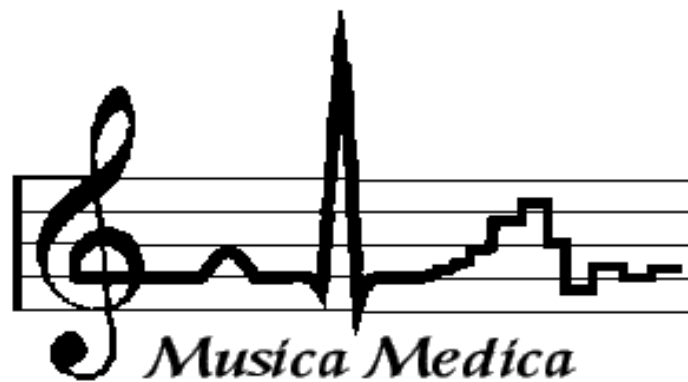


# Ds - Studienarbeit Musiktherapiegerät



Betreuer:  
Auftraggeber:  
Studenten:  
Abgabedatum:

Andreas Ehrensperger  
Dr. med. vet. Schifftan, Rapperswil  
Stefan Heer E96a / Christian Schläpfer E96a  
9. Juli 1999

## 0. 1 Inhaltsverzeichnis

---

<b>Einleitung</b>	<b>3</b>
Zu diesem Bericht.....	3
Ziel der Arbeit / Motivation .....	3
Was ist Musiktherapie .....	3
<b>Pflichtenheft</b> .....	<b>4</b>
Grundsätzliches.....	4
Anforderungen vom Auftraggeber.....	4
Studienarbeit .....	5
<b>Zeitplan</b> .....	<b>5</b>
<b>Verwendete Hilfsmittel und Geräte</b> .....	<b>6</b>
<b>Dokumentation der Arbeit</b>	<b>7</b>
Abstract.....	7
Evaluation DSP Board .....	7
Signalflussbild.....	8
Benutzer-Interface .....	9
Schnittstelle Matlab-Sharc .....	10
Übergabeparameter .....	10
Initialisierung / Synchronisation Matlab-DSP .....	11
Flussdiagramm .....	12
<b>Implementierte Funktionen</b> .....	<b>13</b>
Lautstärkenregelung / Balance.....	13
FIR-Filter .....	13
IIR - Filter / Tiefpass der Vibratoren .....	21
Zeitverzögerungen .....	25
Schwebung.....	25
<b>Bedienung</b> .....	<b>26</b>
<b>Probleme</b> .....	<b>26</b>
Rauschsignal auf DSP Ausgängen.....	27
Defekt der DSP Karte .....	27
Zeitprobleme / nicht Erreichtes.....	27
<b>Schlusswort</b> .....	<b>27</b>
<b>Anhang</b>	<b>29</b>
<b>Software</b> .....	<b>Anhang</b>
Quellcode .....	Anhang
Header File.....	Anhang
Hilfs- und Interfacesoftware .....	Anhang

# I. Einleitung

## 1. 1 Zu diesem Bericht

---

Dieser Bericht entstand während rund 14 Wochen im Sommersemester 99 und dokumentiert die zweite Studienarbeit von Stefan Heer und Christian Schläpfer im Fach digitale Signalverarbeitung.

In der Einleitung wird auf die Thematik der Musiktherapie ganz allgemein eingegangen. Ausserdem werden äussere Zusammenhänge der verwendeten Arbeitsmittel aufgezeigt, um einen Rahmen zur Arbeit zu schaffen. Der Teil ‚Dokumentation der Arbeit‘ beschäftigt sich mit der eigentlichen Lösung der Probleme und der Implementation der Algorithmen auf Matlab und dem DSP. In einem dritten Teil werden die aufgetauchten Probleme erläutert.

## 1. 2 Ziel der Arbeit / Motivation

---

Ein persönliches Ziel dieser Arbeit war das Kennenlernen eines DSP. Uns interessierten die Möglichkeiten, die einem ein DSP eröffnet, aber auch die Grenzen und die Probleme die auftauchen können. Ausserdem reizte es uns, wie schon bei der ersten Studienarbeit, ein Problem aus der Praxis zu behandeln. Eine Motivation war die Arbeit mit Musik als solcher und speziell die Arbeit mit Musik im Zusammenhang mit der Medizin.

Der Auftraggeber war Herr Dr. med. vet. Schifftan. Der Tierarzt aus Rapperswil beschäftigt sich seit langem mit Musiktherapie und gelangte mit der Bitte an die Schule, ein Gerät nach seiner Anforderungsspezifikation zu bauen.

## 1. 3 Was ist Musiktherapie?

---

Musik wird von den Menschen seit Urzeiten nicht nur als Unterhaltungs- und Kunstmedium verwendet, sondern auch für deren beruhigende und harmonisierende Wirkung geschätzt. Die Musiktherapie arbeitet mit diesen Effekten. Unter Musiktherapie versteht man ganz allgemein die medizinische Nutzung der Auswirkungen von Musik auf Körper und Geist. Es ist wissenschaftlich erwiesen, dass die Beeinflussung des Menschen durch Sinneswahrnehmungen vervielfacht wird, wenn diese Wahrnehmungen über mehrere Kanäle auf den Menschen wirken. Aus diesem Grund wird in der Musiktherapie nicht nur Musik gehört, sondern auch gespürt und zwar über kleine Vibratoren die vorzugsweise auf den Brustkasten (Resonanzkörper!) gelegt werden. Vor allem tiefe Frequenzen (Bässe) sind so deutlich zu spüren. Es wird ein gewisser ‚Live‘- Eindruck bemerkt und die Musik wird intensiver ‚gehört‘. Selbstversuche unsererseits haben bestätigt, dass sich mit geeigneter Musik sehr rasch ein Gefühl der Entspannung breitmacht.

Angewendet werden kann diese Art ‚Meditation‘ eigentlich überall, da Nebenwirkungen verständlicherweise nicht vorhanden sind (Ausgenommen sind Ohrenschäden...). Bis anhin hat unser Auftraggeber, Herr Dr. Schiftan, mit einem Gerät gearbeitet, welches sehr beschränkte Möglichkeiten bietet. Dabei hat er auf verschiedensten Gebieten gute Erfahrungen gemacht. So konnte z. B. Vielfliegern (Piloten, Stewardessen, etc.) mit Jet-Lag-Problemen geholfen werden, indem ihr Organismus durch die Musik ‚reharmonisiert‘ wurde. Gebärenden Frauen dient die Therapie als Schmerzmittel da sie Verkrampfungen durch Entspannung lösen kann. Herr Schiftans Idee ist es, durch die vom DSP ermöglichten Spezialeffekte die Anwendungsgebiete der Musiktherapie auszudehnen.

## **1. 4 Pflichtenheft**

---

### **1. 4. 1 Grundsätzliches**

Das Musiktherapie-Gerät soll eine Stereo-Audioquelle in zwei unabhängige Stereokanäle, also vier physikalische Kanäle auftrennen. Für jeden Kanal sind diverse Parameter einzeln verstellbar. Zwei Kanäle führen auf einen Kopfhörer, die anderen beiden auf zwei Vibratoren, welche die Musik in tieferen Frequenzen fühlbar machen.

### **1. 4. 2 Anforderungen vom Auftraggeber**

#### **DSP**

- Vier autonome Audioausgänge, zwei auf ein Kopfhörerpaar, zwei auf ein Vibratorenpaar.
- Jeder der 4 Kanäle soll über einen separaten Equalizer geführt sein.
- Der rechte Kanal (1x Kopfhörer, 1x Vibrator) soll gegenüber dem linken Kanal dynamisch zeitlich verschoben werden können (Schwebung). Ebenfalls soll das Kopfhörersignal gegenüber dem Vibratorensignal zeitlich verschoben werden können.
- Dem Signal soll Rauschen beigemischt werden können.(Normalverteilt)
- Dem Signal soll ein Beat unterlegt werden können. Unter Beat wird hier eine rythmische Wiederholung eines wählbaren Geräusches verstanden.
- Die Lautstärke/Balance kann für jeden Kanal einzeln geregelt werden.

#### **Hardwareanforderungen (nicht den DSP betreffend)**

- Autonomes, computerunabhängiges Gerät
- Mikrofon-Eingang, der direkt auf Kopfhörerausgänge wirkt
- Zusätzlicher Output (Identisch Kopfhöreroutput) um eine Lichtquelle anzusteuern
- Output-Verdoppelung, damit 2 Patienten gleichzeitig bedient werden können
- Akku-Betrieb
- Display zur Parametereinstellung und Funktionsüberwachung
- Kein Kabelsalat

### Nebenbedingungen

- Das Ziel wäre, das Gerät als Kleinserie von ca. 10 Stück zu realisieren

### 1. 4. 3 Studienarbeit

Da im Rahmen einer Studienarbeit die Zeit nicht ausreicht, um all diesen Anforderungen gerecht zu werden, änderten wir die Aufgabenstellung in folgenden Punkten ab:

- Realisierung der unter Punkt 1. DSP aufgeführten Anforderungen, allerdings nicht auf einem autonomen Board, sondern auf dem rechnerinternen Sharc-Board
- Simulation des Endgerätes mit Matlab, Echtzeitverarbeitung auf dem Sharc-Board. Dadurch soll die Türe offengehalten werden, eine spätere, autonome Realisierung des Projektes zu ermöglichen.

