BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

Realisierung und Analyse des IIR-Filters

Laborbericht

angefertigt von

Robby Kozok, Nic Frank Siebenborn, Pascal Kahlert

in dem Fachbereich VII – Elektrotechnik - Mechatronik - Optometrie –
für das Modul Digitale Signalverarbeitung III
der Beuth Hochschule für Technik Berlin im Studiengang
Elektrotechnik - Schwerpunkt Elektronische Systeme

Datum 4. Januar 2016

Lehrkraft
Prof. Dr.-Ing Marcus Purat Beuth Hochschule für Technik

Inhaltsverzeichnis

Kapitel 1

Realisierung des IIR-Filters

1.1 Aufgabenstellung

Die Aufgabe dieser Übung bestand darin, einen IIR Filter zu realisieren. Dabei galt besonderes Augenmerk der Minimierung von Rundungsfehlern sowie der Verhinderung von Überläufen.

1.2 Durchführung

Das Filter wird auf die nachfolgenden vier Varianten implementiert. Vorab wurden dazu die Filterkoeffizienten entsprechend der Aufgabe nach aufsteigendem Radius im Einheitskreis sortiert.
Es wurden wie in der Aufgabe beschrieben die Dateien in ein neues Projekt eingefügt und wie im folgenden beschrieben entsprechend angepasst.

1.2.1 Skalierung am Eingang

Die einfachste Form der Skalierung ist die Skalierung am Eingang, wobei mit einem Diviesor vor verwenden des Filters dividiert wird. Dazu wird zuerst die gesamte Verstärkung der Filter durch Multiplikation ermittelt. Da es sich um einen Tiefpass handelt wird die Verstärkung bei 0Hz berechnet. Es folgt: z=1. Gemäß folgender Formel ergibt sich eine Gesamtverstärkung von 474.5917.

$$\Pi H_i(1) = 38,3655 * 4,8221 * 1,8921 * 1,3558 = 474.5917 \tag{1.1}$$

Todo: robby bitte die werte anpassen so dass die rechnung stimmt hier stimmt nur das ergebnis mit unserem sourcecode überein. Daraus folgt die Anpassung der Datei process_data.c wie folgt.

```
1  // Definition der Filterkoeffizienten
2  #define BIQUAD_STAGES 4
3
4  // coef includes for all stages a scaling factor (2^s) and five coefficients
5  // in the order s, b2, b1, b0, a2, a1
6
7  // No scaling in stages (Low Pass)
8  const short coef[6*BIQUAD_STAGES] = {
9          1, 16384, 6581  , 16384, -10390, 25748  //H4
10         1, 16384, -20909, 16384, -12529, 26454, //H3
11         1, 16384, -25583, 16384, -14605, 27191, //H2
12         1, 16384, -26706, 16384, -15896, 27808, //H1
13 };
14
15 int iDelayline[2*BIQUAD_STAGES+2];      // delayline for left and right channel samples
16
17 IIRstateStereo iirLR={coef,iDelayline,BIQUAD_STAGES};
18
19 void process_data()
20 {
21         //!! Scale before filtering
22         sADC1L = sADC1L / 474.5917; //skalierungsfaktor aus Rechnung
23         sADC1R = sADC1R / 474.5917;
24         *(int*)(&sDAC1L) = iir_stereo(*(int*)(&sADC1L),&iirLR);
25
26         //!! Scale after filtering
27 }
```

processdata.c

Die Interprätierung als 1.15 Format findet sich in den eingetragenen Koeffizienten nicht, sie wurden als short implementiert. Die Konvertierung erfolgte wie in früheren Laborübungen gesehen. Die Implementierung als short ermöglicht eine bessere Übersichtlichkeit und vermeidet Fehler. Die Zweite Anpassung war die eigentliche Skalierung am Eingang, hierzu wurde der zuvor ermittelte Divisor verwendet.

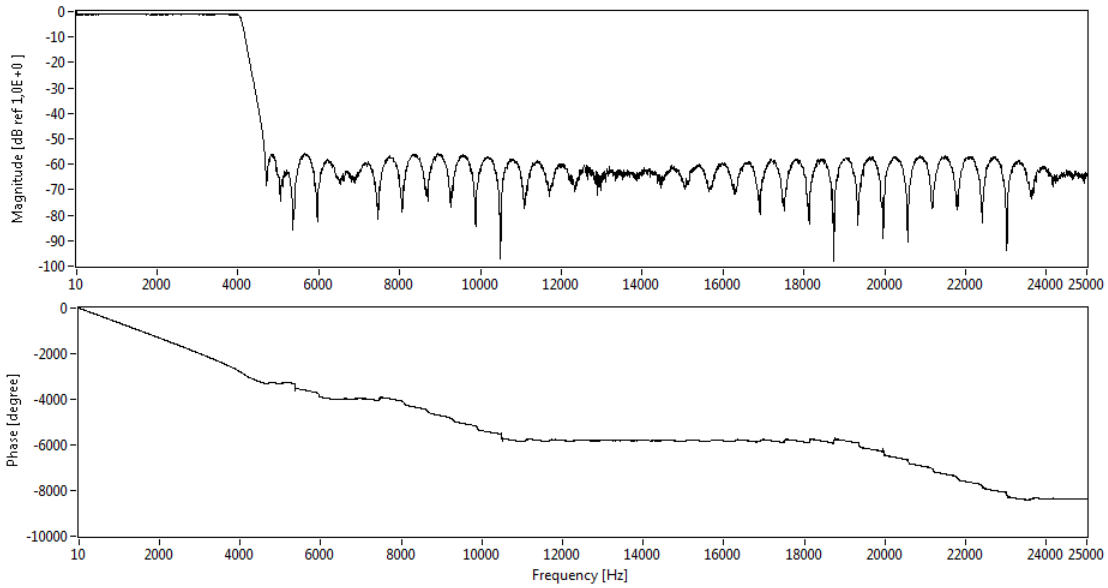

Abbildung 1.1: Frequenzgang

Hier wurde der Frequenzgang des Systems visualisiert, es ist deutlich das Tiefpassverhalten auszumachen. Die Grenzfrequenz liegt bei etwa 4kHz.
Zur weiteren Untersuchung wurde folgendes Sinussignal mit 2kHz Frequenz und 50mV Amplitude eingespeist.

