Technische Universität Berlin
Fakulät IV
Institut für Energie und Automatisierungstechnik
Fachgebiet Elektronische Mess- und Diagnosetechnik

Praktikum Messdatenverarbeitung
Betreuer: José-Luis Bote-Garcia
SS2018

Versuchsprotokoll 2

Anti-Aliasing Filter und Aufbau einer Messkette

Serdar Gareayaghi (374183)
Ongun Türkcüoglu (371690)
Onur Akdemir (375959)
Marjan Chowdhury (344675)

31. Mai 2018

Inhaltsverzeichnis

1 Einleitung

In diesem Labor werden die Messdaten von dem Sende-Puffer des Mikrocontrollers aufgenommen und laut Praktikumsaufgaben in Matlab bearbeitet. Für die vollständige Aufbau der Messkette soll ein passender Antialiasing-Filter entworfen, um die Aliasing-Fehlern zu vermeiden. Die Eigenschaften des Antialiasing-Filters außer die Abtastfrequenz sind schon gegeben. Für die richtige Einstellung des Antialiasing-Filters muss die passende Abtastfrequenz berechnet. Somit wird der passende Antialiasing-Filter für die Messkette in Matlab entworfen.

2 Praktikumsaufgaben

2.1 Testen des ADUs mit einer Gleichspannung

Es wird zwei unterschiedliche Gleichspannungen zwischen -7 und $+7$ am Gleichspannungsquelle eingestellt und auf den Eingang der Datenerfassungsplatine angelegt. 1000 Messwerte werden mit Hilfe der Funktion ucAnalogRead() aufgenommen und geplottet. Entsprechend der Gleichspannung sind die aufgenommene Messdaten konstant und zeigen damit zu einem Binär-Code. Die blaue Linie in der Abbildung 1 gehört zu der Gleichspannung von $+5$ Volt und die orange Linie zum -3 Volt. Für die Spannungsgrenzen -7 und 7 sind die umgewandelte Werten im ideal Fall zwischen -512 und 512. Unter Berücksichtigung dieser Grenzen sehen die Messwerte in der Abbildung treffend aus.

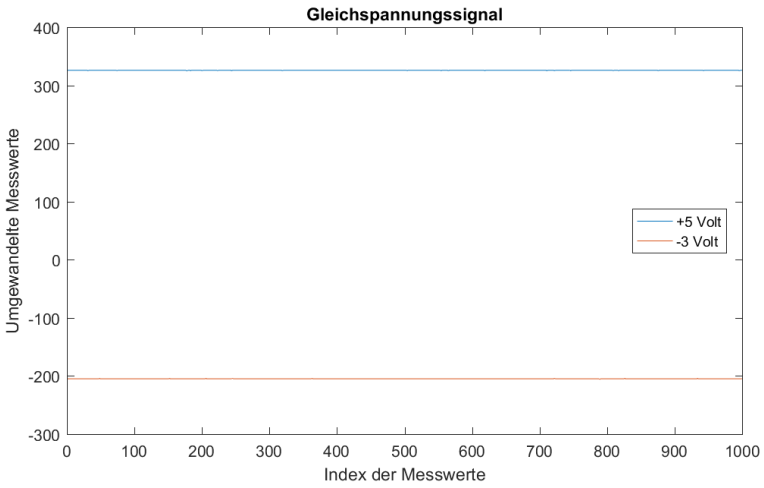

Abbildung 1:

2.2 Aufnahme des Dreiecksignals

Die Aufnahme des Dreiecksignals läuft analog zu der Aufnahme der Gleichspannung in der Aufgabe 2.1. Dabei wird am Funktionsgenerator die peak-to-peak-Amplitude von 14 Volt eingestellt, welche eine Spannung im Bereich von 7V bis -7V versorgt. Die analoge Spannung wird dann mit dem Datenerfassungsschnitt im Mikrocontroller bzw. mit der Funktion ucAnalogRead() aufgenommen, umgewandelt und geplottet. Die Abbildung 2 zeigt das Plot.

