Roadmap

*Python Filter Design und Analyse Tool mit Qt GUI (**pyFDA**)*

Version 1.0

Christian Münker

Historie der Dokumentversionen

Version	Datum	Autor	Änderungsgrund / Bemerkungen
1.0	30.10.2013	Prof. Dr. Münker	Ersterstellung aus Aufgabenstellung für Abschlussarbeit

Motivation

Analyse und Entwurf von zeitdiskreten Filtern sind zentrale Aufgaben der digitalen Signalverarbeitung, und nehmen auch in der Lehre entsprechenden Platz ein.

Es gibt bereits zahlreiche Softwarepakete für den rechnergestützten Filterentwurf aus den letzten 30 Jahren, die meisten davon werden aber nicht mehr weiterentwickelt und können nicht mehr ohne weiteres unter aktuellen Betriebssystemen betrieben werden. Die bekannteste aktuelle Software ist das „Filter Design and Analysis Tool" (FDATool, Abb. 1), [1] von Mathworks, ein Graphical User Interface aus der MATLAB® Signal Processing Toolbox.

Das FDATool ermöglicht es, schnell und komfortabel digitale FIR oder IIR - Filter nach vorgegebenen Spezifikationen zu entwerfen. Filter können aus dem MATLAB® Workspace importiert und durch Hinzufügen, Verschieben oder Löschen von Polen und Nullen modifiziert werden. Die Eigenschaften des entworfenen Filters können auf verschiedene Arten dargestellt werden, z.B. durch Betrags- und Phasengang, Impuls- und Stoßantwort, Pol-Nullstellen Diagramme. Bei Fixpoint-Filtern kann zusätzlich das Spektrum des Quantisierungsrauschens angezeigt werden.

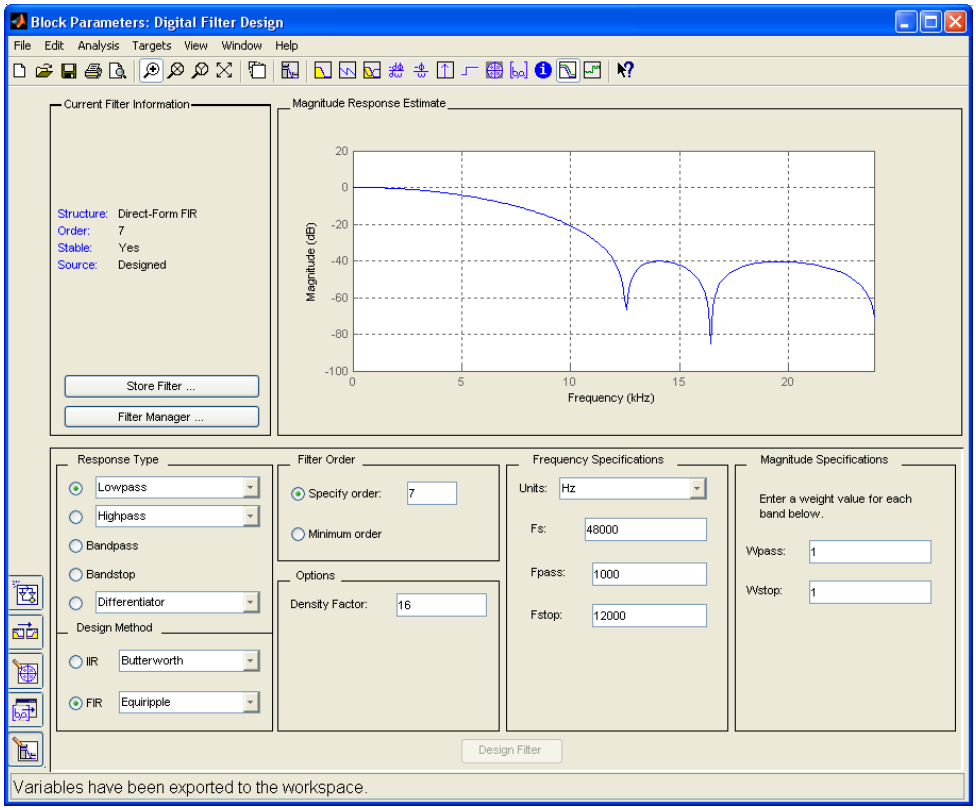

Abb. 1: Screenshot des Mathworks FDATools [1]

Weitere etablierte kommerzielle Produkte mit ähnlichen Möglichkeiten sind z.B. ScopeIIR [2] und ScopeFIR [3] von Iowegian International (Abb. 3 bzw. 4) oder das LabVIEW Filter Design Toolkit [4].