### 1. 5 Zeitplan

Woche	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Definition der Studienarbeit														
Evaluation des DSPs														
Versuche auf Matlab														
Einarbeitung Matlaboberflächen und Sharc														
Entwickeln und programmieren der Features auf Sharc														
Programmieren des GUI														
Dokumentation erstellen														

## 1. 6 Verwendete Hilfsmittel und Geräte

---

Kernstück der Arbeit bildet der Digitale Sound Processor ADSP 21060 von Sharc, montiert auf einem Blacktip PCI-Board. Damit 4-Kanäle bearbeitet werden können, wurde darauf eine bitsiBB2 Karte aufgesetzt. Alle signalbezogenen Rechenalgorithmen werden auf diesen zwei Karten durchgeführt.

Die graphische Benutzeroberfläche wurde auf Matlab von TheMathWorks Inc. (Version 5.2.1) aufgebaut.

Im weiteren dienten uns folgende im DS-Labor zur Verfügung stehende Geräte:

- HP KO 54600A
- HAMEG Sine-Wave Generator HM 8032
- PC-Aktivlautsprecher
- Kopfhörer
- Mikrofon
- CD-Player
- Fluke 85 Multimeter

## II. Dokumentation der Arbeit

### 2.1 Abstract

---

Dieses Dokument beschreibt die Realisierung eines Prototypen des Musiktherapiegerätes ‚MusicaMedica‘, welches wir im Auftrag von Herr Dr. Schiftan, Rapperswil erstellt haben.

Zweck des Gerätes ist die Veränderung von Musiksignalen zu therapeutischen Zwecken. Der Stereoausgang eines CD-Players wird aufgeteilt in zwei separate Stereokanäle. An einen Kanal wird ein Kopfhörerpaar angeschlossen, an den anderen zwei Vibratoren welche dem Patienten z. B. auf das Brustbein (Resonanzkörper!) gelgegt werden.

Beide Stereokanäle werden über ein Filter geführt, der Kopfhörerkanal über ein FIR-Filter mit einstellbarer Filterkurve, der Vibratorenkanal über ein Tiefpass-IIR-Filter mit einstellbarer Grenzfrequenz. Diese insgesamt vier Kanäle sind individuell zeitlich gegeneinander verzögerbar. Ausserdem wurde im Kopfhörer-Stereokanal eine Schwebung zwischen dem linken und dem rechten Kanal simuliert. Die Lautstärke und die Balance beider Stereokanäle ist verstellbar. Der Prototyp wird mit Hilfe eines auf Matlab programmierten Benutzer-Interfaces bedient. Alle Berechnungen, auch jene der Filterkoeffizienten sind auf dem DSP 21060 Sharc implementiert.

### 2.2 Evaluation DSP Board

---

Zur Auswahl standen uns von der Schule her zwei verschiedene Boards. Ein Motorola 56002 und der ADSP 21060 Sharc. Da unser Betreuer, Herr Ehrensperger, sich mit dem Motorola-Board bestens auskennt, lag es nahe, dieses genauer unter die Lupe zu nehmen. Nach einigem Probieren und Üben mit Tutorials mussten wir aber feststellen, dass sich nur sehr schwer und mit grossem Hardware-Aufwand aus dem ursprünglich zweikanaligen Board ein Vierkanaliges machen liesse.

Als sich dann herausstellte, dass für den Sharc ein vierkanaliges Zusatzboard in der Schule vorhanden war, war der Entscheid schnell gefällt. Dass das bitsi-BB2 Vierkanalboard nicht ganz ohne Probleme funktioniert, stellte sich leider erst später heraus. Ein weiterer Punkt, der für den Sharc sprach, war dessen C-Compiler. Auf dem 56002 von Motorola wäre die Programmierung in Assembler zu bewältigen gewesen. Als DSP-Einsteiger und Assembler-Anfänger war uns jedoch die C-Programmierung zum Anfang symphathischer. Zudem hatten wir für den Sharc einen kompetenten Berater in Person des DS-Labor-Assistenten André Rüegg gerade am Arbeitsplatz.

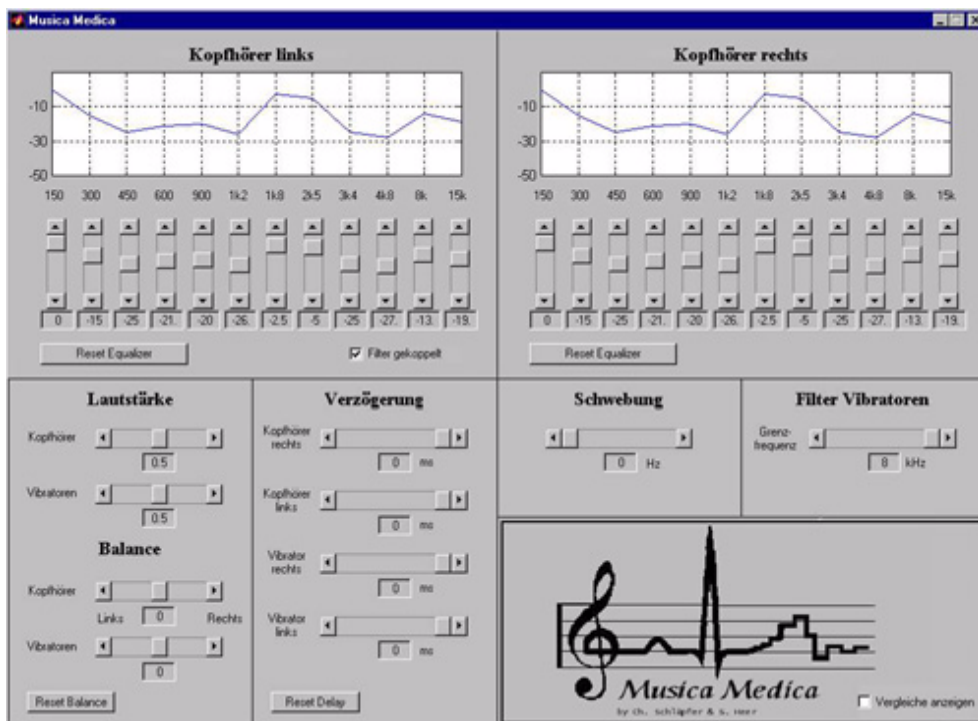


## 2. 4 Benutzer-Interface

Das Benutzer Interface wurde auf Matlab programmiert. Matlab bietet dazu eine Palette von Bedienelementen an, welche nach dem objektorientierten Ansatz erzeugt und einfach verwendet werden können. Bei der stolzen Anzahl Kontrollelemente auf unserer Oberfläche wird der zeilenmässige Codeaufwand jedoch beträchtlich.

In Anbetracht darauf, dass das Gerät unter Umständen in einer weiterführenden Arbeit über ein Display gesteuert werden soll und somit weniger Bediennmöglichkeiten zur Verfügung stehen, hielten wir uns mit der Darstellung an einen möglichst einfachen Rahmen. Wir stellten uns die Aufgabe, das einfachst mögliche Interface zu erstellen, welches aber trotz allem noch selbsterklärend bedient werden kann.

Aus demselben Grund führten wir in Matlab keine Berechnungen von Filterkoeffizienten oder Ähnlichem durch. Alles was gerechnet wird, geschieht auf dem DSP.



Auf dem obigen Bild ist ein Screenshot der Oberfläche zu sehen.

In der oberen Hälfte befinden sich die Equalizer des Kopfhörerkanals. Dort kann der gewünschte Frequenzgang vorgegeben werden. Die Kurve ist in einer nichtlinearen Skalierung aufgezzeichnet. Nach jeder Bewegung eines beliebigen Schiebers tasten wir in Matlab die Filterkurve in konstanten Abständen ab. Diese Parameter werden auf den DSP geladen. Dort werden daraus die Filterkoeffizienten berechnet und auf das Signal angewendet. Dasselbe geschieht mit den Koeffizienten die in der unteren Hälfte manipulierbar sind. Darunter sind neben den Lautstärken- und Balanceschiebern die ‚Spezialfeatures‘ wie Verzögerung, Schwebung und Vibratorenfilterung.

## 2. 5 Schnittstelle Matlab-Sharc

### 2. 5. 1 Übergabeparameter

Die Aufgaben zwischen Matlab und DSP sind wie folgt verteilt: Matlab dient einzig als GUI. Alle Filterkoeffizienten werden auf dem DSP gerechnet. Diese Trennung erfolgte im Hinblick auf eine spätere DSP-Standalone-Lösung. Die Schnittstelle umfasst die in der Tabelle folgenden Parameter. Die Spalte „Richtung“ sagt aus, von wo nach wo die Parameter geschickt werden. ‚Up‘ bedeutet vom DSP ins Matlab, ‚Down‘ bedeutet vom Matlab auf den DSP.