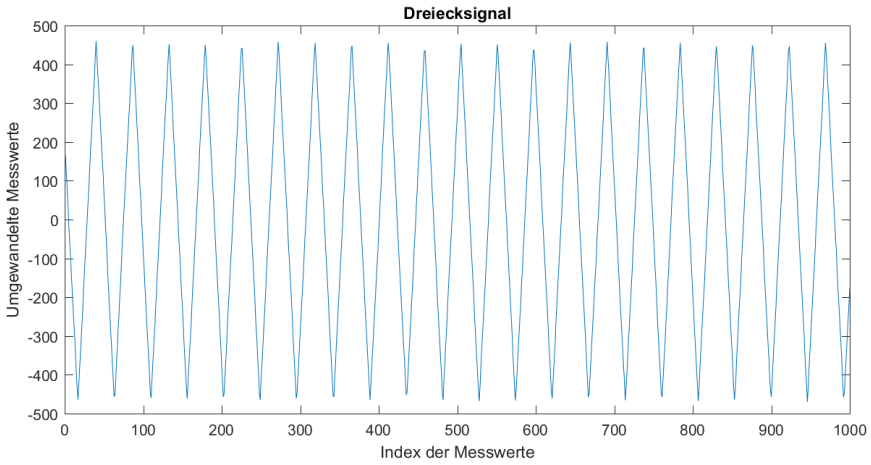

Abbildung 2:

Dabei ist zu beachten, dass obwohl mit dem analogen Signal den ganzen Wertebereich abgedeckt ist, der maximale bzw. minimale Binär-Code entspricht nicht dem idealen Wert von 512 bzw. -512. Dies ist als einen Messfehler zu interpretieren, und die Tatsache wird in den nächsten Abschnitten noch relevanter.

2.3 Kennlinie des ADUs und Rücktransformation der Messwerte

Eine steigende Flanke des Dreiecksignals wird mit der MATLAB-Funktion herausgeschnitten und die Zeitachse wird in eine Spannungsachse transformiert, und damit steht an der y-Achse die ADU-Code und an der x-Achse die Eingangsspannung. Aufgrund der rekursiven Methode der MATLAB-Funktion ist es wichtig, das analog Signal im Niederfrequenzbereich einzustellen, damit es nicht zu einem Fehler kommt. Der MATLAB-Code für die steigende Flanke ist im Listing 1 zu sehen, bzw. es wird hochgeladen.

```
1 % steigende Flanke
2 [min_aduwerte2, index_min_aduwerte2] = min(aduwerte_a2);
3 i = 0;
4 while(aduwerte_a2(index_min_aduwerte2 + i) <
     aduwerte_a2(index_min_aduwerte2 + i + 1))
5      i = i + 1;
6 end
7 clf
8 flanke = aduwerte_a2(index_min_aduwerte2: index_min_aduwerte2 +
     i);
```

Listing 1: MATLAB-Code für Isolation der steigenden Flanke aus pr2.m

```
1 function voltage = Code2Volt(code, adu_steigung, adu_offset)
2
3 voltage = (code - adu_offset)/adu_steigung;
4
5 end
```

Listing 2: Code für die Rücktransformation der Messdaten in den Spannungsbereich

Die Steigung und Offset der Kennlinie sind durch eine lineare Regression bestimmt. Die Steigung ist dabei 65.5549 und Offset ist -9.8077. Die MATLAB-Funktion Code2Volt.m (siehe 2) dient dazu, dass die binäre Codes mit der Steigung und dem Offset des ADUs in den Spannungsbereich rücktransformiert werden können. Die reale bzw. die ideal Kennlinie eines 10-Bit ADUs ist in der Abbildung 3 zu sehen.