Die Softwarepakete Matlab® und Simulink® der Firma Mathworks haben sich als Quasi-Standard für Systemanalyse und -entwurf für die digitale Signalverarbeitung etabliert. Für den akademischen Bereich hat die Verwendung einer komplexen closed-source Software verschiedene Nachteile:

- Ein Blick hinter die Kulissen zu werfen ist gerade für die Ausbildung sehr interessant
- Für die eigene Forschung und im Rahmen von studentischen Projekten und Abschlussarbeiten kann Open-Source Software leicht erweitert werden
- Zumindest für Mathworks-Produkte sind die Nutzungsrechte der vergünstigten (in Summe immer noch sehr teuren) Academic / Classroom Lizenzen ssehr restriktiv, es dürfen weder In-

dustrie- noch Drittmittel- oder sonstige Forschungsprojekte (z.B. Doktorarbeiten) damit durchgeführt werden.

- Es gibt immer wieder Schwierigkeiten bei der Lizenzvergabe (z.B. für Studierende im Master ohne Hochschulaccount)
- Starke Abhängigkeit von Hersteller, in der Vergangenheit wurden z.B. schon häufiger Lizensierungsmodelle geändert.

Das Interesse von Studierenden und Entwicklern an Filterentwurfssoftware wie dem FDATool ist groß, da Filter eine zentrale Rolle in der Signalverarbeitung spielen. Ein gut gemachtes Filterentwurfstool kann daher auch eine überzeugende Demonstration und Werbung für den Einsatz von Open-Source Software sein.

Für die Entwicklung wurde Python als „Lingua Franca" der Open Source Community für Scientific Computing ausgewählt, da es hier bereits umfangreiche Bibliotheken für die Signalverarbeitung gibt; die Module numPy / sciPy [9] und matplotlib [10] stellen z.B. sehr viele Funktionen von Matlab zur Verfügung. Ein Skript ohne GUI mit Grundfunktionen des Filterdesigns wurde bereits von mir erstellt.

Mit PyQT können auch leistungsfähige Graphical User Interfaces (GUIs) konstruiert werden, Binaries können leicht für verschiedene Betriebssysteme kompiliert werden.

