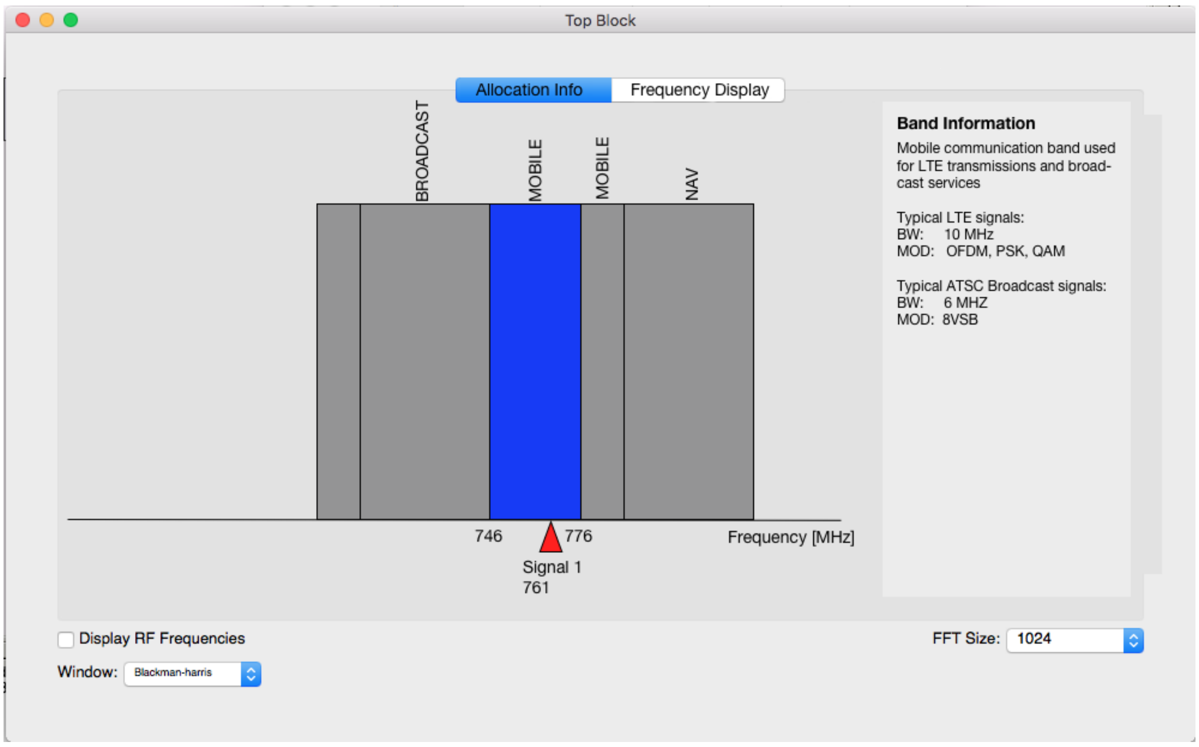
Documentation For an optimal collaboration, a good documentation of my work will be necessary. This will be realized by various source code comments and a distinct documentation in doxygen. Also, my work progress will be published on a blog with weekly updates. When necessary, video examples will be provided there. In addition, example flowgraphs with typical application cases will be integrated in the project.

4.2 Additional deliverables

Blind synchronization In order to examine the constellation diagram and to demodulate the signal, a (blind) synchronization has to be performed beforehand. This can be done using frequency synchronization techniques with PLL or Costas loops. Timing recovery can be achieved with derivative matched filters or polyphase filter banks as described in [9]. As most modern radio services use preambles for receiver synchronization, common preambles can be searched in the signal (with aid of allocation information). In this case, the Correlate and Sync block is suited for synchronization.