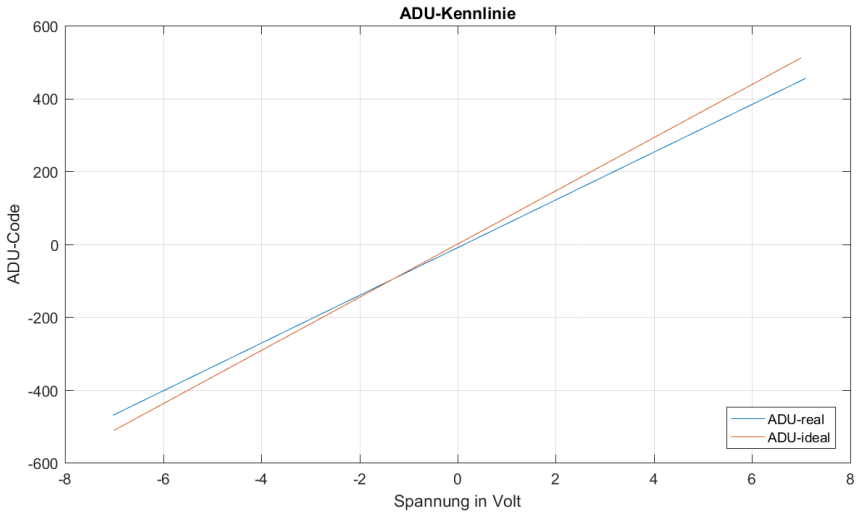

Abbildung 3: die reale bzw. die ideale Kennlinie des ADUs

Für die Rückwandlung der Messdaten mit der Funktion Code2Volt, müssen die Steigung das Offset der Adu in der Funktion eingesetzt zusammen mit der Code.

2.4 Amplitudenfrequenzgang

Für Amplitudenganganalyse des Antialiasing-Filters wird am Funktionsgenerator ein Sinussignal mit der Amplitude 14V eingestellt. Um die Dämpfung unterschiedlichler Frequenzanteile zu untersuchen werden manuel unterschiedliche Frequenzen eingestellt und mit dem Mikrocontroller aufgenommen. Die Frequenzen sind folgendermaßen ausgewählt: 10, 100, 500, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000 Hz. Bei diesen Frequenzen werden die peak-to-peak Amplituden aufgenommen und in die Formel 1 für die Berechnung der Amplitudengang eingesetzt. Der resultuierende Amplitudenfrequenzgang ist in der Abbildung 4 zusammen mit der Kurve des idealen Tiefpassfilters gleicher Ordnung zu sehen.