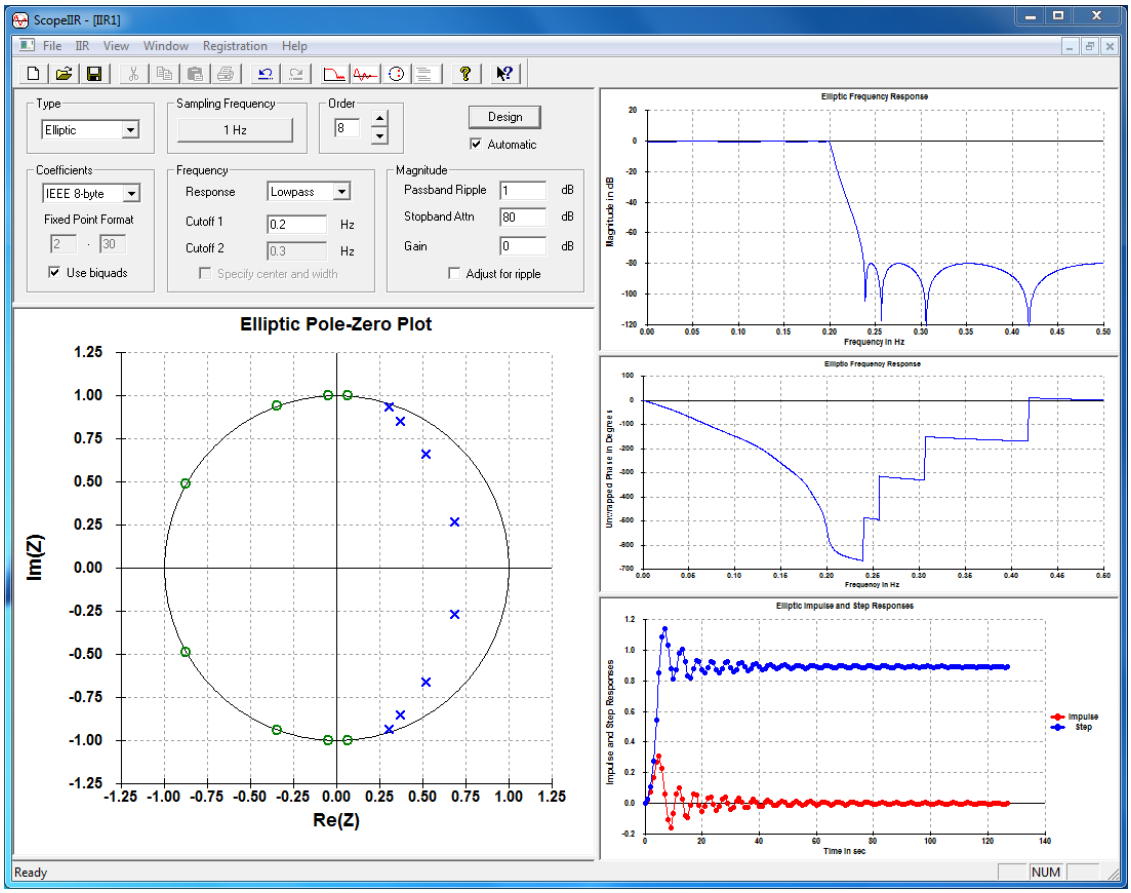

Abb. 2: Screenshot von ScopeIIR [2]

Interaktive Java-Applets sind eine gute Möglichkeit, schnell und ohne Installation mit verschiedenen Filtertypen und -optionen „herumzuspielen". Ein sehr gelungenes Applet findet man z.B. unter [8]. Wie bei den meisten Applets steht allerdings auch bei diesem die Lehre im Vordergrund; zum ernsthaften Filterdesign eignet es sich mangels Import- und Exportmöglichkeiten nicht.

Auf GitHub wurde ein Repository eingerichtet, auf dem der Code gemeinsam weiterentwickelt wird.

1 Anforderungen und Zielstellung

Es soll eine Applikation erstellt werden, die die Vielzahl von Filterspezifikationen, Entwurfs- und Analysemethoden, Filtermodifikationen und Exportarten für den Benutzer übersichtlich strukturiert. Ein möglicher und üblicher Ansatz **für das GUI** ist eine

Workflow-orientierte Struktur (vgl. Microsoft Ribbons)

die weitgehend auch für das FDATool verwendet wurde (siehe Screenshots).

Aufgrund der vielfältigen Ein- und Ausgabeoptionen ist **für die Software** vermutlich eine

Model-View-Architektur

sinnvoll [13].

Generell ist es wichtig, den Code so zu gestalten, dass die Arbeit leicht als Open Source Projekt weitergeführt werden kann. Übersichtlichkeit und Struktur ist wichtiger als Vollständigkeit!