Name	Beschrieb	Datentyp	Richtung
delay1-4	Verzögerung der Samples im CB	integer	down
samples1,2	Frequenzabtastrwerte	float[(N+1)/2]	down
coeffs1,2	Filterkoeffizienten FIR	float[N]	up
iir_grenz	Grenzfrequenz IIR-Tiefpass	float	down
ANZCOEFF	Anzahl Filterkoeffizienten	const int	up
changed1	Filterkoeffizienten samples1 geändert -> Neue Berechnung	int Flag	down/up
changed2	Filterkoeffizienten samples2 geändert -> Neue Berechnung	int Flag	down/up
changediir	iir_grenz geändert	int Flag	down/up
changedbeat	Beatfrequenz beating geändert	int Flag	down/up
changedvol	vol_phones / vol_vibes / bal_phones / bal_vibes geändert	int Flag	down/up
fc	Samplingfrequenz des DSP	const float	up
vol_phones	Volume Phones (0..1)	float	down
vol_vibes	Volume Vibratoren (0..1)	float	down
bal_phones	Balance Phones (-1..1)	float	down
bal_vibes	Balance Vibratoren (-1..1)	float	down
beating	Schwebungsfrequenz [Hz] (0..10)	float	down

## 2. 5. 2 Initialisierung / Synchronisation Matlab-DSP

Beim Aufstarten der Benutzeroberfläche lädt Matlab folgende Parameter vom DSP herauf:

- ANZAHLCOEFF
- fc

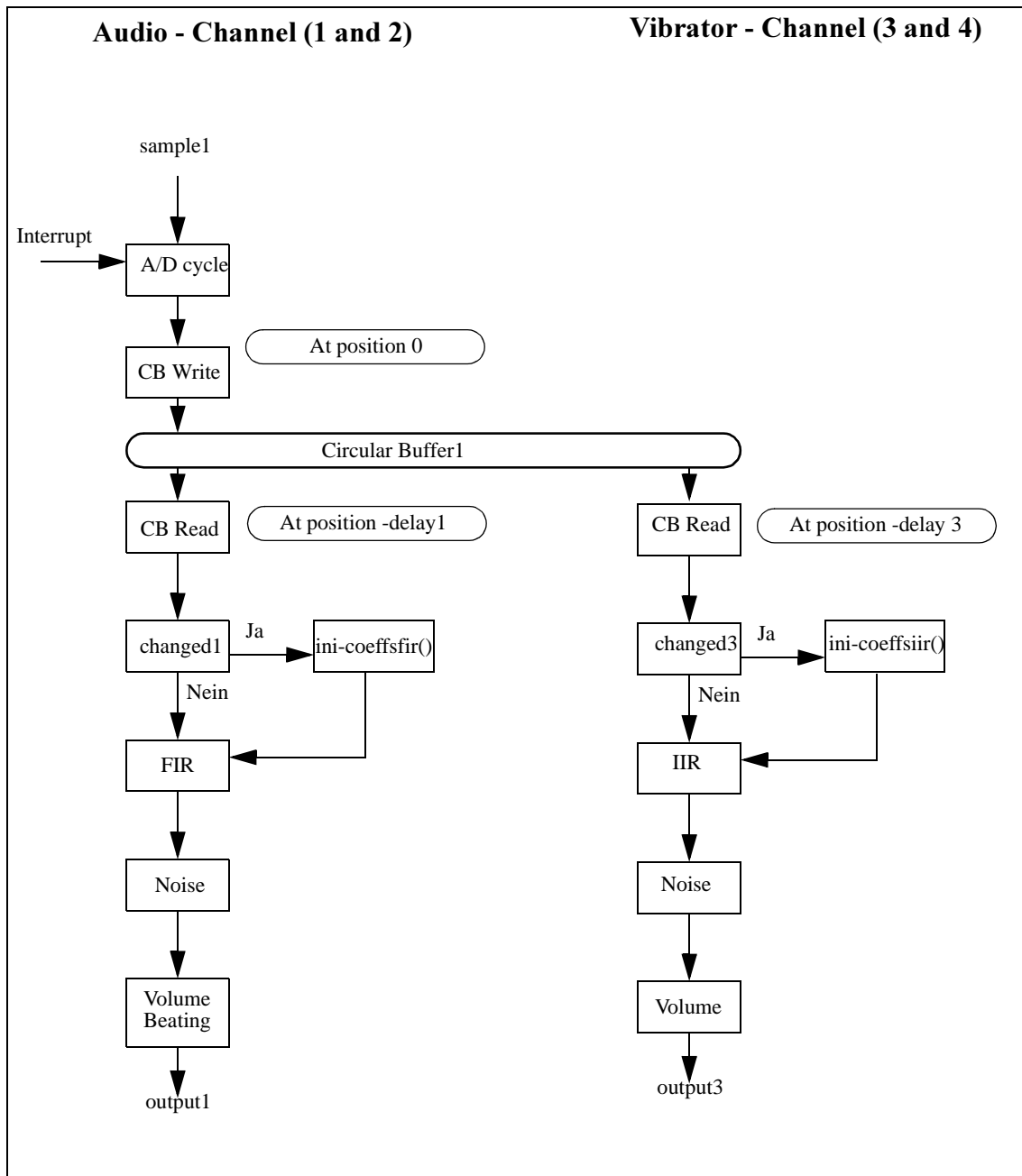
Daraus wird der Tastabstand beim Abtasten der Filterkurven berechnet. Ausserdem wird die zu Testzwecken eingebaute Anzeige des Frequenzgangs formatiert.

Danach initialisiert Matlab den DSP mit folgenden Parametern:

- samples1 -> changed1                      Equalizerstellung des linken Kopfhörerkanals
- samples2 -> changed2                      Equalizerstellung des rechten Kopfhörerkanals
- iir\_grenz -> changediir                      Default-Grenzfrequenz des Vibratorentiefpass (8000Hz)
- delay1,2,3,4                                  Default-Verzögerung der Delays (0)

Zum jetzigen Zeitpunkt ist der DSP initialisiert, d.h. es befinden sich Defaultwerte in allen Filtern und Zeitverzögerungen.

## 2.6 Flussdiagramm



Diese Struktur besteht zweimal. Einmal für den linken Kanal mit Sample1 als Eingang und Output 1 und 3 als Ausgang, sowie für den rechten Kanal mit Sample2 als Eingang und Output 2 und 4 als Ausgang.

Die A/D-Wandler arbeiten mit dem Takt  $f_c$ . Bei Musikanwendungen beträgt dieser Takt 44100 Hz. Ein Timer triggert die A/D-Wandler, so dass alle  $1/f_c$  Sekunden ein neuer Samplewert ansteht und ruft die IRQ-Handler-Routine auf. In ihr werden die Samples formatiert und die entsprechenden Manipulationen ausgeführt. Auch wird geprüft, ob die Filterkoeffizienten über Matlab geändert wurden (changed-Flag gesetzt). Wenn dies der Fall ist, wird die entsprechende Initialisierungsroutine aufgerufen.

## 2. 7 Implementierte Funktionen

### 2. 7. 1 Lautstärkenregelung / Balance

Die Lautstärkenregelung ist durch vier Faktoren realisiert, mit welchem die Samples vor der D/A - Wandlung multipliziert werden.

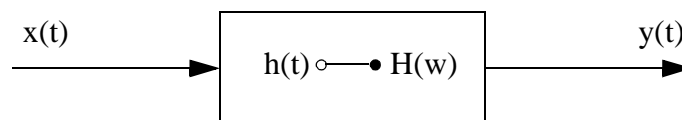
In diesen Faktoren sind Volume und Balance so miteinander verrechnet, dass bei maximalem Volume und neutraler Balance auch die maximale Leistung ansteht. Das ist wichtig, damit die D/A 's in diesem häufigen Betriebsfall voll angesteuert werden. Somit ergibt sich dort keine zusätzliche Qualitätseinbusse wegen Quantisierungsfehlern. Wird nun der Balanceregler verschoben, reduziert sich das Volumen eines Kanals. Das Volumen des Anderen bleibt sich gleich.

### 2. 7. 2 FIR-Filter

#### Grundlagen

Auf die Grundlagen der FIR-Theorie wird hier nicht weiter eingegangen. Als Nachschlagewerk diene uns die Autographie von Herr Dr. Schüeli aus der Ds-Vorlesung.

Die Grundstruktur sei hier jedoch rasch gestreift. Das FIR-Filter benutzt die Stossantwort um im Zeitbereich mittels diskreter Faltung die gewünschte Filterwirkung zu erhalten.



Bekanntlich erhalten wir das gefilterte Signal (y) aus dem Eingangssignal (x) mit dem Faltungsintegral

$$y(t) = \int_0^t x(\tau)h(t-\tau)d\tau$$

Das Faltungsintegral lässt sich nun approximativ als Summe auffassen, wobei das Argument von x() und h() vertauscht werden darf:

$$y(iT) = T \sum_{n=0}^{N-1} x[(i-n)T]h(nT)$$

Dabei entspricht T dem Abtastintervall.