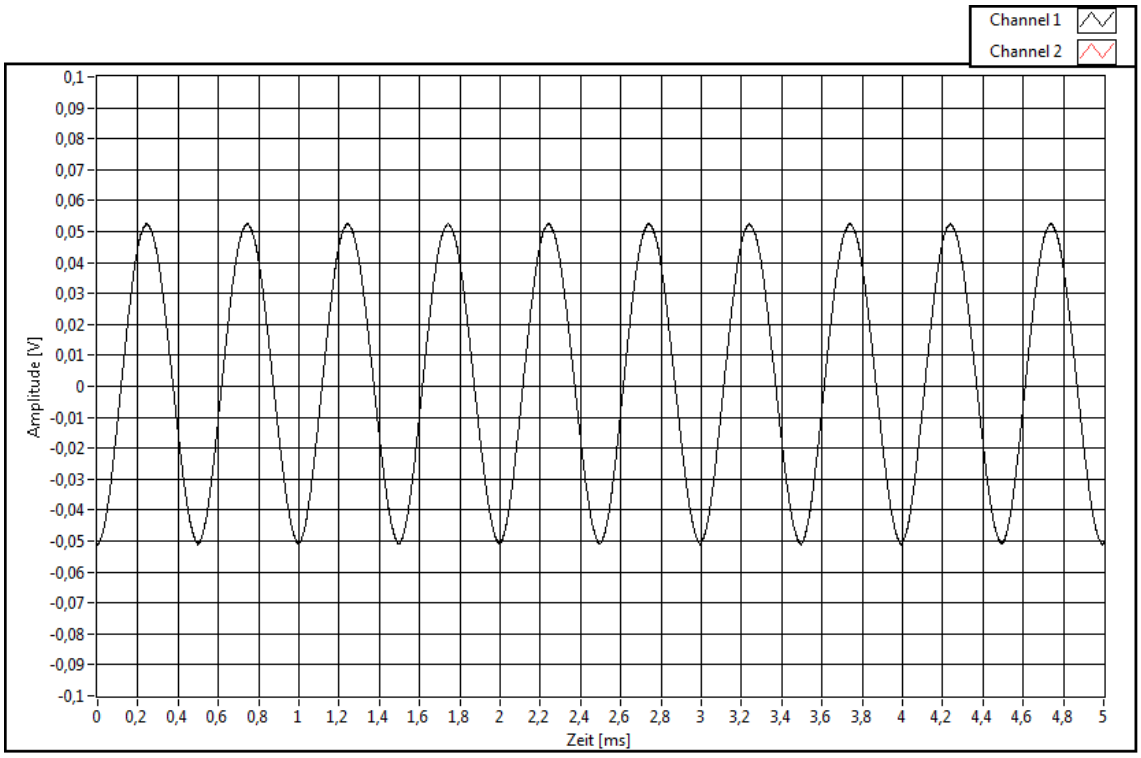

Abbildung 1.2: Eingangssignal

Wir haben am Eingang das nachfolgende Spektrum gemessen.

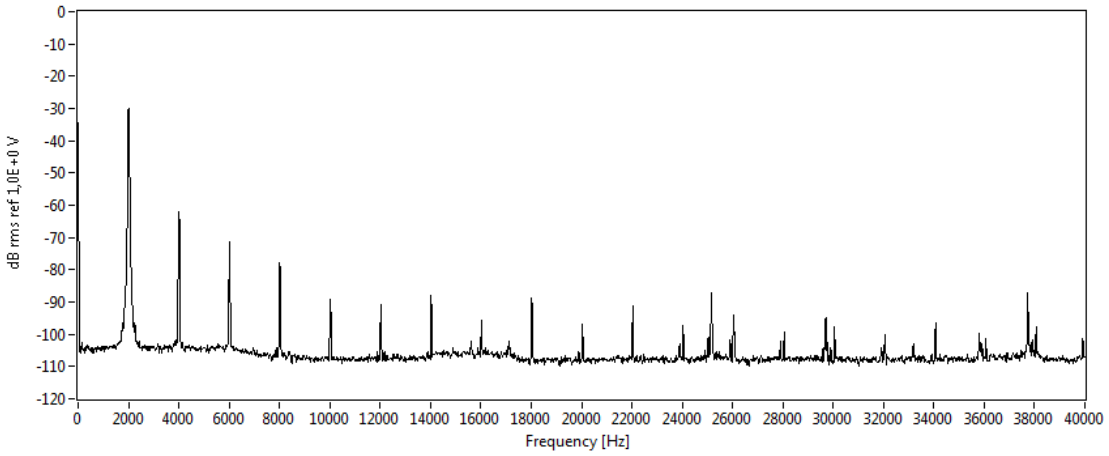

Abbildung 1.3: Spektrum am Eingang

Und am Ausgang hat man in nachstehendem Spektrum das Rauschen gut beobachten können.