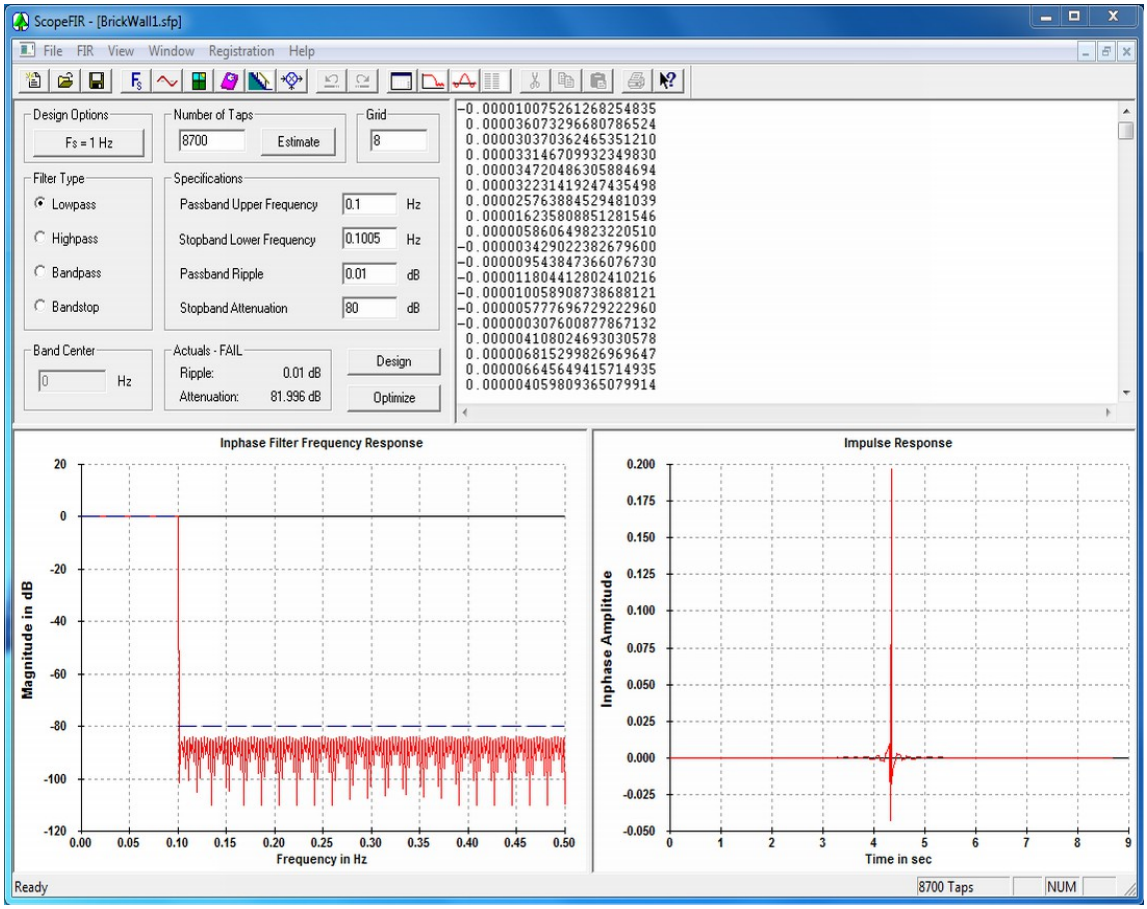

Abb. 3: Screenshot von ScopeFIR [3]

Mathworks fdatool als OpenSource Projekt 1:1 nachzubauen, erscheint wenig sinnvoll, da sich aus den geplanten Einsatzfeldern andere Anforderungen ergeben. (+) kennzeichnet Features, die das fdatool nicht bietet:

1.1 Allgemeine Eigenschaften

- Filter-Manager zum Speichern und Verwalten verschiedener Entwürfe
- Automatische Dokumentation der Filtereigenschaften als PDF / PNG / ODT … (**+**)
- Kontext-Hilfe und eine spätere Internationalisierung soll möglichst von vorneherein eingeplant werden, z.B. mit Hilfe von Qt Linguist

- Entwurf und Visualisierung von Multiraten-Filtern: Gesucht sind gute Visualisierungen und Entwurfsalgorithmen für Lehre und Labor

1.2 Lehre

- einfache, übersichtliche Bedienung, lieber auf ein paar Optionen verzichten. Auch Aufruf / Installation sollen möglichst einfach sein, daher sollen Binaries zur Verfügung gestellt werden.
- gleichzeitige Darstellung von zwei Plots z.B. Zeitbereich oder PN-Diagramm und Frequenzbereich **(+)**
- Visualisierung der Eigenschaften verschiedener Filtertypen und der Konsequenzen der Quantisierung. Hierfür ist es sehr hilfreich, wenn mehrere Entwürfe gleichzeitig dargestellt werden können **(+)**
- Audio-Unterstützung: Wie „klingt" ein Filter? Quellen: Rauschen, Mehrton-Sinus, eigene Audiosignale **(+)**

1.3 Labor Schaltungstechnik und uC / FPGA-Entwickler

Nicht nur Studierende, sondern auch Hobby-Entwickler mit geringen Vorkenntnissen in digitaler Signalverarbeitung sind oft bereits mit einfachen Filterentwürfen überfordert. Das gilt vor allem, wenn es um die Abschätzung der Effekte begrenzter Wortlängen geht (Überläufe, eingeschränkter Dynamikbereich, Einfluss verschiedener Filtertopologien) und um die Implementierung (VHDL, C, …).