$$A = 20log\left(\frac{U_{peak-to-peak}}{14}\right) \tag{1}$$

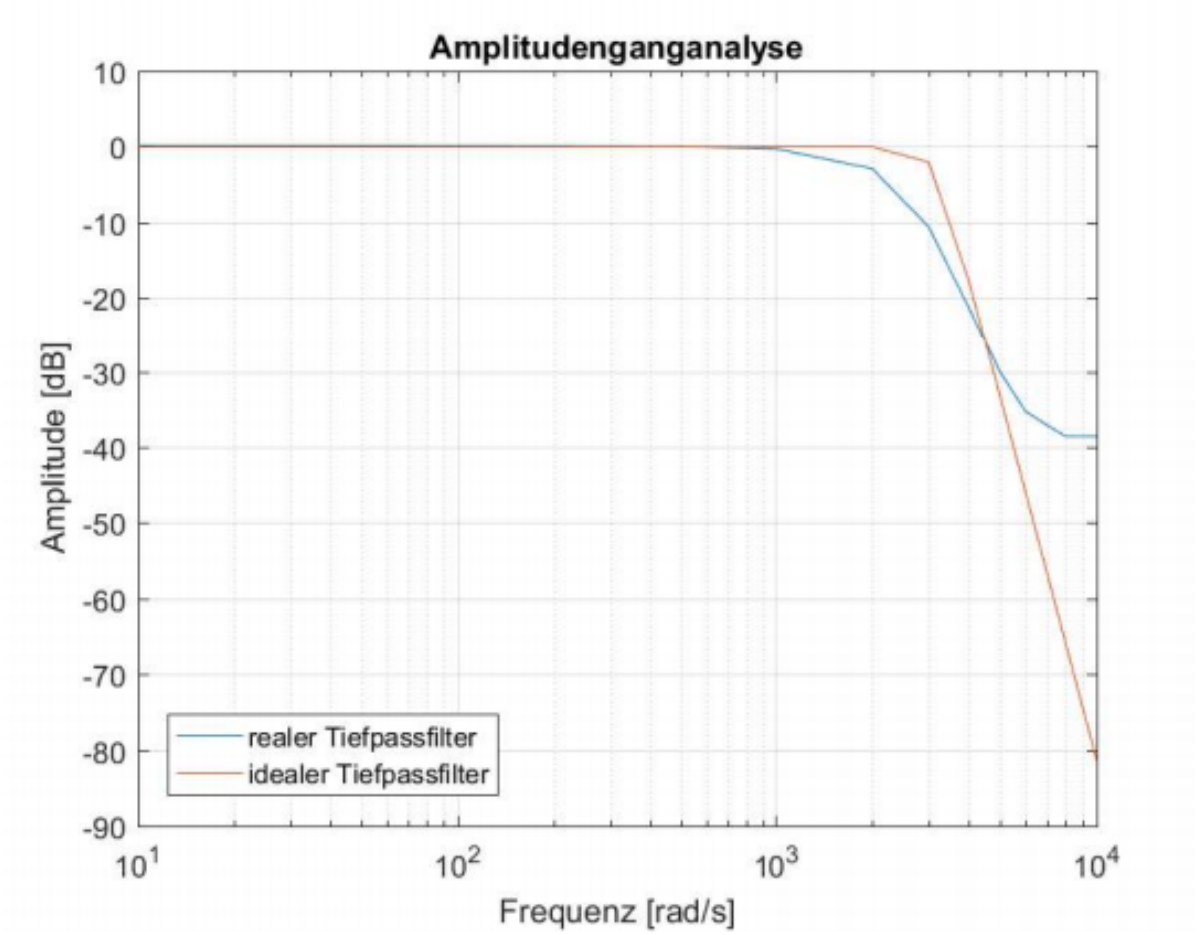

Abbildung 4: Vergleich der Amplitudenganganalyse des realen bzw idealen Butterworth-Tiefpassfilter 8. Ordnung

Der reale Butterworth-Tiefpassfilter hat dabei eine Grenzfrequenz kleiner als 3.1kHz, und die Steilheit der Linie ist nicht exakt 20dB/Dekade. Trotzdem sind die Charakteristika des realen Tiefpassfilters anpassend mit dem idealen Tiefpassfilter.

3 Abgabeaufgaben

3.1 Aufgabe 1

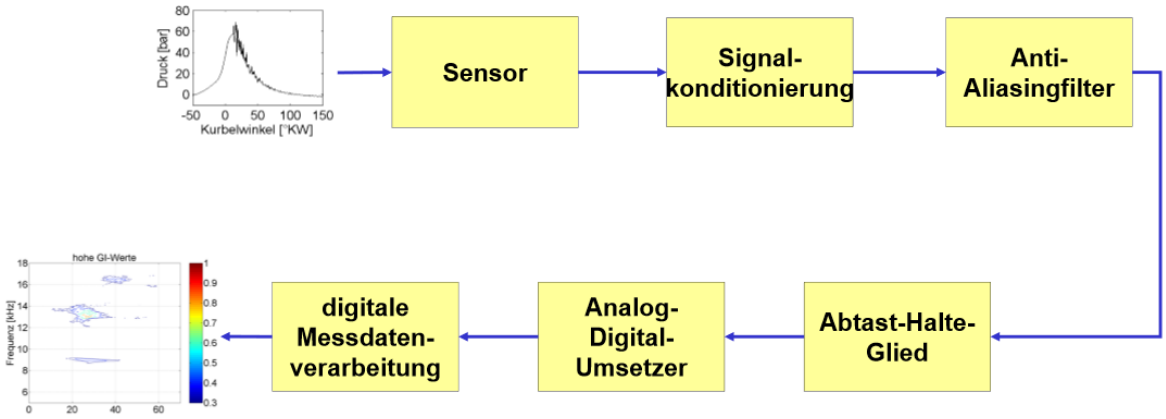

Abbildung 5: Aufbau einer digitalen Messkette [1]

In Abbildung 5 ist der schematische Aufbau einer digitalen Messkette dargestellt. Eine digitale Messkette besteht aus einem Sensor (zur Aufnahme der Messwerte), mehreren Verstärkern, Antialiasing-Filter, Abtast-und-Halteglied, Analog-Digital-Umsetzer und ein PC oder ein Mikrocontroller (zur Auswertung und Verarbeitung der Messwerte). Mithilfe einer Messkette können kontinuierliche (analoge) Signale in diskrete (digitale) Signale umgewandelt werden.