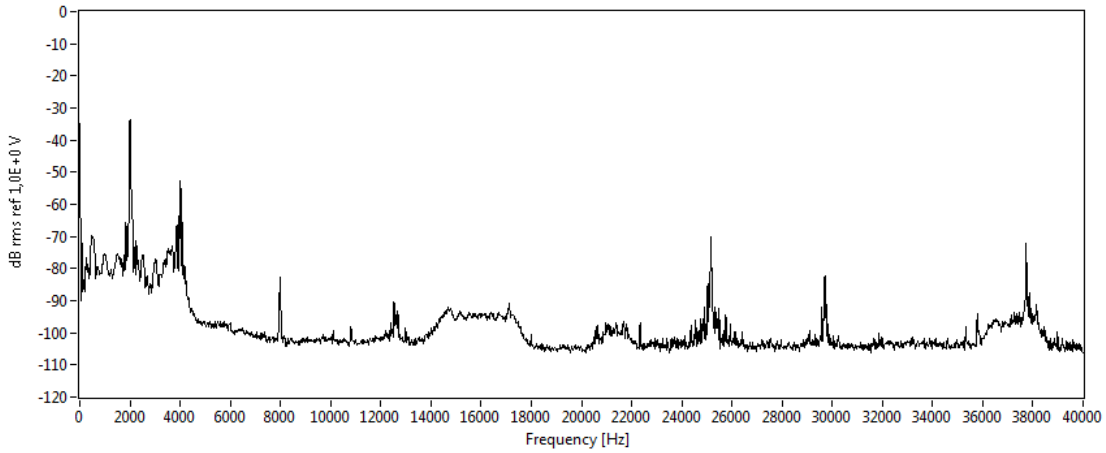

Abbildung 1.4: Spektrum am Ausgang

Es ist gut zu sehen, dass am Eingang mehrere Harmonische alle 2kHz zu sehen sind, umgeben von einem sehr geringen Rauschteppich bei unter -100dB. Am Ausgang hingegen ist das Tiefpassverhalten zu erkennen, was die erste Harmonische bei 4kHz (der Grenzfrequenz) noch leicht gedämpft darstellt, alle weiteren Harmonischen aber hinreichend unterdrückt. Außerdem lässt sich ein Rauschen ausmachen, was bis zur Grenzfrequenz um -80dB liegt. Dieses Bild entspricht unseren Erwartungen.

1.2.2 Skalierung am Ausgang

Entsprechend der Aufgabenstellung haben wir auch die Skalierung am Ausgang umgesetzt, wie erwartet übersteuerte das System massiv, was die Analyse der Implementierungsvariante unnütz machte. Aus diesem Grund verzichten wir wie Abgesprochen auf eine Auswertung an dieser Stelle.

1.2.3 Gleichmäßige Skalierung

Im dritten Teil der Übung wurde das Filter gleichmäßig über alle Teilfilter skaliert. Dazu wurde die Gesamtverstärkung aus dem ersten Teil gleichmäßig auf die Teilfilter aufgeteilt. Der Divisor ergibt sich also wie folgt:

$$\sqrt[4]{474.5917} = 4,6675 \tag{1.2}$$

Bereits in der Vorbereitung wurden die Filterkoeffizienten durch diesen Quotienten geteilt und in das short Format gebracht. Somit ergibt sich folgende Änderung:

```
1
2   // No scaling in stages (Low Pass)
3   const short coef[6*BIQUAD_STAGES] = {
4           1, 3514, 1412, 3514, -2229, 5523, //H4
5           1, 3514, -4485, 3514, -2687, 5674,//H3
6           1, 3514, -5488, 3514, -3133, 5832,//H2
7           1, 3514, -5728, 3514, -3410, 5965 //H1
8   };
9
10
11
12  int iDelayline[2*BIQUAD_STAGES+2];      // delayline for left and right channel samples
13
14  IIRstateStereo iirLR={coef,iDelayline,BIQUAD_STAGES};
15
16  void process_data()
17  {
18          //!! Scale before filtering
19          *(int*)(&sDAC1L) = iir_stereo(*(int*)(&sADC1L),&iirLR);
20          //!! Scale after filtering
21
22  }
```

processdata.c

Es ist auch ersichtlich, dass die Skalierung am Ende im Gegensatz zur vorherigen Aufgabe entfernt wurde.
Bei dieser Art der Implementierung erwarten wir ab einer gewissen Amplitude Übersteuerungsfehler. So haben wir bei immer 2kHz verschiedene Amplituden auf das Signal gegeben und sind so sukzessive zu folgenden drei Bildern gelangt.