- Entwurf effizienter rekursiver Filter (Multistage (S)CIC-Filter, CSD-Filteroptimierung, Resonator-Filter, u.U. WDF, Filter in ΣΔ-Arithmetik, Lattice-Filter, ...): Für diese Filtertypen gibt es kaum Entwurfssoftware, die meisten Entwicker wissen nicht einmal, dass es Filter gibt, die nur einen Bruchteil der Ressourcen eines Direktform-Filters benötigen **(++)**
- Netzlisten – Export nach VHDL, Verilog, C etc. **(+)**
- Performance-Abschätzung von Dezimatorfiltern für verschiedene Rauschspektren (PCM, $\Sigma\Delta^N$, …) **(+)**
- Filter für Asynchronous Sample Rate Conversion (ASRC) **(+)**

1.4

2 Struktur und Funktionen - Milestones

2.1 Entwurf / Import

Hier kann der Filtertyp (IIR / FIR) ausgewählt werden, Filterentwurfsmethode, Spezifikationen. Bestimmte Eingabefelder sollen dabei mit bestimmten Entwurfsmethoden gekoppelt sein. Um mehrere Filterentwürfe schnell mit einander vergleichen zu können, können mehrere Entwürfe gespeichert und schnell abgerufen werden.

2.2 Analyse

Die folgenden Analysen können dargestellt werden:

- Betrags- und Phasengang
- Gruppenlaufzeit
- Stabilität (*)
- Impuls- und Stoßantwort
- Pol-Nullstellen Diagramm
- Filterkoeffizienten (textbasiert)

- Anzeige der Fensterungsfunktion bei fensterbasierten Methoden

2.3 Quantisierung und Filterstruktur

- **Koeffizientenquantisierung:** Hier kann das Fixpoint-Format der Koeffizienten gewählt und anschließend die Koeffizienten quantisiert werden. Eigenschaften von quantisiertem und unquantisiertem Filter können verglichen werden
- **Quantisierte Arithmetik:** Für das Signal-Rauschverhältnis von Fixpointfiltern sind die Fillter-topologie und die internen Wortlängen (z.B. Reduktion der Wortlänge nach Multiplikationen) entscheidend. Daher müssen für jeden Filtertyp **interne Zahlenformate** (Akkumulatorlänge, Quantisierer) und **Filtertopologie** (Direktform, Transponierte Direktform, Biquad, Noiseshaping, …) gewählt werden
- **Quantisierungsrauschen** von verschiedenen Filtertopologien und Wortlängen: Für schnelle Simulationen muss für jeden Filtertyp ein statistisches Rauschmodell erstellt werden

Aufgrund der vielfältigen Anforderungen sollte spätestens in diesem Schritt eine objektorientierte Implementierung der verschiedenen Filter gewählt werden.

2.4 Export

Ausgehend von den Filterkoeffizienten und der gewählten Topologie können „Codeschnipsel" in C, Python oder VHDL/Verilog generiert werden und in eigene Projekte eingebunden werden. Dies wäre ein wichtiger USP für die Entwicklercommunity von Microcontrollern oder FPGA-Projekten!

3 Arbeitspakete

- **„Spike" zu erster Version der Designsoftware und folgenden Fragen:**
  - Welche Interfaces benötige ich zwischen Eingabe-Entwurf-Analyse?
  - Wie speichere ich Filter ab?