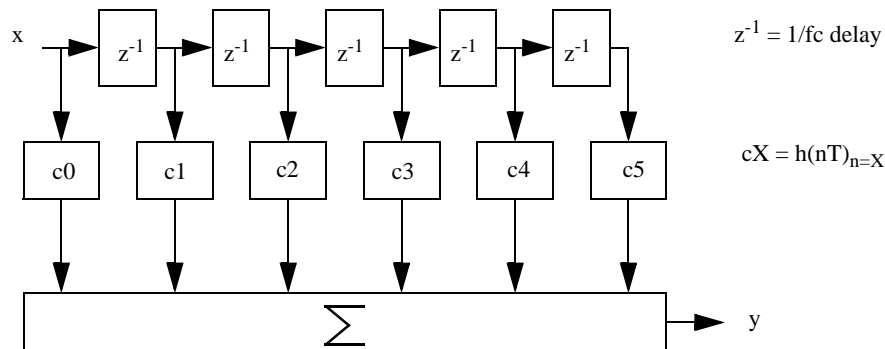
Jetzt können wir die Stossantwort h(t) noch durch die Filterkoeffizienten ersetzen:

$$c_n = Th(nT)$$

N+1 ist die Anzahl der Filterkoeffizienten. Dies deshalb, weil in den meisten Formeln Anzahl Filterkoeffizienten minus Eins, also N, benötigt wird („Telefonstangeneffekt“: 3 Telefonstangen haben 2 Zwischenräume). Der Laufindex der Filterkoeffizienten, das kleine ‚n‘ läuft von 0

bis N.

Das Folgende Signalflussbild bildet die Faltungssumme für  $N+1 = 6$  nach:

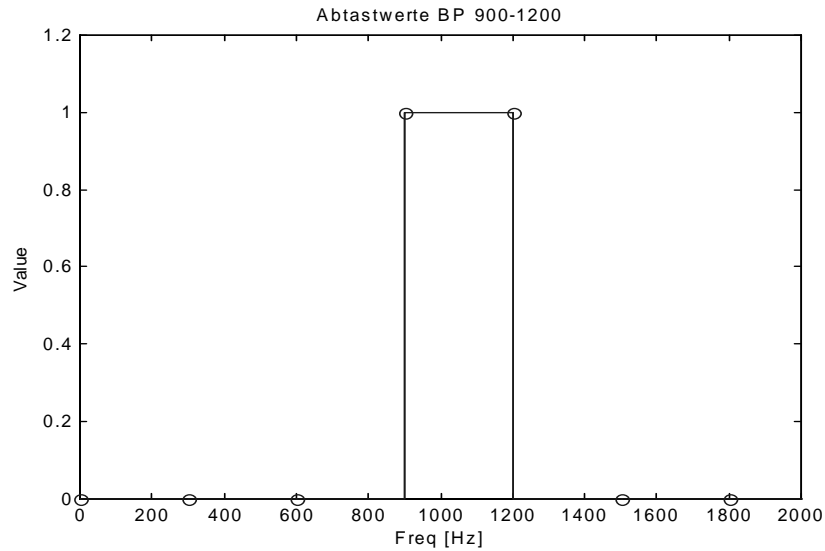


Über das Signalflussbild kommen wir zur Implementation des Filters. Die Reihe der Verzögerungsglieder entspricht einem Speicher, der mit jedem Taktzyklus vorne einen neuen Wert erhält und alle anderen um einen Platz weiterschiebt. Der Hinterste fällt dabei heraus. Diese Aufgabe löst ein Ringspeicher zeitoptimaler als ein normales Array, weil nicht die Werte umgespeichert werden, sondern bloss ein Zeiger geändert wird. Ein Ringbuffer stellt man sich als eine Art Array vor. Steht der Zeiger allerdings auf dem letzten Element dieses Arrays, und wird um eins erhöht, so befindet er sich automatisch wieder auf dem ersten Element. Die absolute Position interessiert im Folgenden gar nicht mehr. Es muss bloss noch darauf geachtet werden, dass bei jedem Samplezyklus die Position um 1 erhöht, und der Sample dann geschrieben wird. Somit überschreibt er automatisch den ältesten Sample, der sich im Ringspeicher befindet.

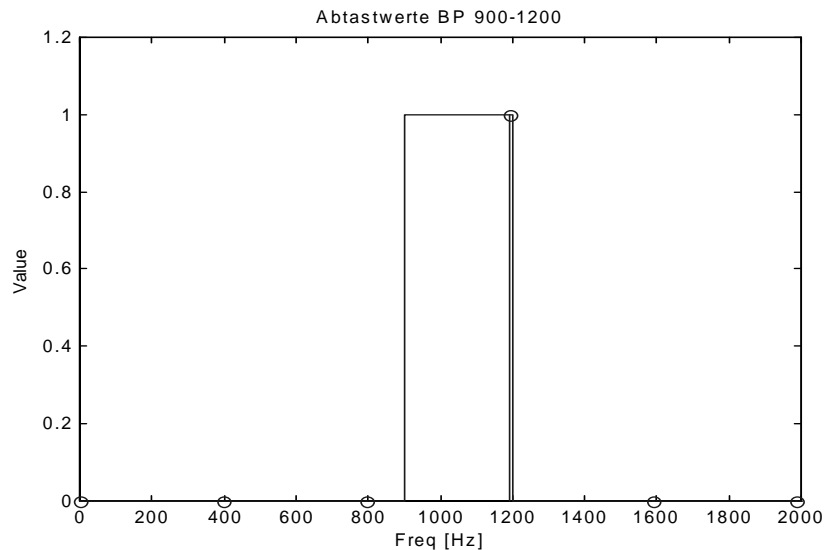
## Implementation

Je ein Equalizer soll die beiden Audiokanäle beeinflussen. Dies wurde mit zwei eigenständigen FIR-Filtern realisiert. Die Ordnung  $N+1$  wird später so hoch festgelegt, dass der DSP gerade ausgelastet ist. Im Moment erreichen wir mit Ordnungen um 201 gute Ergebnisse.

Die Filterkoeffizienten werden in der Frequenzabstasttechnik gerechnet. Auf dem GUI kann ein Frequenzgang vorgegeben werden. Diese äquidistanten Abstastwerte im Frequenzbereich werden auf die DSP-Karte geladen und dort in FIR-Koeffizienten umgerechnet. Es ist zu beachten, dass bei zu niedriger Koeffizientenzahl  $N+1$  nicht alle Schieber des Equalizers einen Einfluss auf die Filterkurve haben! Bei  $N+1 = 147$  beträgt die Auflösung noch 300 Hz. Zur Illustration ist auf der folgenden Figur der Frequenzgang eines idealen Bandpasses dargestellt. Die Kreise markieren die Frequenzabstastpunkte:



Wir haben Glück; Die Eckpunkte werden genau erwischt und es resultiert der erwartete Bandpass. Reduzieren wir nun aber die Filterordnung auf  $N+1 = 111$ , sieht derselbe Bandpass folgendermassen aus:



Anstatt des erwarteten Bandpasses mit 300 Hz Bandbreite, ergibt sich ein äusserst schmalbandiger Bandpass, da bloss ein Eins-Wert im Durchlassband ist. Zudem ist die Mittenfrequenz verschoben, da sich der Einswert bei fast 1200Hz befindet. Anzumerken ist noch, dass bei einer Filterordnung von beispielsweise  $N+1 = 100$ , dieser Bandpass gar nicht mehr aufgelöst würde.

$$\Delta f = \frac{f_c}{N} = \frac{44100}{100} = 441$$

Ein Abtastwert würde also bei  $2\Delta f = 882$  Hz liegen, der Nächste bei  $3\Delta f = 1323$ Hz. Der Durchlassbereich enthält keinen Einswert.

Diese Problematik ist über das ganze Frequenzspektrum vorhanden. Bei höheren Frequenzen kommt sie jedoch nicht allzusehr zum Tragen, weil mit zunehmender Frequenz auch die Sensibilität des Ohrs auf Frequenzunterschiede abnimmt und somit sehr schmale Bandpässe gar

nicht gefragt sind. Will man aber einen Bandpass von 300 bis 450 Hz realisieren, entstehen unweigerlich Fehler. Natürlich tauchen ähnliche Fehler auch bei Hochpässen und Tiefpässen auf, ja sogar bei beliebigen Filterkurven. Der Bandpass wurde als Demonstrationsobjekt gewählt, weil er die Problematik hübsch veranschaulicht. Mit dem gegenwärtig gewählten  $N+1$  von 201 ergibt sich ein Frequenzraster von

$$\Delta f = \frac{f_c}{N} = \frac{44100}{201} = 219.4 \text{ Hz}$$

Das ist garant für eine ausreichende Filtergenauigkeit ab Bandbreiten um 1 KHz. Darunter ist die Anzeige mit Vorsicht zu geniessen.

Kommen wir nun zur Theorie für die Implementation in Frequenzabstasttechnik.

$$H(m\Delta f); m = 0, \pm 1, \pm 2, \pm 3, \dots, \pm \frac{M}{2}$$

- $H(f)$  ist ein im Allgemeinen beliebiger Frequenzgang im äquidistanten Raster  $\Delta f$ . Der Frequenzbereich erstreckt sich von  $-f_c/2 \dots f_c/2$
- $M+1$  ist die Zahl der Abtastwerte und muss ungerade sein.

Realisiert wird dabei ein Frequenzgang mit Stützwertinterpolation durch  $\sin(x)/x$ :

$$H(f) = \sum_{m=-M/2}^{(M/2)} H(m\Delta f) \frac{\sin[\pi\tau(f-m\Delta f)]}{\pi\tau(f-m\Delta f)} \quad \text{mit der Dauer der Stossantwort:} \quad \tau = \frac{1}{\Delta f}$$

Die Dauer der Stossantwort ergibt sich auch aus der Filterlänge und der Samplingrate. Dieser Zusammenhang lautet:

$$\tau = \frac{N}{f_c}$$

- $N+1$  entspricht der Filterlänge
- $f_c$  ist die Samplingfrequenz. Für unsere Audioanwendung beträgt sie 44,1kHz.