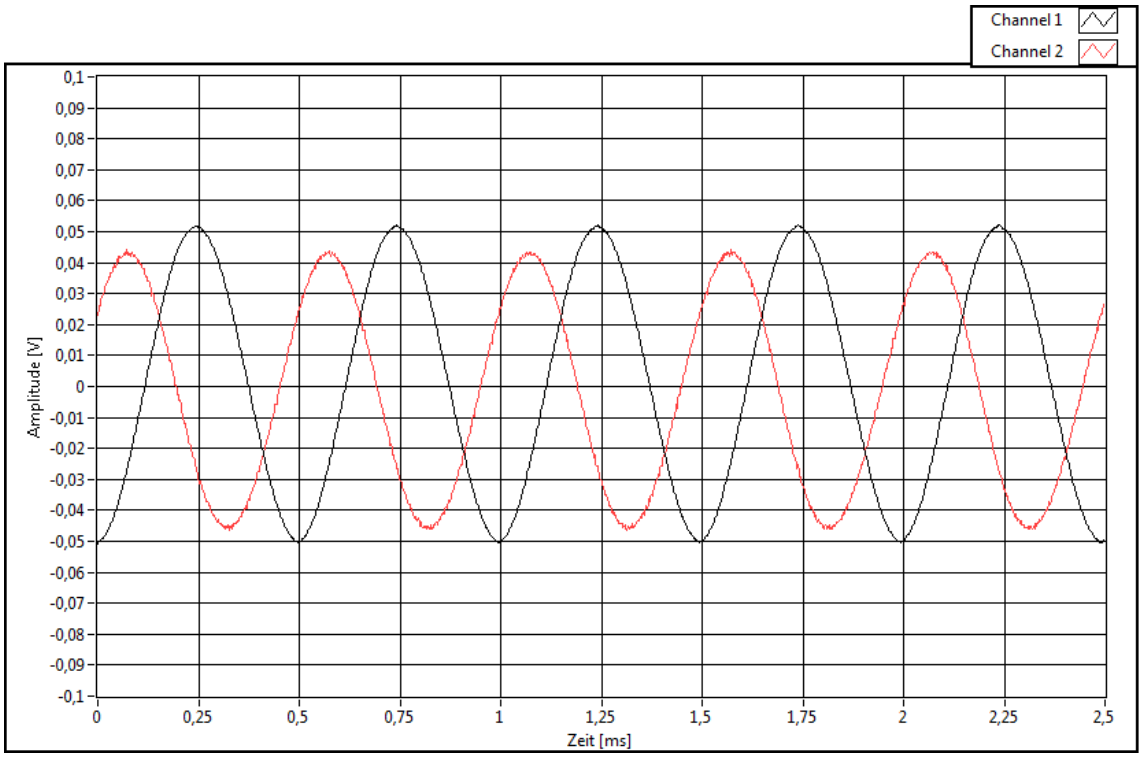

Abbildung 1.5: 50mV Amplitude

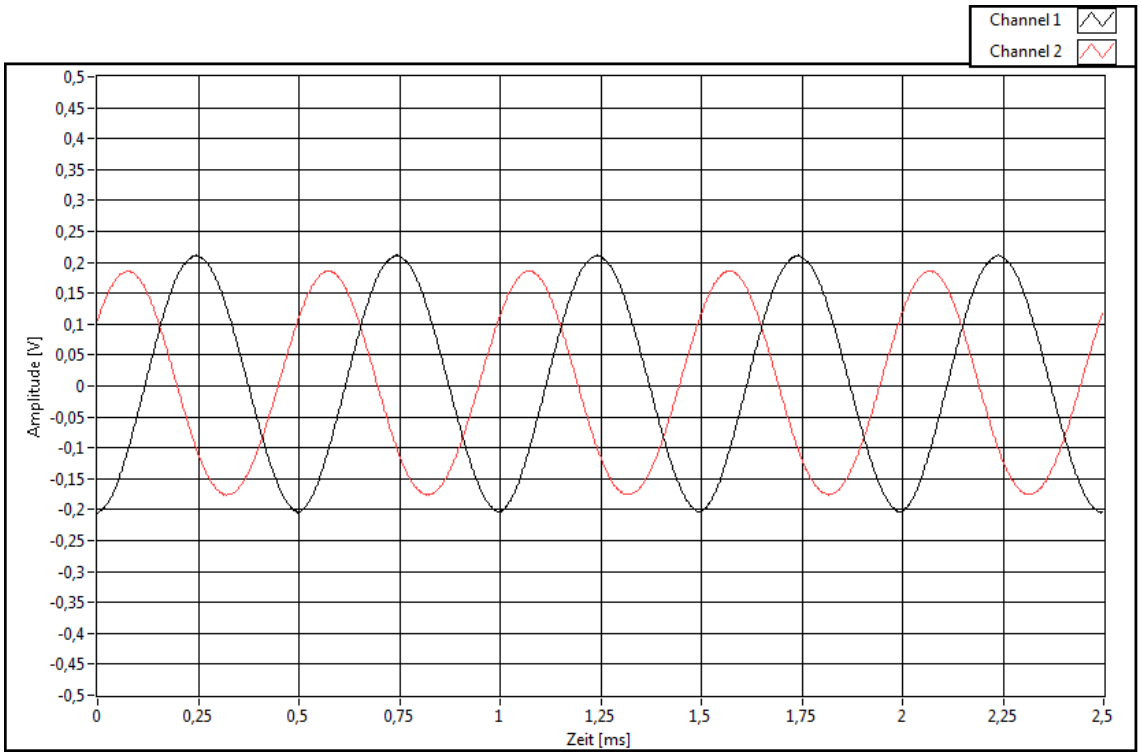

Abbildung 1.6: 200mV Amplitude

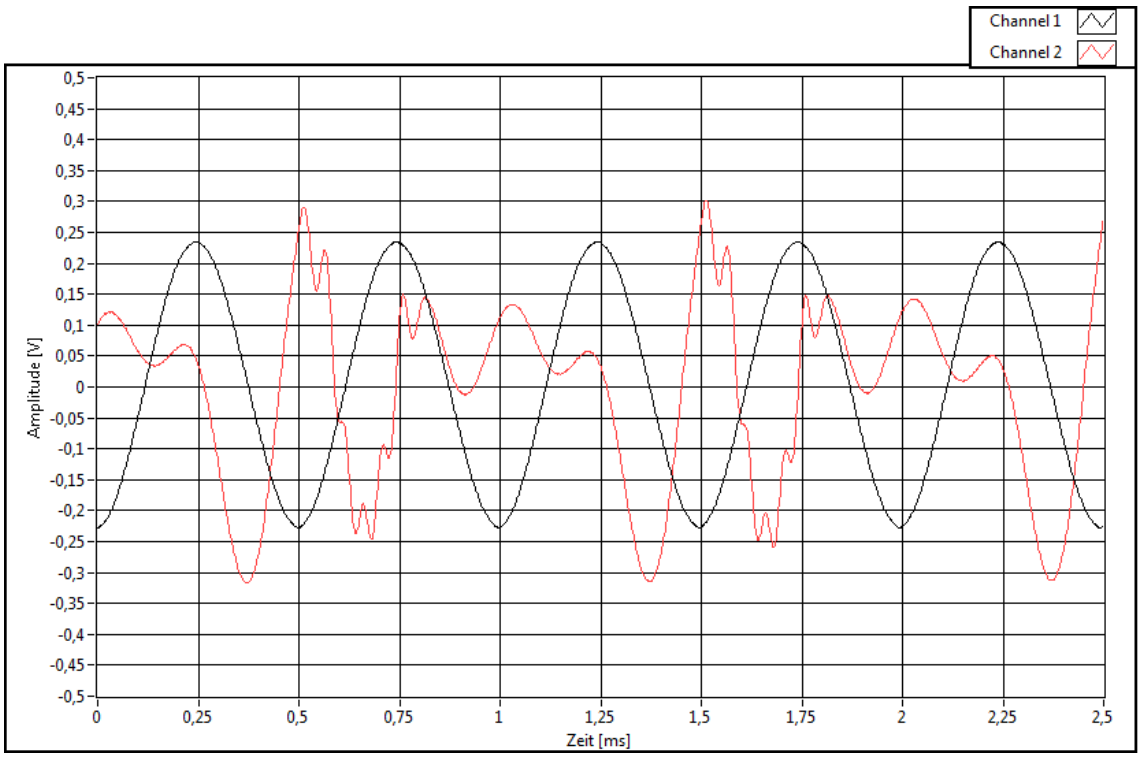

Abbildung 1.7: 230mV Amplitude

Man sieht, dass das Ausgangssignal(rot) jeweils leicht gedämpft dem Eingangssignal entspricht außer bei 230mV Amplitude am Eingang. Der Erwartete Übersteuerungseffekt tritt also zwischen 200mV und 250mV auf. Dieses Ergebnis ist bei weitem besser als im

Fall der Skalierung am Eingang.
Wir haben für die Fälle 50mV und 230mV die Spektren am Ausgang aufgenommen.