Smart demodulation When a modulation scheme is estimated, there needs to be a demodulation which regards that result trying to demodulate the received signal. The message API can also be used here. It will be tested if existing demodulation blocks can be used for this purpose or if a sepearate block is needed. The samples per symbol ratio can be estimated by estimating the symbol rate under knowledge of the sample rate, for instance with the results of [10]. Other parameters like shaping filters can be made selectable by the user. Research on this topic was made in [11].

Radio service database When trying to demodulate signals, it will be helpful to have a database with known radio signal services and their properties. I will do research on this topic and try to find a compatible solution with GNU Radio. When there is no database to find, I will try to make a basic, extendable example. An example of the allocation information integration in the GUI can be seen in Figure 2.



Band Information
Mobile communication band used for LTE transmissions and broad-cast services

Typical LTE signals:
BW: 10 MHz
MOD: OFDM, PSK, QAM

Typical ATSC Broadcast signals:
BW: 6 MHZ
MOD: 8VSB

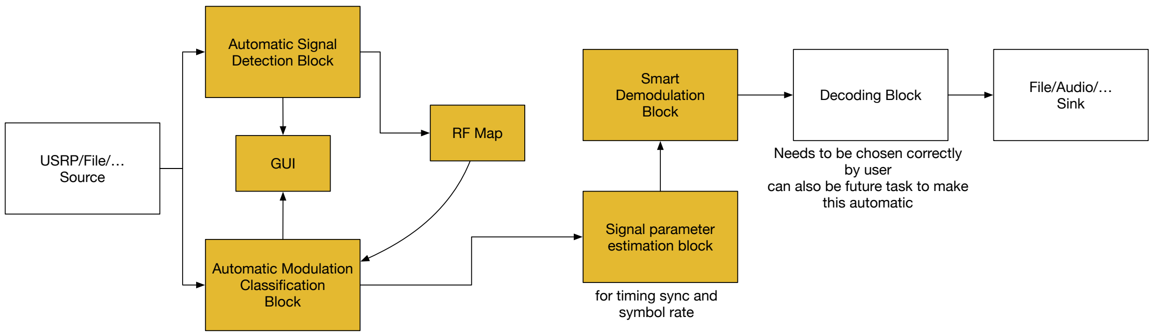
☐ Display RF Frequencies

Window: Blackman-harris

FFT Size: 1024

Figure 2: Concept of allocation GUI (derivated from QT GUI Sink)

A basic block diagram of a signal intelligence workflow derivated from [2] can be seen in figure 3.



Automatic Signal Detection Block
USRP/File/... Source
GUI
RF Map
Automatic Modulation Classification Block
Smart Demodulation Block
Signal parameter estimation block
for timing sync and symbol rate
Decoding Block
Needs to be chosen correctly by user can also be future task to make this automatic
File/Audio/... Sink

Figure 3: Block diagram of signal intelligence workflow. The yellow blocks are planned to be examined/developed during the GSoC program.

5 Timeline

The timeline is orientated at the official GSoC timeline. As I have completed most of my studies with only one exam left, I can focus perfectly on my GSoC task. I also have a side job, but with variable workload (I can decline tasks as I wish). Be assured that the GSoC project will be my top priority during this summer.

Apr 22 - May 22 Get in touch with the GNU Radio community. Read the docs, install the software and play around to get used to the current version. Set up working environment and set up blog homepage. Do some literature research on my topic and begin to write first test code to evaluate possible approaches to the problem.

May 23 - Jun 02 Begin coding by creating new OOT module. Begin prototyping of signal detection block in Python. Try to detect signals in self-generated over-the-air samples and file sources. Access to hardware is provided, so over-the-air testing can be done.

Jun 03 - Jun 06 Implementation of previously evaluated signal detection algorithm in C++. Wrap algorithm in new GNU Radio block and write corresponding documentation.

Jun 07 - Jun 16 Start implementing GUI. Use existing QT sink or write own block to display results of signal detection block. Graphical feedback of signal detection will be implemented. Working GUI shall be availiable until midterms.