Kombiniert man die beiden Formeln, ergibt sich das Frequenzraster:

$$\Delta f = \frac{f_c}{N}$$

Für die Filterkoeffizienten interessiert die Stossantwort im Zeitbereich. Sie lautet:

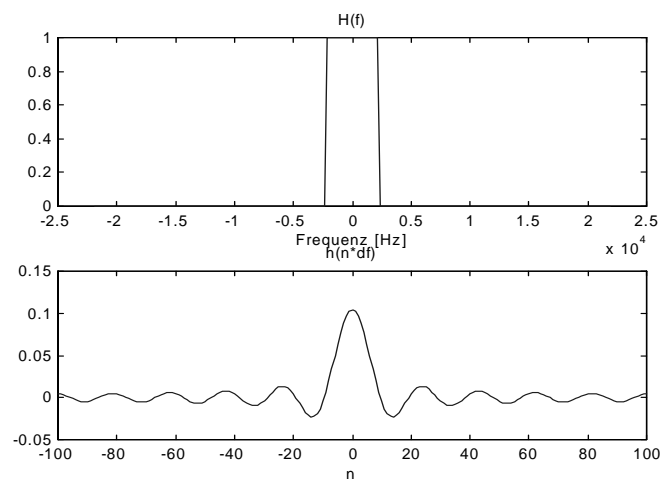
$$h(t) = \Delta f \left[ H_o(0) + 2 \sum_{m=1}^{(M/2)} H(m\Delta f) \cos(2\pi m\Delta f t) \right] \quad \text{falls} \quad |t| \leq \tau/2$$

Ausserhalb dieses Intervalles ist die Stossantwort = 0.

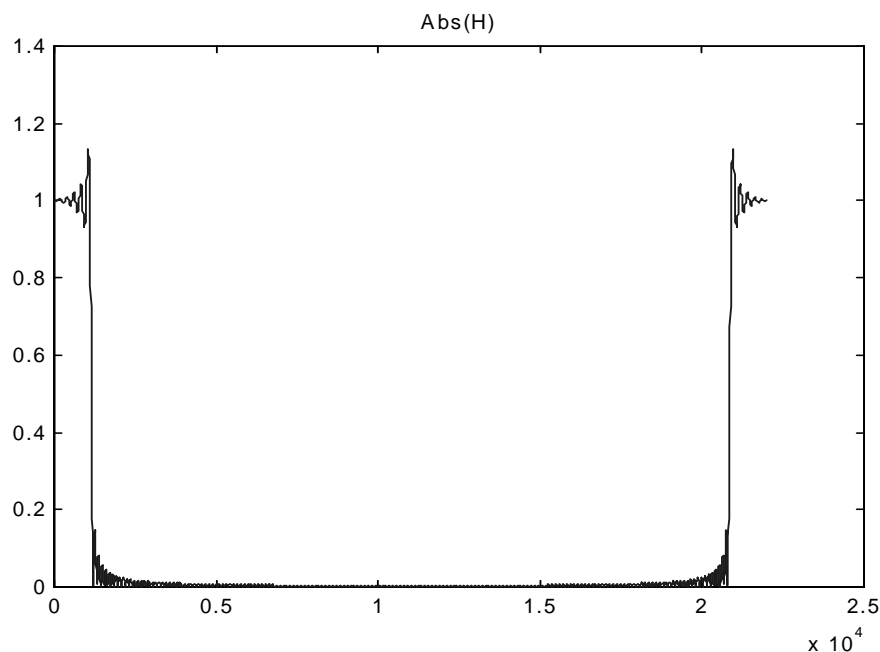
Im zeitdiskreten Fall lautet die Stossantwort für die ungerade Filterlänge  $N+1$ :

$$h(n) = \frac{1}{N+1} \left[ H_o(0) + 2 \sum_{m=1}^{(M/2)} H(m\Delta f) \cos\left(2\pi m \frac{n}{N+1}\right) \right]$$

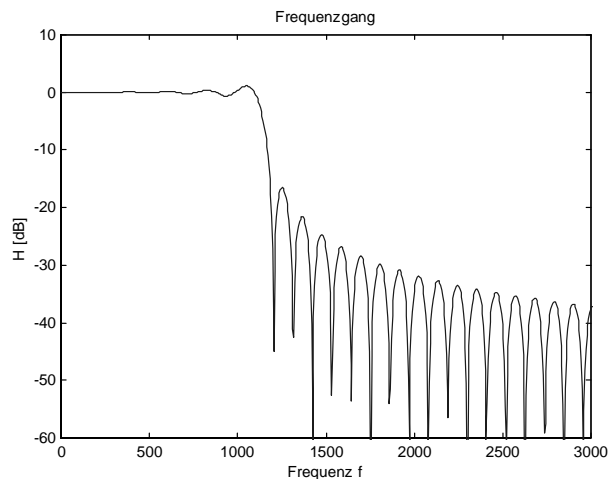
Diese Formel wurde im Matlab-m-File dl.m (siehe Anhang) implementiert. Sie liefert die Filterkoeffizienten und, durch die FFT selbiger, den Frequenzgang. Auch werden die Filterkoeffizienten auf das DSP-Board geladen. Somit kann das Filter auch in Echtzeit verifiziert werden. Der erste Plot, den das File liefert, zeigt den zweiseitigen Frequenzgang, normiert auf die Clockfrequenz, sowie die Filterkoeffizienten. Sie entsprechen der Fourier-Rücktransformierten des Frequenzganges, also der Stossantwort eines Systemes mit diesem Frequenzgang.:



Weil die Stossantwort von endlicher Länge ( $N+1$ ) ist, approximiert sie den Frequenzgang lediglich. Dies allerdings mit ausreichender Genauigkeit. Bildet man die FFT der Stossantwort zeigt sich folgender, realer Frequenzgang:



Zoomt man in den interessanten Bereich und wählt zudem den dB-Masstab für die y-Achse, zeigt sich der reale Frequenzgang so:



Zu sehen ist eine Sperrbereichsdämpfung von ca. -17dB. Das entspricht der Dämpfung eines Tiefpasses ohne Zwischenwert, berechnet durch Frequenzabstasttechnik.

In einem nächsten Schritt wurde der Algorithmus direkt auf dem DSP implementiert. Ein Flag meldet bei Veränderung der Filterkurve im Interface, dass sich die Frequenzabstastwerte geändert haben, worauf der DSP in der Subroutine `ini_coeffsfir()` (siehe Anhang) die Filterkoeffizienten  $c_n$  neu rechnet.

Da das FIR Filter die zeitintensivste Berechnung des ganzen Programmes ist, wurde es direkt in Assembler programmiert und optimiert. Der DSP bietet speziell für FIR Filter den so genannten MAC Befehl an, welcher paralleles Multiplizieren und Addieren erlaubt. Der Effizienz des Algorithmus kann um ein Vielfaches gesteigert werden, wenn er direkt in Assembler programmiert wird.

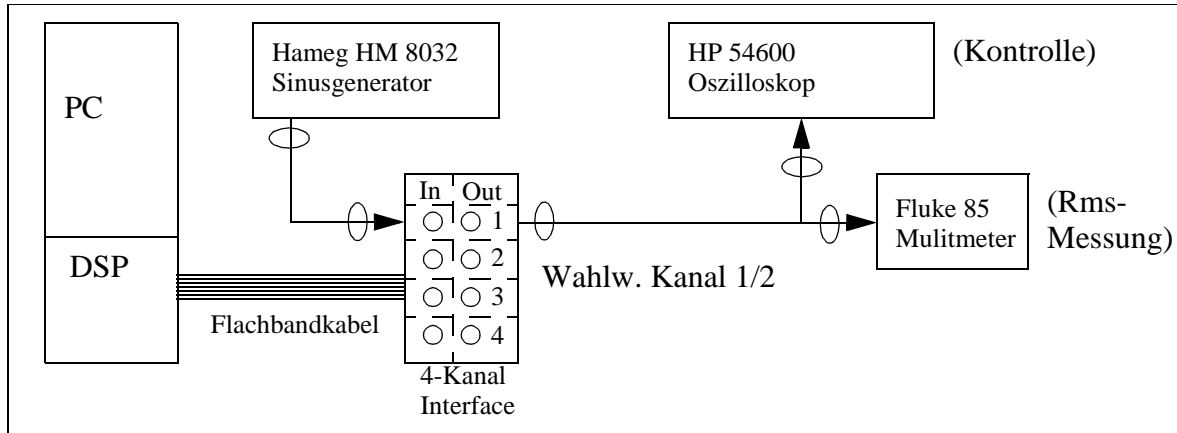
In einem ersten Schritt untersuchten wir ein in C geschriebenes FIR Filter, mit welchem wir Ordnungen  $N+1$  um die 50 erreichten. Dann bauten wir den von André Rüegg in Assembler geschriebenen FIR Filter ein. Das File heisst `Fir.h` und befindet sich im Anhang. Mit diesem optimierten Filter konnten wir eine Filterordnung  $N+1$  um 650 erreichen! Da zwei parallele arbeitende Filter benötigt werden und die Schwebung, der IIR Tiefpass und die Volumen Regelung auch noch Rechenzeit in Anspruch nehmen, erreichen wir aktuell eine Filterordnung  $N+1$  von 201.

