GSOC'12: Extend and improve filter design components

Sreeraj Rajendran

April 6, 2012

1 Introduction

Digital filter is one of the inevitable signal processing blocks in any DSP system. GNU Radio provides a true free and open platform for designing practical filters which can be used in real time. The proposal addresses ideas for improving filter design components as suggested by Martin Braun (CEL, KIT). Section 2 covers basic details about digital filter designs and the filters currently available in GNU Radio. A small review of the feature additions suggested by the mentor is included in section 3 for a better understanding. Basic details of the proposed implementation is covered in section 4 of the document. A rough project schedule is given in section 5. The proposal is concluded in section 6.

2 Digital filters and GNU Radio filters

Digital filters have a long design history from 1960s onwards. The major filter design techniques include Butterworth filter design, Chebyschev approximation, Remez exchange algorithm, Parks-McClellan algorithm (based on remez exchange algorithm) etc. Most of the algorithm classification is based on the nature of the passband and stopband (equiripple, monotonic etc). A brief explanation of some of these filter designs is covered in the implementation section.

GNU Radio provides two ways for designing digital filters. First one by making use of "gr_firdes" which returns taps for windowed FIR filters. The

second method uses "optfir" module which calls "gr_remez" function which implements Parks-McClellan algorithm for finding optimum filter taps. The filter taps thus obtained are given to the filter blocks (gr_fft_filter_ccc etc) which does basic convolution in time to complete the filter functionality. The program "gr_filter_design.py" is a QT frontend (pyqt frontend which uses optfir module) which makes use of these modules to design the filter.

3 Suggested feature improvements

A summary of the feature improvements suggested by the mentor is listed below

- **Estimation of number of taps with the Parks-McClellen algorithm:** The estimation of the order of the filter for given requirements (passband, stopband, magnitude, allowable magnitude deviations etc) is always crucial.
- **Design tools for IIR filters:** Currently there are no design tools available in gnuradio to support design of IIR filters, half band filters etc.
- **Pole-zero plots, complete QA and test code for design algorithms:** Pole zero plots will be very useful in understanding the rationality and stability of the final digital filter. Test codes need to be updated for all filter designs to make sure that the design algorithms are robust.
- **Improved GUI look and feel and an interactive GUI:** This is the prefinal stage of project and this includes addition of filter responses and other features to the graphical user interface once the filter design is complete. Filter responses gives us an idea about how well the designed filter perfroms.
- **GRC integration for gr_filter_design.py:** In the final stage the designing tool should be integrated to GRC so that the filter taps and other values can be imported to filter blocks directly.

4 Implementation details

This section provides a brief description about the implementation specific details. A general filter design procedure is given in Figure 1.