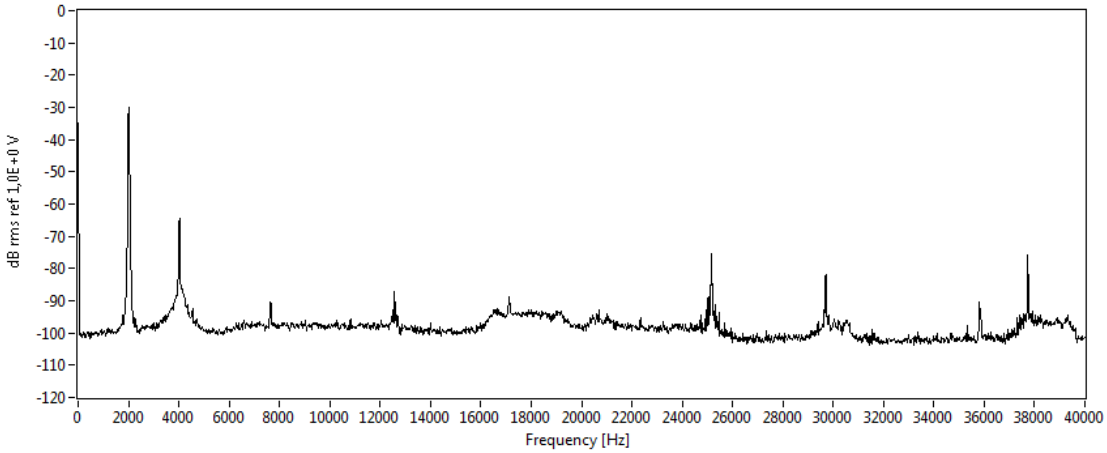

Abbildung 1.8: Spektrum am Ausgang 50mV

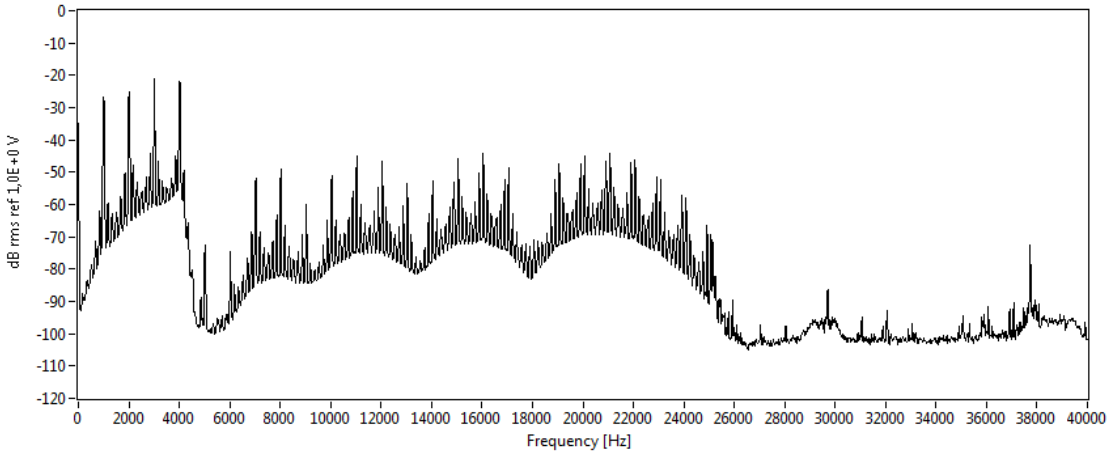

Abbildung 1.9: Spektrum am Ausgang 230mV

Bei dem Spektrum bei den 50mV ist das Ergebnis besser als bei der Skalierung am Eingang, so ist ein geringerer Rauschteppisch sowie eine Stärkere Unterdrückung der höheren Harmonischen zu erkennen. Die erste Harmonische ist wie erwartet leicht gedämpft gut erkennbar. Das Spektrum der Variante mit 230mV ist wie erwartet nicht sehr eindeutig. Mit Fantasie erkennt man ein Tiefpassverhalten mit einer Grenzfrequenz um 4kHz, weitere Deutung lässt die Abbildung nicht zu.

1.2.4 Optimierte Skalierung

Der letzte Optimierungsschritt ist die Skalierung an jedem Teilfilter opitmal passend zum Filter. Dafür wurde folgendes Matlabfile erstellt.

```
 1   H1 = [-1 0 1 -0.9619429222893 1.847929303926];
 2  H2 = [-1 0 1 -0.9633409060843 1.81768444585];
 3  H3 = [-1 0 1 -0.9836465239584 1.819862361978];
 4  H4 = [-1 0 1 -0.9850722601925 1.806715933748];
 5
 6  H=abs(freqz(H1(1:3),[H1(4:6)]));
 7  g1=1/max(H);
 8  H=g1*H;
 9  H=H.*abs(freqz(H2(1:3),H2(4:6)));
10  g2=1/max(H);
11  H=g2*H;
12  H=H.*abs(freqz(H3(1:3),H3(4:6)));
13  g3=1/max(H);
14  H=g3*H;
15  H=H.*abs(freqz(H4(1:3),H4(4:6)));
16  g4=1/max(H);
17  H1n = (g1 .* H1(1:3) * 2^15)/2;
18  H2n = (g2 .* H2(1:3) * 2^15)/2;
19  H3n = (g3 .* H3(1:3) * 2^15)/2;
20  H4n = (g4 .* H4(1:3) * 2^15)/2;
```

optimierung.m

Kapitel 2

Analyse des FIR-Filters

2.1 Aufgabenstellung

In dieser Aufgabe sollte zunächst die 3dB-Grenzfrequenz des Tiefpass-Filters ermittelt werden. Im weiteren Verlauf sollte mithilfe des Filter Design and Analysis (FDA) - Tools ein IIR-Bandpass-Filter entworfen werden. Folgende Charakteristik ist dabei zu erreichen:

- Butterworth-Charakteristik
- Stoppband-Frequenzen: 2000Hz, 4000Hz
- Passband-Frequenzen: 2000Hz, 4000Hz
- Passband-Welligkeit: 1dB
- Stoppband-Dämpfung: 60dB

2.2 Durchführung

IIR-Tiefpass-Filter

Zur Ermittlung der 3dB-Grenzfrequenz sollte die Frequenz so eingestellt werden, dass das Ausgangssignal 70% der Amplitude des Ausgangssignals bei 50Hz entspricht. Dafür wurde die Amplitude bei 50Hz auf 1V eingestellt, danach haben wir uns das Spektrum des Systems angeschaut.

Dies ist in Abbildung 2.1 zu sehen. Dort sehen wir 3dB-Dämpfung bei 4060Hz. Dies entspricht den Erwartungen, da der Filter seine Grenzfrequenz bei ungefähr 4kHz haben sollte.

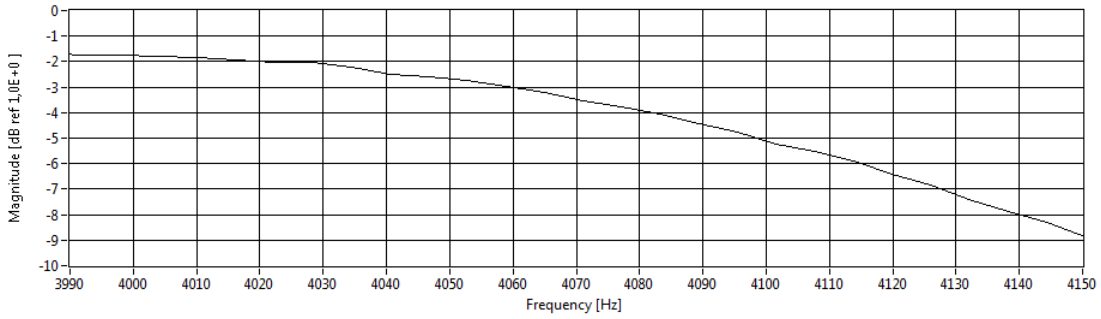

Abbildung 2.1: Ausschnitt des Spektrums des Systems

Im weiteren haben wir dann eine Aufnahme mit dem VI gemacht. Diese ist in Abbildung 2.2 zu sehen, dieses Ergebnis war zu erwarten. Es ist zu sehen das das rote Ausgangssignal bei ungefähr 0,7V liegt und damit 70% der ehemals 1V hat. Die Frequenz wurde auf 4060Hz eingestellt.