Der wichtigste Baustein in einem digitalen Messskette ist der Analog-Digital-Umsetzer. Ein ADU ist eine elektrische Schaltung die wertekontinuierliche (analoge) Signale bzw. Eingangsgrößen in eine Binärzahl (diskrete Signale) umwandelt. Also der analogen Messgröße wird zu diskreten Zeitpunkten geteilt. Der zeitlicher Abstand dieser Zeitpunkte ist die Periode T. Die Frequenz $f_a = 1/T$ wird als Abtastfrequenz bezeichnet.

Um das Signal ohne Verlust wieder zu erstellen, muss der Abtastfrequenz mindestens doppelt so groß wie die Frequenz unser Signals sein $f_a > 2f$ (Nyquist-Shannon Theorem). Wenn diese Bedingung verletzt wird, dann entstehen Spiegelungsfehler (Aliasing). Um Aliasing zu vermeiden verwendet man in der Regel einen Tiefpassfilter.

Danach wird das gefilterte Signal mittels Abtast-und-Halteglied zu äquidistanten Zeitpunkten abgetastet. Abtast-und-Halteglied ist der erste Schritt von einem ADU. Durch die Abtastung entsteht ein zeitdiskretes Signal, jedoch ist die Amplitude immer noch kontinuer-

lich. Hold-Glied haltet die Amplitude auf konstante Werte bis der nächster Wert ankommt. Letztendlich kann das Signal von einem Computer verarbeitet werden.

3.2 Aufgabe 2

Bei der zweiten Aufgabe muss eine geeignete Abtastfrequenz f_0 eingestellt werden, damit Aliasing-Effekte nicht mehr auftreten. Das heißt, die Frequenzanteile $f > \frac{f_0}{2}$ so gedämpft werden, dass ihre Amplitude kleiner als die analoge Auflösung des ADUs sind. Um die Eigenschaft analytisch zu ermitteln, sind folgende Zusammenhänge zu benutzen:

$$|A_F(\omega_0/2)|_{db} \geq 20 \cdot \log_{10}\left(\frac{U_{max} - U_{min}}{U_{LSB}}\right) = 20 \cdot \log_{10}(2^N - 1)dB \approx 6.02N \cdot dB \quad (2)$$

$$m = \frac{\Delta y}{\Delta x} = \frac{|A_F(\omega_0/2)|_{db}}{\log_{10}\left(\frac{w_0/2}{w_g}\right) - \log_{10}(1)} \quad (3)$$

$$\omega = \frac{2\pi}{T} = 2\pi f \quad (4)$$

Dabei ergibt die Gleichung 2 die erforderliche Dämpfung an der halben Abtastfrequenz, die Gleichung 3 für die Ordnung des Filters von der minimal erforderlichen Dämpfung und die Gleichung 4 den Zusammenhang zwischen Kreisfrequenz und Frequenz.

Mit den oben genannten Gleichungen und die Auflösung des ADUs (N = 10 Bits) kann die Aufgabe 2 (bzw. Aufgabe 3) gelöst werden. Also:

Die minimal erförderliche Dämpfung ergibt sich aus der Gleichung 2:

$$|A_F(\omega_0/2)|_{db} = 6.02N \cdot dB = 60.2dB$$

Einsetzen in die Gleichung 3, mit m = 40 dB (Einen Tiefpassfilter 2. Ordnung ergibt dann 40 dB, denn es gilt 20 dB pro Ordnung Steilheit):

$$40dB = \frac{60.2dB}{\log_{10}\left(\frac{w_0/2}{w_g}\right) - \log_{10}(1)}$$

$$\log_{10}\left(\frac{w_0/2}{w_g}\right) = \frac{60.2}{40} \implies \frac{w_0/2}{w_g} = 10^{1.5050}$$