## Verifikation

Die FIR-Filter wurden direkt im Betrieb verifiziert. Als Signal dient ein Sinus mit 8V Amplitude. Gemessen wurde das Rms-Signal des Ausganges.

Der Messaufbau bestand aus einem Sinusgenerator als Signalquelle, einem KO zur optischen Kontrolle des Ausgangssignals (Aliasing, Nyquist) sowie einem Multimeter mit  $V_{RMS}$  Anzei-

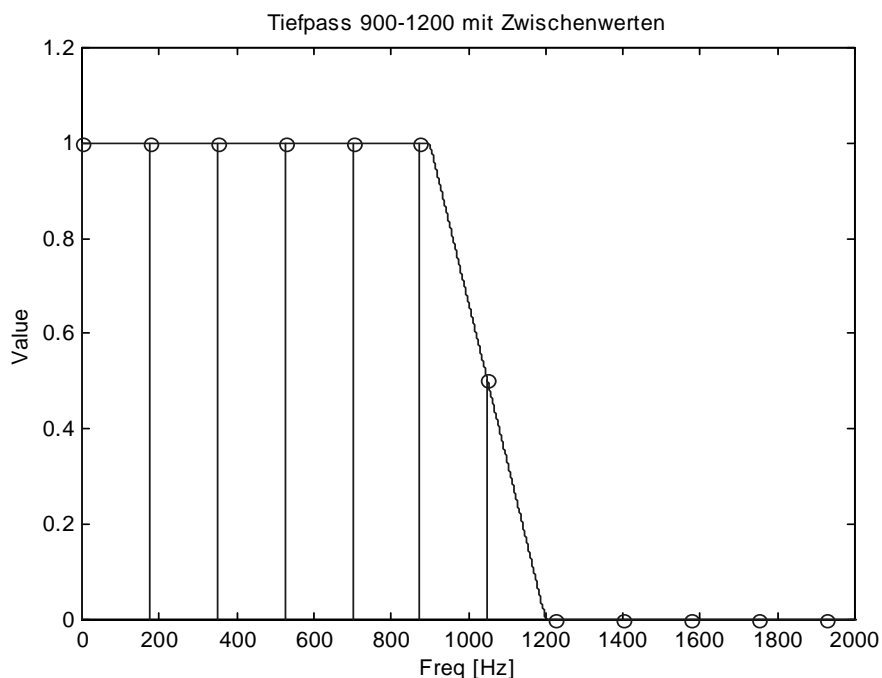
ge, welches auf Sinussignale geeicht ist.



Ein Hoch-, Band-, und Tiefpass wurden ausgemessen.

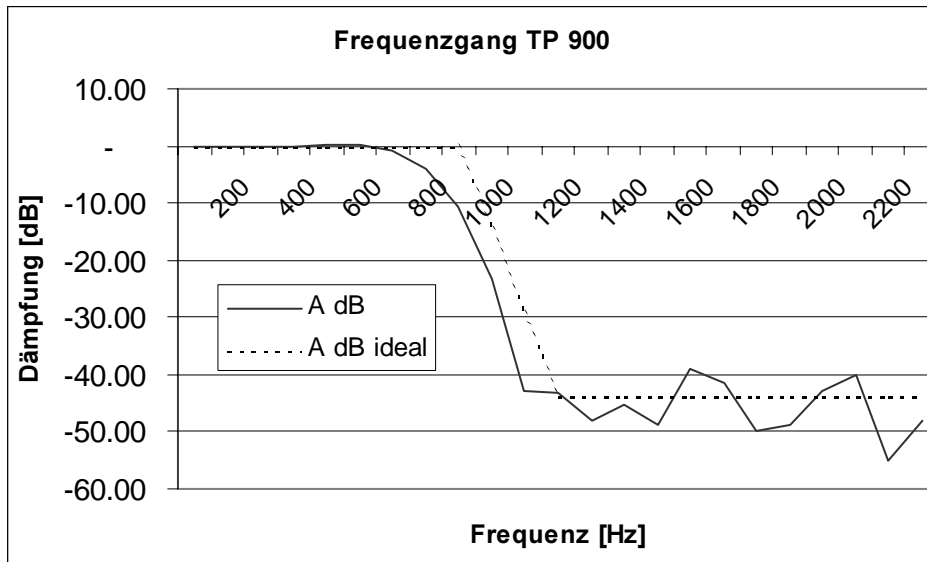
Laut Theorie liefert ein Tiefpass, berechnet in Frequenzabtasttechnik, -17dB Sperrbereichsdämpfung. Falls ein Zwischenwert der Höhe 0.38 eingefügt wird, erhöht sich die Sperrbereichsdämpfung auf -44 dB. Dies allerdings auf Kosten der Filtersteilheit; Die Forderung heisst, ohne Zwischenwert in nur einem Intervall vom Durchlass- in den Sperrbereich zu wechseln. Mit Zwischenwert reduziert sich die Forderung, dies innerhalb von zwei Intervallen zu tun.

In unserem Falle befinden wir uns irgendwo zwischen diesen beiden Varianten. Unser Fenster wird ja abgetastet und enthält je nach Form keinen, einen oder mehrere Zwischenwerte. Als Beispiel diene hier ein Tiefpass, welcher von 900 bis 1200 Hz sukzessive zumacht.  $N+1 = 251$ , Frequenzraster 175.7Hz.



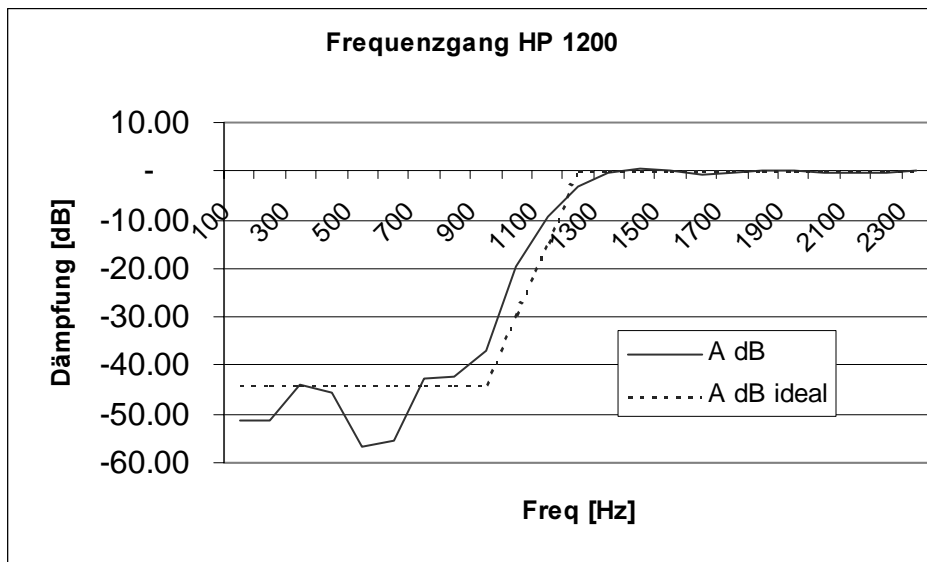
Ausgezogen dargestellt ist wiederum der gewünschte Frequenzgang. Die Kreise stellen die Abtastwerte dar. Es ist schön ersichtlich, wie durch die Art und Weise, wie wir die Abtastwerte berechnen, automatisch Zwischenwerte entstehen.

Genau diesen Tiefpass nehmen wir jetzt messtechnisch unter die Lupe. Der gestrichelte, vorgegebene Frequenzverlauf ergibt sich durch die Eingabe am Equalizer. Gemessen wurde alle 100 Hz. Es resultiert folgender Frequenzgang:



Es ergibt sich eine Sperrbereichsdämpfung von ca. -40 dB. Wir sind also von der Dynamik her etwa in dem Bereich eines Tiefpass mit Zwischenwert.

Als nächstes wurde ein Hochpass untersucht. Der vorgegebene Frequenzgang ist wiederum gestrichelt dargestellt. Die mit Fenstertechnik plus Zwischenwert erreichbare Dämpfung von -44dB wurde hier erreicht.