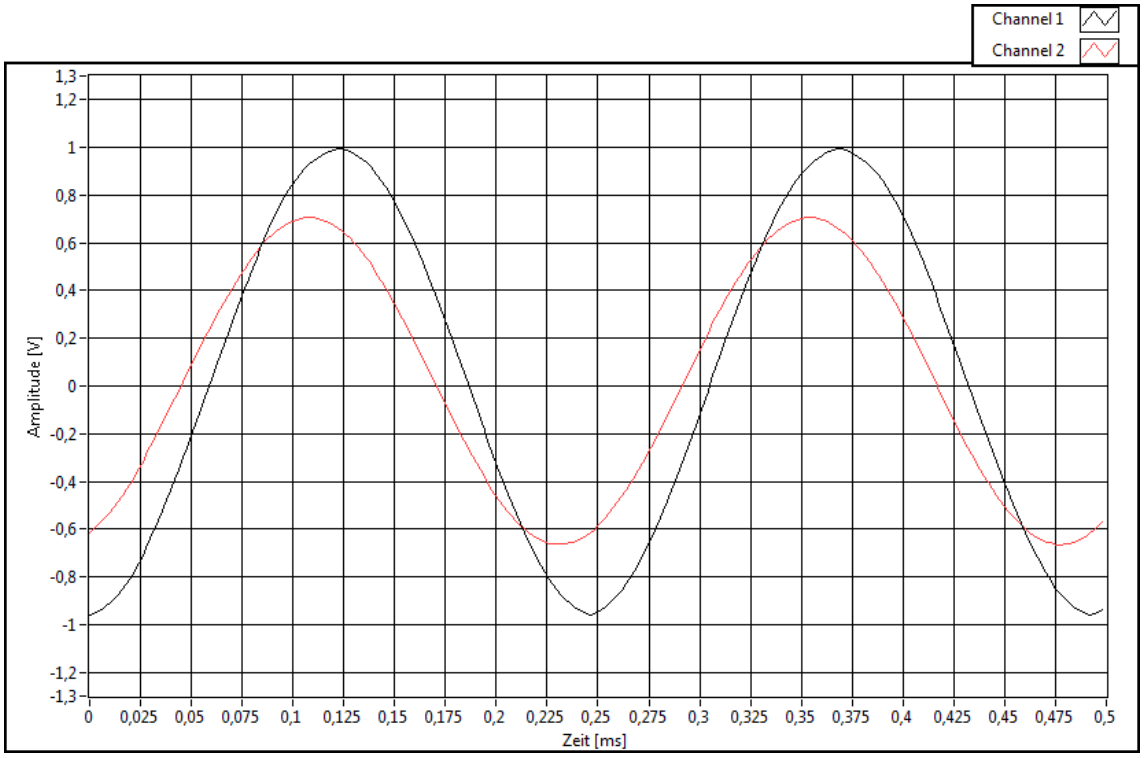

Abbildung 2.2: Ein- und Ausgangssignal nebeneinander

IIR-Bandpass-Filter

Der zweite Teil behandelte die Erstellung IIR-Bandpass-Filters. Zu allererst mussten dafür die Koeffizienten generiert werden. Diese wurde dann mit Matlab abermals optimiert.

```
1  %Hi = [1 b1 b2 1 a1 a2] -  die von Matlab vorgegebene Form
2  H1 = [1 0 -1 1 -1.847929303926 0.9850722601925 ];
3  H2 = [1 0 -1 1 -1.81768444585  0.9836465239584 ];
4  H3 = [1 0 -1 1 -1.819862361978 0.9633409060843 ];
5  H4 = [1 0 -1 1 -1.806715933748 0.9619429222893 ];
6
7  %Berechnung der Teilverstärkungen
8  H=abs(freqz(H1(1:3),H1(4:6)));
9  g1=1/max(H);
10 H=g1*H;
11 H=H.*abs(freqz(H2(1:3),H2(4:6)));
12 g2=1/max(H);
13 H=g2*H;
14 H=H.*abs(freqz(H3(1:3),H3(4:6)));
15 g3=1/max(H);
16 H=g3*H;
17 H=H.*abs(freqz(H4(1:3),H4(4:6)));
18 g4=1/max(H);
19
20 %Die von unserem Filter benötigte Form
21 H1 = [-1 0 1 -0.9850722601925 1.847929303926]/2;
22 H2 = [-1 0 1 -0.9836465239584 1.81768444585 ]/2;
```

```
23 H3 = [-1 0 1 -0.9633409060843 1.819862361978]/2;
24 H4 = [-1 0 1 -0.9619429222893 1.806715933748]/2;
25
26 %Teilverstärkungen anwenden
27 for i=1:3
28     H1(i) = H1(i) * g1;
29     H2(i) = H2(i) * g2;
30     H3(i) = H3(i) * g3;
31     H4(i) = H4(i) * g4;
32 end
33 %Werte runden und in unseren Wertebereich verschieben
34 H1n = round(H1 .* 2^15);
35 H2n = round(H2 .* 2^15);
36 H3n = round(H3 .* 2^15);
37 H4n = round(H4 .* 2^15);
```

Matlab-Code zur Optimierung der Koeffizienten

Die Berechnung erfolgte wie bereits für den Tiefpass-Filter. Anschließend mussten die Koeffizienten in den C-Code eingefügt werden.

```
1 // Definition der Filterkoeffizienten
2 #define BIQUAD_STAGES 4 // Anzahl der Koeffizienten
3
4 const short coef[6*BIQUAD_STAGES] = {
5         1,   -122,          0,    122, -16139, 30276, //H4
6         1,   -651,      0,        651, -16116, 29781, //H3
7         1,   -343,      0,        343, -15783, 29817, //H2
8         1,   -451,          0,    451, -15760, 29601  //H1
9 };
```

Codeausschnitt der modifizierten process_data.c

Zur Überprüfung der Implementierung wurde der Amplitudengang und die Sprungantwort aufgenommen. Diese wurden dann mit den von Matlab generierten Idealen Vorgaben verglichen.