Mit dem Zusammenhang (4) ergibt sich dann:

$$f_0 = 2 \cdot 10^{1.5050} \cdot f_g = 198330Hz$$

Mit der Abtastfrequenz von 198330 Hz sind die Aliasing-Effekte gedämpft. Die Frequenzganganalyse ist in der Abbildung 6 zu sehen:

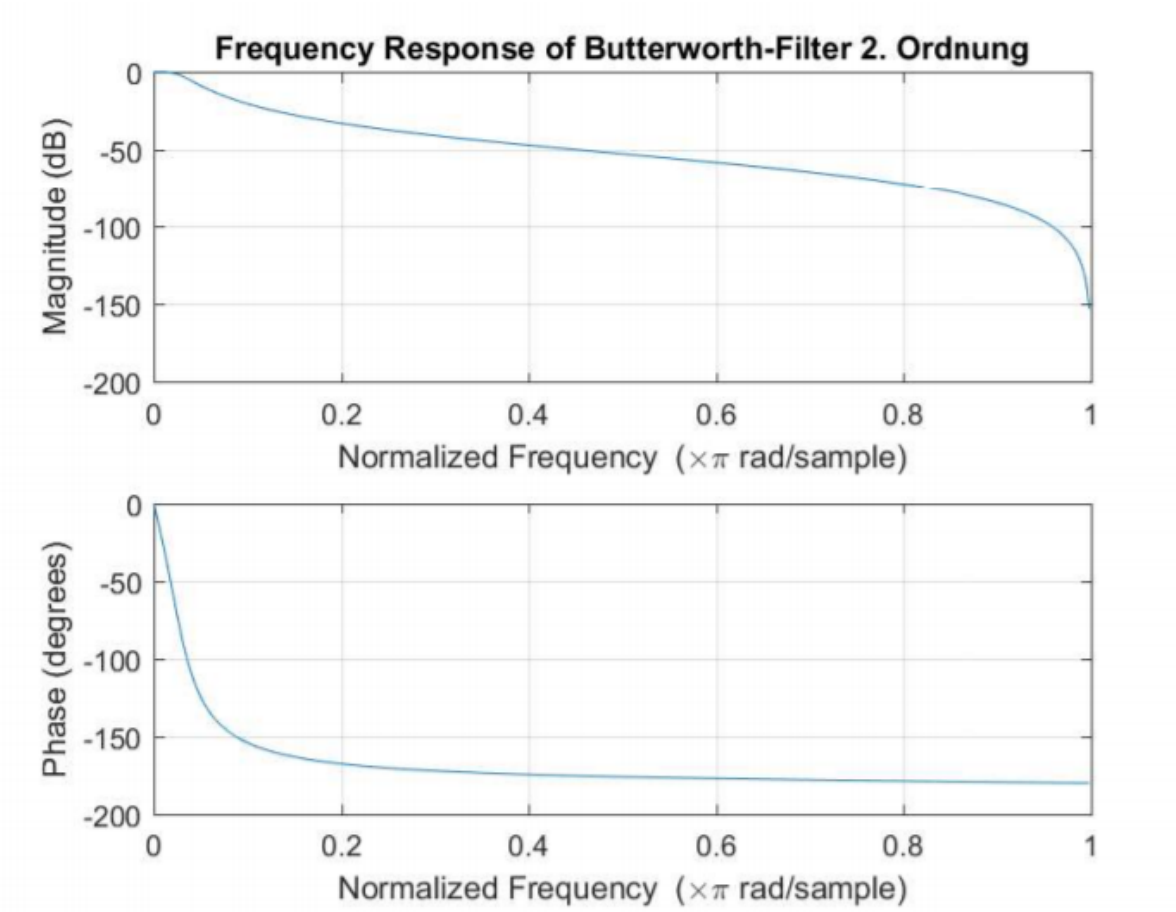

Abbildung 6: Frequenzganganalyse eines Butterworth-Filters 2. Ordnung

Die Frequenzanteile größer als die halbe Abtastkreisfrequenz (also größer als 0.5π rad/sample) sind mit der minimal erforderlichen Dämpfung gedämpft.