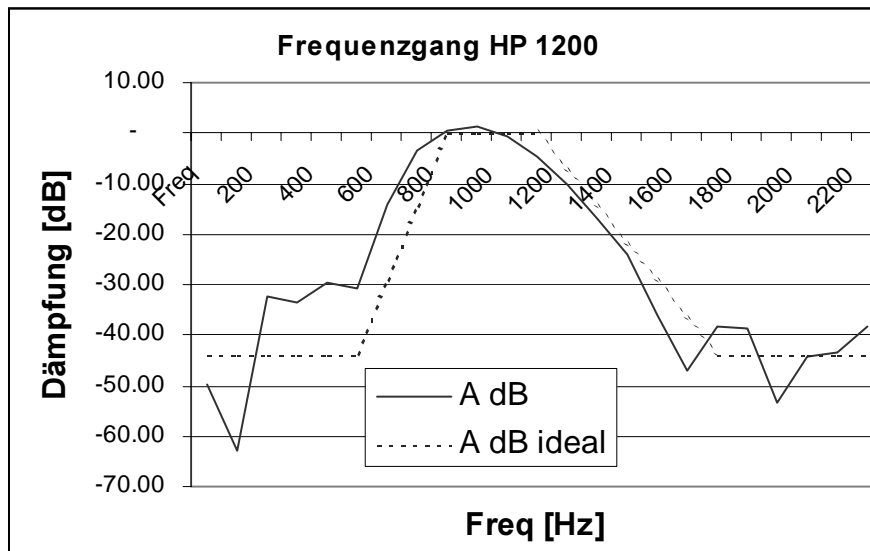


Als letztes Versuchsobjekt diente ein Bandpass, der von 600 bis 900 Hz öffnet, und von 1200 bis 1800 Hz wieder schliesst. Hier werden etwas schlechtere Unterdrückungswerte erreicht. Im unteren Sperrbereich erreichen wir -30 dB, im oberen knapp -40 dB. Der Bandpass wurde mit einem Filter der Ordnung  $N+1 = 201$  gerechnet. Das ergibt einen Frequenzabstand von knapp

220Hz.

$$\Delta F = \frac{f_c}{N} = \frac{44100}{201} = 219.4\text{Hz}$$

Der Durchlassbereich des Bandpasses beträgt 300 Hz. Das heisst, dass der Bandpass nicht mehr optimal erfasst werden kann. Dieser Bandpass hat bloss ein bis zwei Einswerte. Wir kommen also langsam an die Grenze der Auflösbarkeit.

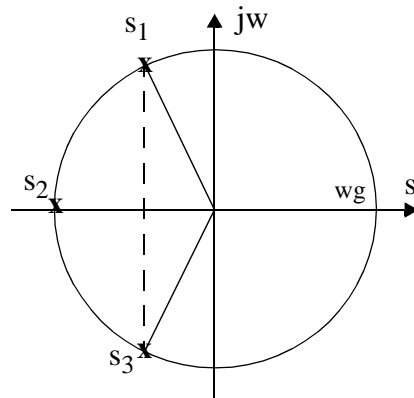


Das implementierte FIR Filter läuft mit einer Dynamik, die vom Hörgefühl her gute Resultate liefert. Auf dieser Grundlage wäre jedoch mit Fensterfunktionen im Sinne einer Optimierung noch mehr Dynamik herauszuholen. Im Rahmen dieser Arbeit beschränken wir uns auf diese Grundlage.

### 2. 7. 3 IIR - Filter / Tiefpass der Vibratoren

Da die Vibratoren nur relativ tiefe Frequenzen übertragen und die hohen Frequenzen eine unnötige Belastung darstellen, bauten wir dieses Filter ein. Es werden keine hohen Anforderungen, wie eine steile Flanke oder eine starke Dämpfung gestellt. Da die Rechenleistung, die ein IIR-Filter niedriger Ordnung benötigt, vernachlässigbar ist, lag ein solches Filter nahe. Wir entschieden uns für einen Butterworth-Tiefpass 3. Ordnung.

Das spezielle an den Butterworth Filtern ist deren optimal flacher Amplitudengang. Um dies zu erreichen ist folgende Polvorgabe zu machen: Die Pole müssen gleichmässig auf einem Halbkreis, welcher konzentrisch ist zum Einheitskreis, und dessen Radius der Grenzfrequenz entspricht, verteilt werden. Die imaginäre Achse ist dabei die Symmetrieachse. In unserem Fall, d.h., beim Tiefpass 3. Ordnung ist der Winkelabstand der Pole 60 Grad.



Da die Pole  $s_1$  und  $s_3$  konjugiert-komplex sind, lautet die Übertragungsfunktion folgendermaßen:

$$H(s) = \frac{\omega_g^3}{(s + \omega_g)(s^2 + \omega_g s + \omega_g^2)}$$

Um den Term zu transformieren, ist eine Partialbruchzerlegung notwendig. Sie ergibt:

$$H(s) = \frac{\omega_g}{s + \omega_g} - \frac{\omega_g s}{s^2 + \omega_g s + \omega_g^2}$$

Durch Anwendung eines Z-Lexikons erhält man:

$$H(z) = k \left( \frac{az^{-1} - 1}{1 - bz^{-1} + cz^{-2}} + \frac{1}{1 - dz^{-1}} \right)$$

wobei:

$$a = e^{-\frac{\omega_g T}{2}} \left( \cos\left(\frac{\sqrt{3}}{2}\omega_g T\right) + \frac{1}{\sqrt{3}} \sin\left(\frac{\sqrt{3}}{2}\omega_g T\right) \right) \quad b = 2e^{-\frac{\omega_g T}{2}} \cos\left(\frac{\sqrt{3}}{2}\omega_g T\right)$$

$$c = e^{-\omega_g T} \quad d = e^{-\omega_g T}$$

Um  $k$  zu berechnen, muss die Übertragungsfunktion bei  $H(s=0)$ , also Gleichstrom, auf 1 gesetzt werden. In der Z-Ebene heisst dies, dass  $H(z=1)$  1 sein muss:

$$H(z=1) = k \left( \frac{a-1}{1-b+c} + \frac{1}{1-d} \right) = 1$$

$$k = -\frac{(d-1)(b-1-c)}{ad-a+b-c-d}$$

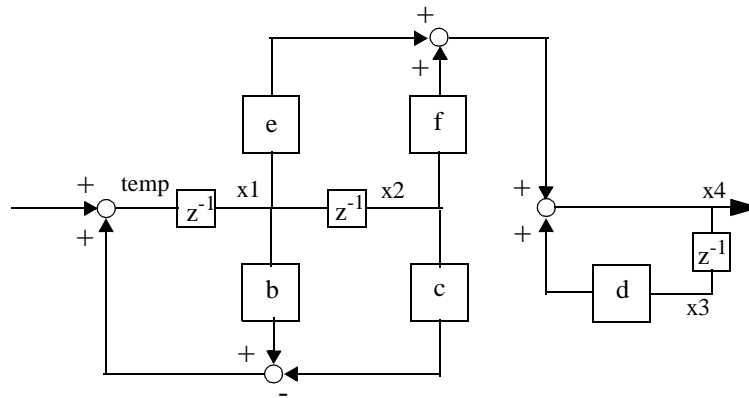
Nun wird die Filterstruktur gewählt. Wir entschieden uns für die Kaskadenform, d.h. eine multiplikative Form von Grundgliedern erster und zweiter Ordnung. Auf das oben hergeleitete  $H(z)$  angewendet kommt dabei

$$H(z) = \frac{ez^{-1} + fz^{-2}}{1 - bz^{-1} + cz^{-2}} \frac{1}{1 - dz^{-1}}$$

heraus mit:

$$e = k(d + a - b) \quad f = k(c - ad)$$

Aus dieser Form kann das Flussdiagramm abgelesen werden.



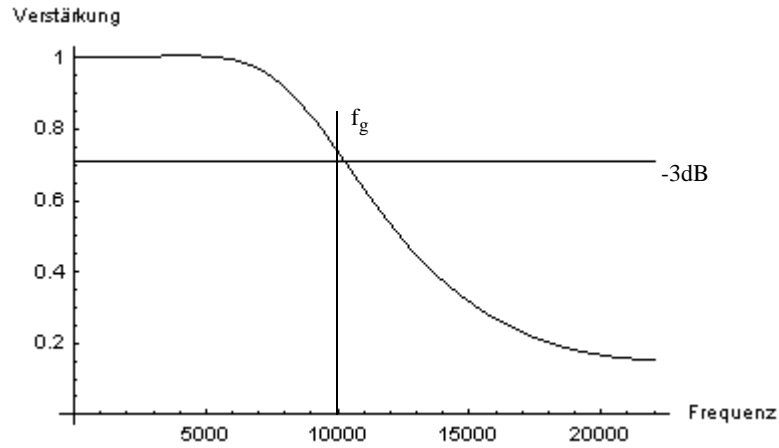
Diese Struktur ist auf dem DSP implementiert. Die Beschriftungen der Speicher (x1-x4, temp) korrespondieren mit den Speicherbezeichnungen auf dem DSP. Wird der Grenzfrequenzschieber in der Benutzeroberfläche bewegt, so setzt Matlab auf dem Sharc ein changed-Flag. Bei jedem Durchlauf der Interruptroutine wird dieses Flag abgefragt. Ist das Flag true, werden die Koeffizienten mit der Funktion ini\_coeffsiir() neu berechnet und in die Struktur eingesetzt. Der C-Code ist im Anhang zu finden.

Zur Verifizierung der oben hergeleiteten Übertragungsfunktion nahmen wir Mathematica zur Hilfe.