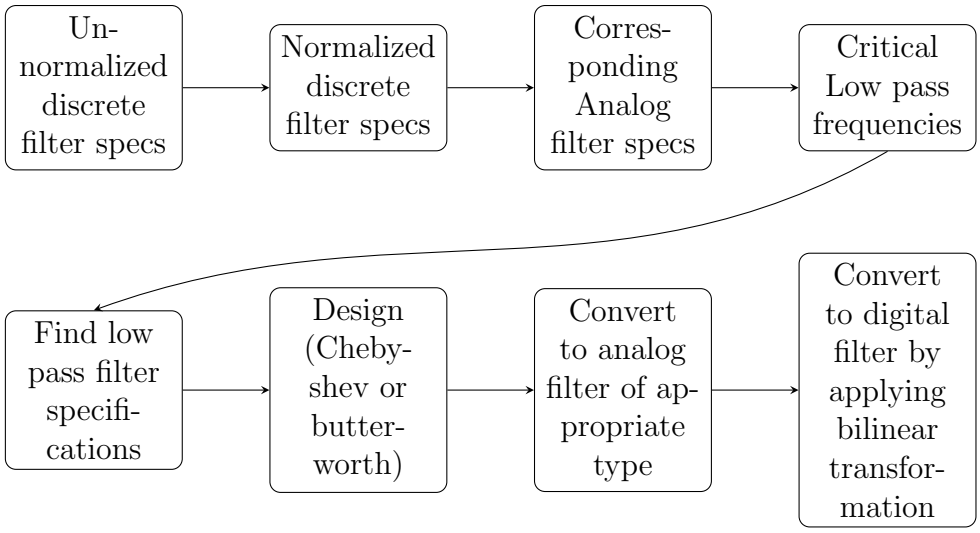

Figure 1: General filter design procedure

Implementation specifics of some of the suggested ideas is included below

- **Estimation of number of taps with the Parks-McClellen algorithm:** Currently there is an implementation of Parks-McClellan taps estimation in optfir module (function name: remezord). Initial stage of implementation will include detailed study of Remez and then Parks algorithm. Details of Remez algorithm and tap estimation can be found in "Theory and Application of Digital Signal Processing" by Rabiner. After study phase proper unit tests will be added to check the proper functionality of the gr_remez and remezord functions. The gr_unittest which is an extension of python standard module Unittest, can be used for adding unit tests.

- **Design tools for IIR filters:** FIR filters overperforms IIR filters when control of phase and instability due to numerical errors are considered. IIR filters usually requires fewer filter coefficients for the same

filtering action when compared to FIR filters and hence it is faster and requires only less memory space. IIR filters are useful in designs where linear phase response characteristics are not required (e.g signal amplitude monitoring applications). Figure 1 gives a general design procedure for IIR filter design. Sample code for IIR and FIR filters can be found in the repository. Half band along with CIC filters are generally used for the implementation of down-samplers in hardware. CIC filters (verilog code) can be found in the FPGA firmware code of USRP. Design tools for half band and cascaded filters will be added after initial discussions with the mentor. Pole zero plots for the designed filters and documentation can be found in the repository (git://github.com/zeroXzero/dsp_filter_design.git).

- **Interactive GUI and GRC integration:** Filter responses can be plotted to analyze designed filter respones and this feature shall be added to the current gr_filter_design.py tool. Filter response plot examples can be found in sample filter design code in github (git://github.com/zeroXzero/dsp_filter_design.git). The author has less experience in designing blocks for GRC (The author personally use normal python code with gnuradio to design new systems and hence some initial learning curve will be there to understand the functioning of GRC). From the GRC documentation it is clear that most of the blocks with a tap parameter automatically import the firdes module. Proper additions should be made to import optfir module and gr_filter_design.py should be incorporated with GRC. This GRC addition will be done after initial discussions with the mentor.

5 Timeline

As per the GSOC schedule there are 18 weeks including midterm evaluation, final evaluation, documentation and code cleanup period. This is just a rough estimate which is very likely to change after discussions with the mentor and the coding time may vary depending on the coding complexity.

- **April 23 - May 4** – Initial learning phase of the algorithms and clarification of implementation with ongoing discussions with the mentor.
- **May 5 - May 11** – Estimation of number of taps with the Parks-McClellen algorithm.

- **May 12 - May 18** – Proper testing by adding good unittests.
- **May 19 - May 25** – Design tools for IIR filters.
- **May 26 - June 1** – Proper testing by adding good unittests.
- **June 2 - June 8** – Addition of Pole-zero plots.
- **June 9 - June 15** – Midterm evaluation submission
- **June 16 - June 22** – Addition of filter response plots.
- **June 23 - June 29** – Addition of good unit tests.
- **June 30 - July 6** – Design tools for other filters.
- **July 7 - July 13** – GRC integration.
- **July 14 - July 20** – GRC integration.
- **July 21 - July 27** – Proper testing of filter modules by adding good unittests.
- **July 27 - Aug 3** – Proper testing of filter modules by adding good unittests.
- **Aug 4 - Aug 10** – Proper testing of filter modules by adding good unittests.
- **Aug 11 - Aug 20** – Integration, code cleanup, adding documentation, and final submission.

6 Conclusion

A general overview of the Filter design enhancement project is given in the previous sections. A sample codeset for FIR and IIR filter designs are also provided in the repository. The project is divided into proper subsections so that the mentor and community can easily track the progress of the project.

References

[1] Theory and Application of Digital Signal Processing, *Rabiner L R.* 1975.

[2] GNU Radio Documentation *http://gnuradio.org/doc/doxygen/modules.html*

[3] Digital Signal Processing and its Applications (EE-603) lecture notes by Prof.Vikram M. Gadre *http://www.ee.iitb.ac.in/wiki/faculty/vmgadre*