3.3 Aufgabe 3

Die Aufgabe 3 ist analog zu der Aufgabe 2, wobei der einzige Unterschied die Ordnung des Filters ist (Statt 2. Ordnung, 8. Ordnung). Wegen dem identischen Analog-Digital-Umsetzer bleiben wir bei der erforderlichen Dämpfung von 60.2 dB. Also:

Die minimal erförderliche Dämpfung ergibt sich aus der Gleichung 2:

$$|A_F(\omega_0/2)|_{db} = 6.02N \cdot dB = 60.2dB$$

Einsetzen in die Gleichung 3, mit m = 160 dB (Einen Tiefpassfilter 2. Ordnung ergibt dann 160 dB, denn es gilt 20 dB pro Ordnung Steilheit):

$$160dB = \frac{60.2dB}{\log_{10}\left(\frac{w_0/2}{w_g}\right) - \log_{10}(1)}$$

$$\log_{10}\left(\frac{w_0/2}{w_g}\right) = \frac{60.2}{160} \implies \frac{w_0/2}{w_g} = 10^{0.3763}$$

Mit dem Zusammenhang (4) ergibt sich dann:

$$f_0 = 2 \cdot 10^{0.3763} \cdot f_g = 14747Hz$$

Die Frequenzganganalyse des Butterworth-Filters 8. Ordnung ist in der Abbildung 7 zu sehen.