Jun 17 - Jul 07 Prototyping of modulation detection algorithms. Use different approaches: cyclostationary features, image comparison and deep learning algorithms. Test algorithms with over-the air signals.

Jul 08 - Jul 12 Implement results of modulation detection research in C++. Create a new GNU Radio block for modulation detection and write documentation.

Jul 13 - Jul 18 Expand GUI to display results of signal modulation detection block.

Jul 19 - Jul 27 Examine existing solutions for blind synchronization (PLL, derivative matched filters, etc.). Use existing blocks or implement new techniques with best performance during examination. Also, implement a simple preamble detection which will be selectable by the user for one signal type.

Jul 28 - Aug 03 Evaluate solution for smart demodulation. Try to use existing demodulation blocks and select them on demand. Otherwise create new smart demodulation block with existing demodulation algorithms and expand documentation with evaluated results.

Jul 04 - Aug 09 Do research on radio service database. Find solution how to connect with GNU Radio. If nothing is available, create own extendable example.

Aug 10 - Aug 13 Finish code documentation of created blocks and polish corresponding doxygen pages. Create example flowgraphs of typical applications with this module and create PyBOMBS recipe. Also use this time as buffer for remaining problems before pencils down deadline.

Aug 15 - Aug 23 Pencils down, provide version 1.0 of package via CGRAN. Submit the code samples to Google, revise documentation and cleanup code.

6 Conclusion

I think GNU Radio is the perfect project for communication engineers and people who want to get involved in radio technology to play around and build great applications. Signal intelligence is a very promising topic and there are endless possibilities that come to my mind for the post-GSoC future.

Participating in the GNU Radio project with the `gr-sigint` module would be a thrilling experience for me. I hope I successfully provided interesting ideas for the community and show why I think I am predestined to realize them. If you have any questions, please do not hesitate to contact me. And of course, Polar Codes are the coolest codes.

References

[1] `https://github.com/kit-cel/gr-radar` Visited March 5, 2016.

[2] O'shea, Timothy J., T. Charles Clancy, and Hani J. Ebeid."Practical signal detection and classification in gnu radio" SDR Forum Technical Conference (SDR). 2007.

[3] F. Harris, E. Venosa, Xiofei Chen and C. Dick,"Interleaving different bandwidth narrowband channels in perfect reconstruction cascade polyphase filter banks for efficient flexible variable bandwidth filters in wideband digital transceivers", Digital Signal Processing (DSP), 2015 IEEE International Conference on, Singapore, 2015, pp. 1111-1116.

[4] Mühlhaus, Michael Sebastian, "Automatische Modulationsartenerkennung in MIMO-Systemen", Dissetation, Karlsruhe, Karlsruher Institute of Technologie (KIT), 2014.

[5] K. Kim, I. A. Akbar, K. K. Bae, J. S. Um, C. M. Spooner and J. H. Reed, "Cyclostationary Approaches to Signal Detection and Classification in Cognitive Radio" New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on, Dublin, 2007, pp. 212-215.

[6] `http://www.sigidwiki.com/wiki/Database#List_of_Known_Signals_in_Database` Visited March 9, 2016.

[7] O'Shea, Timothy J., and Johnathan Corgan. "Convolutional Radio Modulation Recognition Networks" arXiv preprint arXiv:1602.04105 (2016).

[8] `https://github.com/csete/gqrx` Visited March 7, 2016.

[9] V. Nee, "Globalization of mobile and wireless communications: Today and in 2020", R. Prasad, S. Dixit, and R. Van Nee, Eds. Netherlands: Springer-Verlag, 2010.

[10] Y. Gao, M. Li, Z. Huang, and J. Lu,"A symbol rate estimation algorithm based on Morlet wavelet transform and autocorrelation" IEEE, pp. 239–242.

[11] F.F. Liedtke, "Computer simulation of an automatic classification procedure for digitally modulated communication signals with unknown parameters", Signal Processing, Volume 6, Issue 4, 1984, Pages 311-323