3.1 GUI-Design

Aufbauend auf den Python Bindings PyQt / PySide gibt es weitere High-Level Libraries speziell zur Erstellung von wissenschaftlicher Software:

- **Matplotlib** [10] kann auch zur Erstellung von GUIs verwendet werden.
- **Qwt** [17] ergänzt Qt um Widgets für technische Anwendungen, d.h. 2D-Plots und verschiedenste Eingabeelemente für Real-Werte. **PyQwt** [16] stellt dazu die Python - Bindings zur Verfügung, es integriert PyQt, Qt, Qwt, NumPy und SciPy. Allerdings gibt es keine Anbindung an PySide. Für 3D Visualisierungen bietet **QwtPlot3D** mit **PyQwt3D** vergleichbare Funktionalität. **guiqwt** [12] baut auf PyQwt auf und bietet zusätzliche Funktionen für Bild- und Signalverarbeitung (?). Problem: Zu viele Abhängigkeiten (quiqwt → pyqwt / pyqt → qwt → qt), keine Unterstützung von PySide.
- **PyQtGraph** [18] ergänzt Qt mit PyQt / PySide um schnelle (Real-time), interaktive Plots und Images (dank numpy und Qts , Außerdem sind Tools für Rapid Application Development (ParameterTree widget) enthalten und ein gegenüber Qt erweitertes Docking System für GUIs. Probleme: Dokumentation, User Base, Funktionsumfang?

3.2 Audio I/O:

- Evaluierung von verschiedenen Audio I/O – Modulen (pyAudio/pyLame, fastaudio, http://audiotools.sourceforge.net/, Phonon für Qt …)
- Generierung verschiedener Testsignale (Rauschen, SD-QNoise, Mehrton-Sinus, eigene Audiosignale)

  - Implementieren eines einfachen Recorders / Players. Wichtig auch für Multirate / ASRC !
- **Automatische Erzeugung von Datenblättern (HTML / PDF / ODF)**
- **Interaktive Manipulation von Pol/Nullstellen**
- **Fixpoint-Funktionalität**
  - Finden einer geeigneten Fixpoint-Library
  - Koeffizienten-Quantisierung
  - Arithmetik-Quantisierung / Erstellung verschiedener parametrisierbarer Filter-Topologien in Python
  - Einbinden von myHDL für Netzlistenexport
- „Spike" Multirate / Polyphasenfilter-Design und Visualisierung
- „Spike" ASRC Simulation und Design

4 Verwandte Projekte

- pyo: http://code.google.com/p/pyo/
- pyDSP: http://pypi.python.org/pypi/PyDSP
- gnuradio.org: http://www.youtube.com/watch?v=N9SLAnGlGQs&list=PL618122BD66C8B3C4
- gnuRadio Binaries: http://gnuradio.org/redmine/projects/gnuradio/wiki/WindowsInstall
- gnuRadio Filterdesign: http://www.youtube.com/watch?v=20ZrfUZjUUA

5 Referenzen

[1] http://www.mathworks.de/de/help/signal/ug/overview.html#br179zi-4

[2] http://www.iowegian.com/scopeiir

[3] http://www.iowegian.com/scopefir

[4] http://www.ni.com/pdf/labview/us/final_dfd_tutorial.pdf

[5] http://www.gnu.org/software/octave/

[6] http://www.scilab.org/

[7] http://www.fh-kl.de/~hans.neuschwander/download_skripte/digfil/DigFilter.pdf

[8] http://www.falstad.com/dfilter/

[9] http://www.scipy.org

[10] http://matplotlib.org/

[11] http://packages.python.org/guiqwt/sift.html

[12] http://code.google.com/p/guiqwt/

[13] http://openbook.galileocomputing.de/python/python_kapitel_24_007.htm

[14] http://www.daniweb.com/software-development/python/threads/191210/python-gui-programming

[15] http://eli.thegreenplace.net/2009/01/19/moving-to-pyqt/

[16] http://pyqwt.sourceforge.net/

[17] http://qwt.sourceforge.net/

[18] http://luke.campagnola.me/code/pyqtgraph/

[19] http://code.enthought.com/projects/traits/

[20] http://docs.enthought.com/traitsui/

[21] http://code.enthought.com/projects/traits/docs/html/tutorials/traits_ui_scientific_app.html

[22] http://docs.enthought.com/chaco/

[23] http://gnuradio.squarespace.com/home/2012/12/19/update-on-filter-design-tool.html