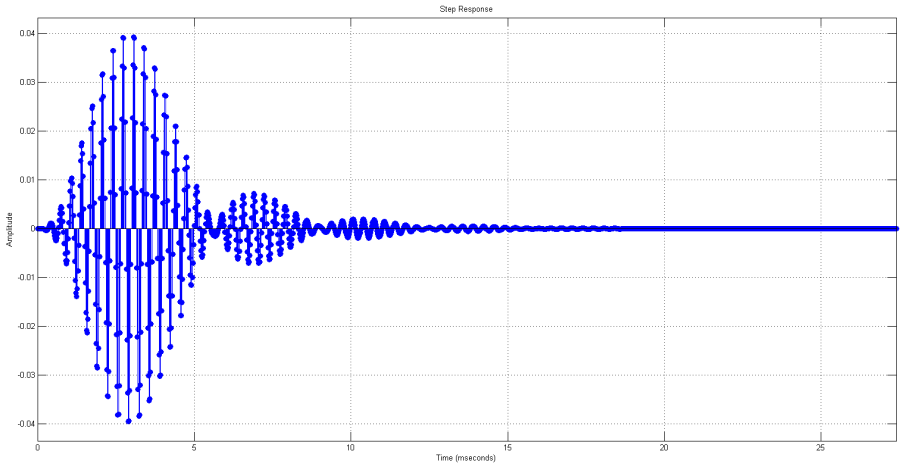

Abbildung 2.3: Ideale Sprungantwort des Filters

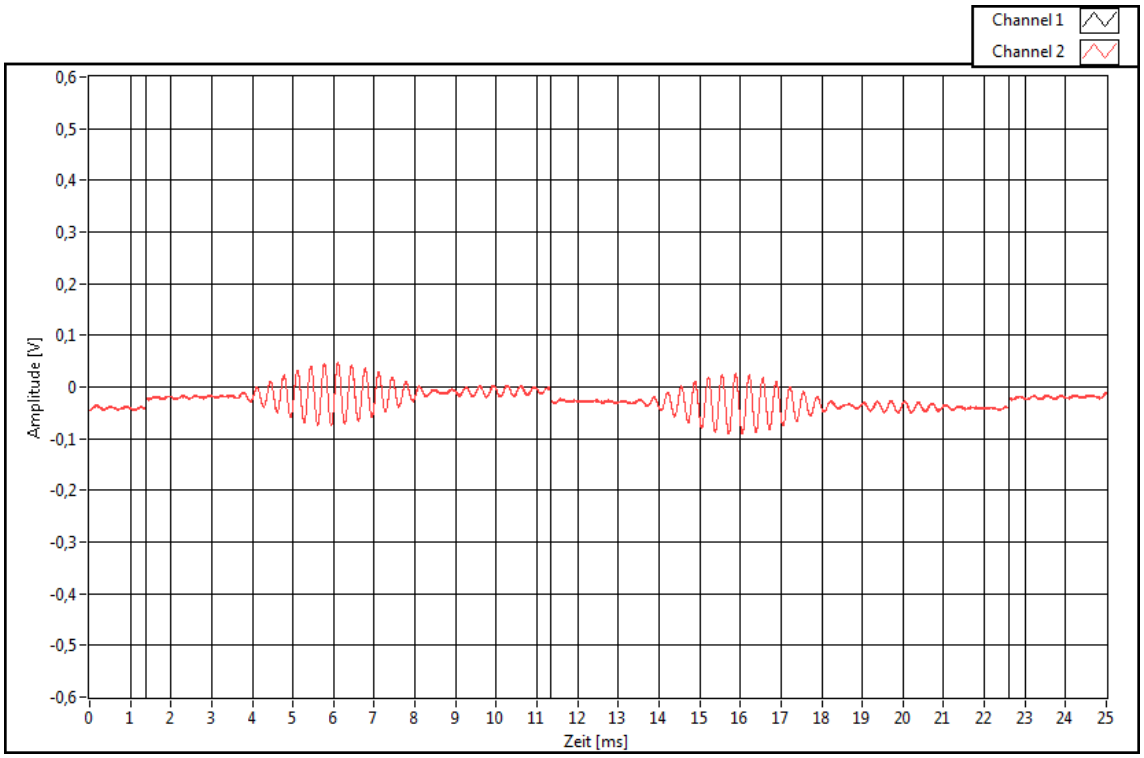

Abbildung 2.4: Reale Sprungantwort des Filters

In Abbildung 2.3 ist die Ideale Sprungantwort und in Abbildung 2.4 die Reale Sprungantwort des Filters zu sehen. Beide Sprungantworten gleichen sich weites gehend. Die Amplituden der beiden Sprungantworten unterscheiden sich leicht, denn der reale Sprung ist bei uns ein Rechteck, welches eine Amplitude von $V_{pp} = 2V$ hat wohingegen der ideale Sprung nur eine Amplitude von $V_{pp} = 1V$ besitzt. Dies erklärt die leicht kleinere Amplitude der idealen Sprungantwort.
Ein weiterer Unterschied, ist die leichte Dämpfung auf dem Gleichspannungsanteil. Dies erklärt sich durch die Dämpfung des Systems.

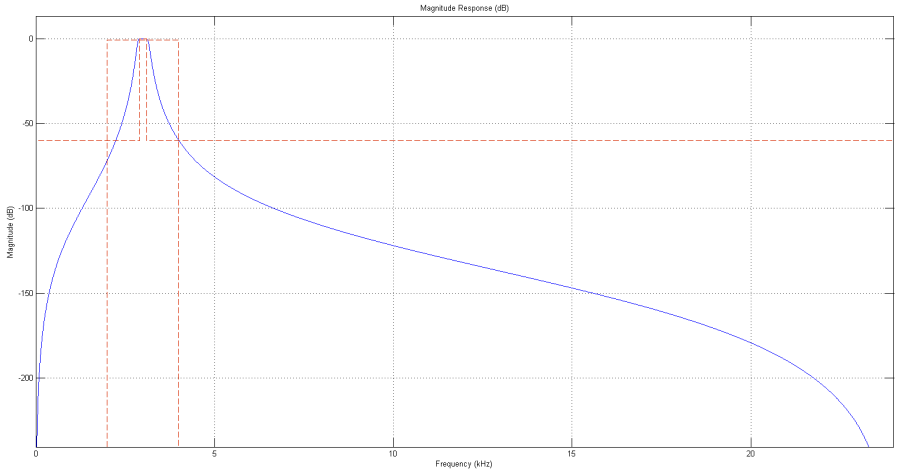

Abbildung 2.5: Idealer Amplitudengang des Filters

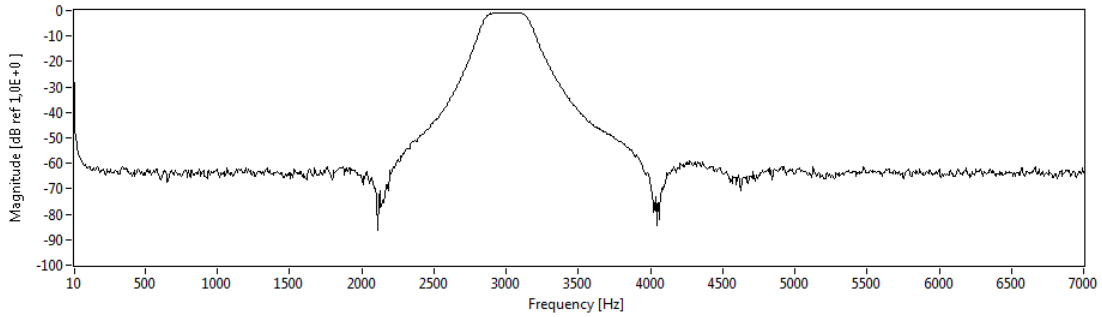

Abbildung 2.6: Realer Amplitudengang des Filters

Abbildung 2.5 ist der ideale Amplitudengang und in Abbildung 2.6 der reale Amplitudengang zu sehen. Beide Amplitudengänge sind sich ähnlich, weisen allerdings im Detail gravierende Unterschiede auf. So erreicht der Reale Filter kein 0dB Dämpfung an der Passfrequenz sondern hat eine Dämpfung von 1dB, dies ist die Dämpfung des Systems. Außerdem wird in den Stoppbändern keine Dauerhafte Dämpfung von unter 60dB erreicht. Dies liegt ebenfalls am System, da dieses 10 Bit nutzt und dabei 0 dB bis -60 dB bei einer Auflösung von 1 dB darstellt. Allerdings erfüllt dies trotzdem die Anforderung.

Negative Peaks erklrären

Anhang A

Quelltext-Dateien

Abbildungsverzeichnis

Abkürzungsverzeichnis

FDA Filter Design and Analysis. 10