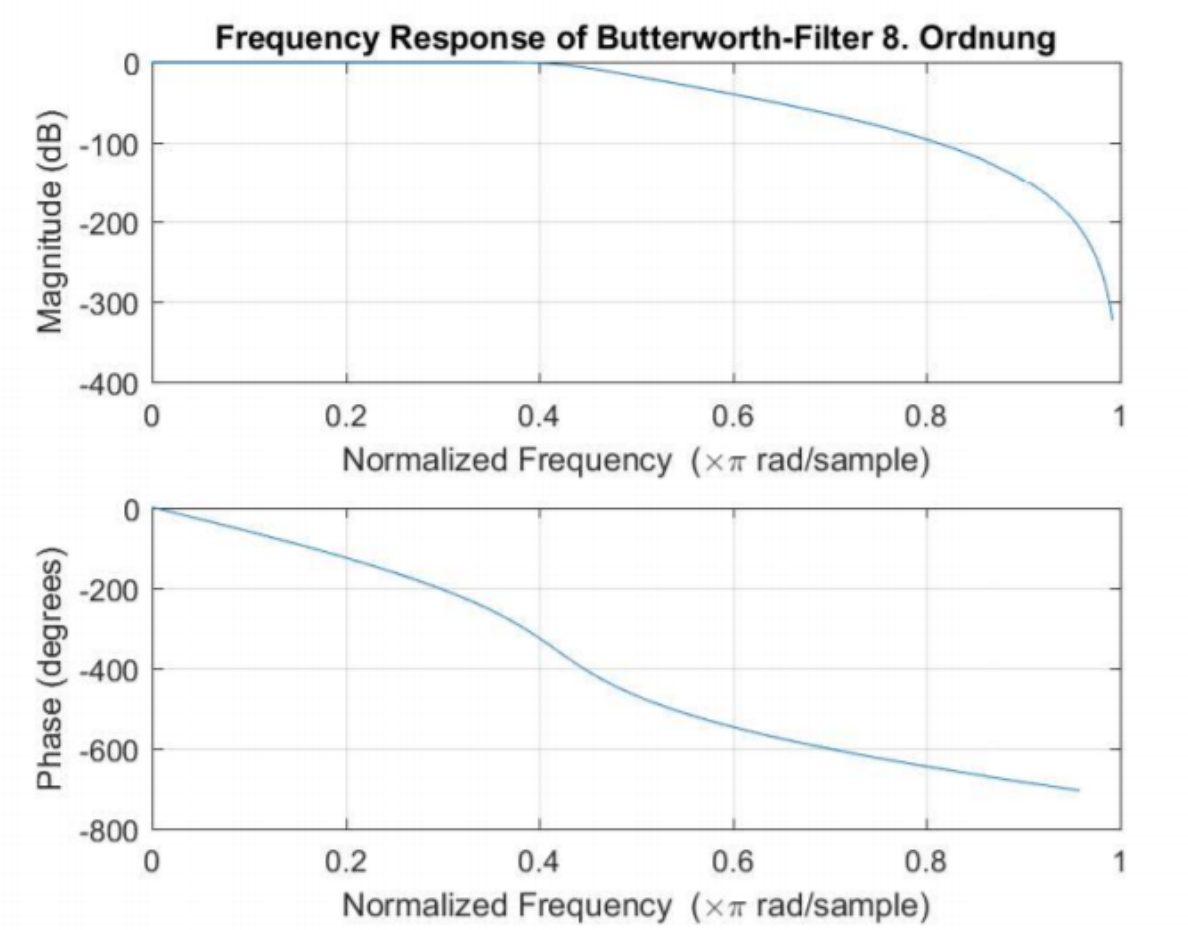

Abbildung 7: Frequenzganganalyse eines Butterworth-Filters 8. Ordnung

3.4 Aufgabe 4

Unter Berücksichtigung auf eine Abtastfrequenz von $14747Hz$ für der 8. Ordnung Tiefpass-Filter, sollte eine Spektralanalyse bis maximal $f_m = 2000Hz$ ohne Aliasing-Fehlern vollständig funktionieren.

3.5 Aufgabe 5

Das Differenzsignal ergibt sich aus der Differenz zwischen dem Signal (also von der steigender Flanke) und der Regressionsgeraden. Das Signal enthält dabei die eigentliche Werten und eine Realiserung vom Gausschen Rauschen, wobei die lineare Regressionslinie die Fehlerquadrate minimiert. Deswegen minimiert die Regressionslinie das Messrauschen, welches man als Signal ohne Messrauschen interpretieren kann.

Notiz von Autoren: *Die steigende Flanke, die wir aufgenommen haben, hat nur 25 Messwerten. Deswegen ist die Messrauschenrealisierung nicht zufällig genug, bzw. nicht stark genug, dass die SNR bzw. Effektive Anzahl von Bits kein sinnvolles Ergebnis liefern. Trotzdem haben wir SNR und ENOB berechnet, mit dem vollen Wissen, dass es keinen Sinn ergibt.*

Signal-Noise-Ratio (SNR, deut. Signal-Rausch-Verhältnis) ergibt sich aus dem Verhältnis der Leistungen des Signals und des Rauschens. Das Signal enthält dabei die eigentliche Werten und eine zufällige Realisierung des Messrauschens, wobei die lineare Regression eine Linie liefert, die die Fehlerquadrate minimiert und dabei das Messrauschen quasi-ausgleicht. Für die Berechnung des SNR wird die folgende Formel verwendet:

$$\mathrm{SNR}_{dB} = 10 \log_{10} \left(\frac{P_{signal}}{P_{noise}} \right) \tag{5}$$

Die Listing 3 zeigt den Code für die Berechnung des SNR und ENOB.

```
% Differenz zwischen Flanke und Regressionsgraden bilden
% digitale Binaercodes in Spannungsbereich transformieren
c2v_p2 = Code2Volt(p3,65.5549,-9.8077);
c2v_flanke = Code2Volt(flanke,65.5549,-9.8077);
differenz = c2v_p2 - c2v_flanke;
% SNR signal(flanke)-noise(differenz)-ratio
signal_to_noise_ratio = snr(flanke,differenz);
% Effektive Anzahl von Bits
enob = (signal_to_noise_ratio - 1.76)/6.02;
% Plot Differenzsignal
```

```
11 plot(differenz)
12 grid on;
13 title('Differenzsignal');
14 ylabel('Spannung [V]');
15 xlabel('Index');
16 ylim([3*-10^-16,10^-16]);
```

Listing 3: MATLAB-Code für SNR und ENOB

Dabei ergibt sich für das maximale SNR 3.602711797285753e+02 (360.271) und für die effektive Anzahl von Bits 59.553352114381280 .

Für den Vergleich der Amplitudenganganalyse siehe bitte 2.4.

Literatur

[1] Prof. Dr.-Ing. Clemens Gühmann
Skript zur Messtechnik, TU Berlin, WS 2016