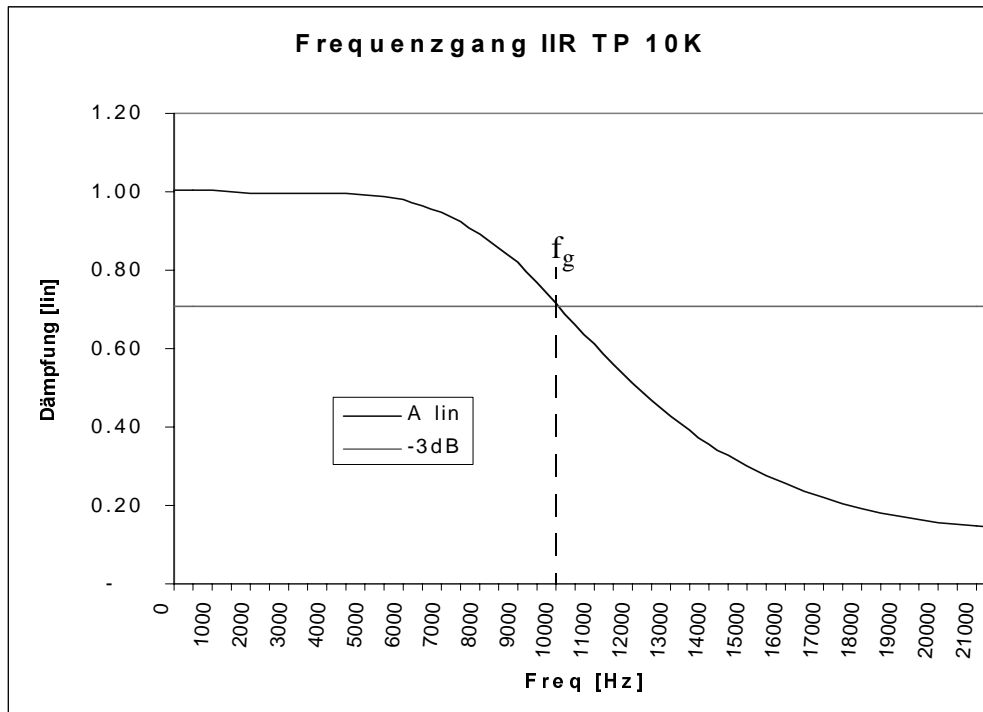
Die Frequenzganganalyse erfolgt bekanntlicherweise wieder in der s-Ebene. Als Abbildungsvorschrift in die s-Ebene gilt:

$$H(z = e^{sT})$$

Die ganzen Formeln seien hier nicht wiedergegeben da sie formal relativ umfangreich sind. Die folgende Abbildung zeigt das Betragsspektrum mit der -3dB-Linie (entspricht 0.7079) für eine Grenzfrequenz von 10kHz.

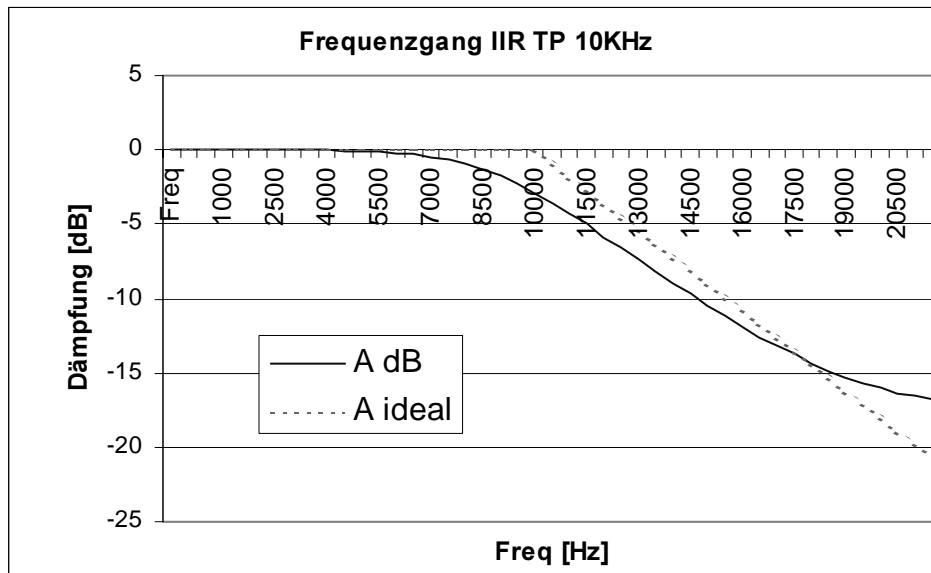


Der Messaufbau entspricht genau dem von der Verifikation des FIR Filters. Natürlich wird das Signal im Gegensatz zur FIR Messung jetzt wahlweise an Kanal 3 oder 4 abgenommen. Misst man den Tiefpass aus, ergibt sich für  $f_g = 10\text{kHz}$  folgende Übertragungskennlinie:



Die Übereinstimmung von Theorie und Praxis ist so gut, dass sich ein weiterer Kommentar erübrigt.

Die logarithmische Darstellung der Frequenzachse lässt einen linearen Abfall erwarten. Gestrichelt dargestellt ist der asymptotische Verlauf eines Tiefpasses 3. Ordnung mit Grenzfrequenz 1000 Hz. Die Asymptote fällt mit  $-60\text{ dB pro Oktave}$ .



In dieser Darstellungsart fällt auf, dass sich der Frequenzgan im oberen Sperrbereich wieder abflacht. Das rührt daher, dass bei diesem getakteten System der Frequenzgang periodisch ist. Die halbe Grenzfrequenz liegt bei 22050Hz. Dort wird die Übertragungskennlinie gespiegelt. Deshalb die Abflachung in der Nähe dieses Bereiches.

#### 2. 7. 4 Zeitverzögerungen

Gefordert ist eine individuelle Verzögerung der vier Kanäle zueinander. Das heisst, jeder Kanal kann gegenüber jedem anderen verzögert werden. Dies natürlich im Rahmen des maximalen Speicherplatzes, welcher für insgesamt 40'000 Samples ausreicht. Das entspricht bei einer Clockfrequenz von 44,1 kHz und Stereosignal (zwei Speicher, siehe auch Signalflussbild) knapp einer halben Sekunde.

$$T_{Delay} = \frac{Bufferlaenge}{f_c} = \frac{20000}{44100} = 454ms$$

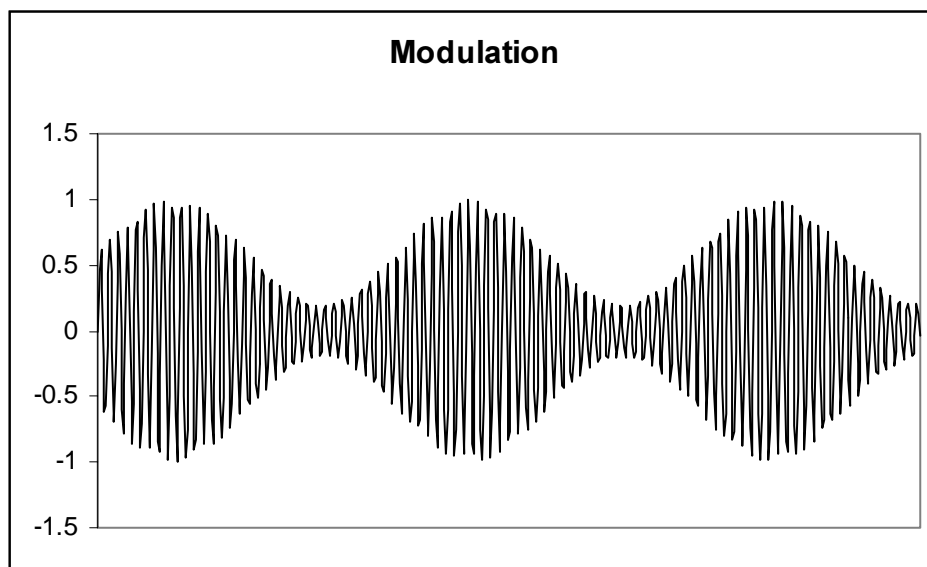
Implementiert wurde eine speicherplatzoptimierte Lösung. Dabei werden die anstehenden Samples für links und rechts in je einen Ringspeicher geschrieben, welcher die maximale Länge hat. Gelesen wird über einen Pointer der auf den Speicher zeigt. Wird in der Benutzeroberfläche ein Verzögerungsschieber bewegt, so wird einfach dieser Pointer um entsprechend viele Stellen zurück-, resp. vorgesetzt und von dort gelesen. Dies geschieht separat, jedoch im selben Speicher für Audio- und Vibratorsignal.

Die Implementation erfolgte direkt in Assembler. C stellt zwar komfortable Routinen zur Verfügung um mit Ringspeichern zu arbeiten, jedoch kann man mit ihnen die Möglichkeiten nicht voll ausschöpfen. Um eine variable Verzögerung zu erzielen, muss auch die Sprungweite des Lese/Schreib-Pointers variabel sein. Dies kann nur direkt in Assembler realisiert werden. Im Code (music.c) ist der nicht lauffähige C- Befehl auskommentiert und gleich darunter in Assembler realisiert.

#### 2. 7. 5 Schwebung

In einem Bericht unseres Auftraggebers ist das in der Physik als Schwebung bekannte Phänomen im Zusammenhang mit Musiktherapie beschrieben. Dort werden niederfrequente Schwingungen als für das Hirn stimulierend beschrieben. Die Idee von Herr Schiftan war es, eine niederfrequente Schwebung der Musik zu überlagern.

Realisiert haben wir diese Schwebung, indem wir die Ausgänge der Kopfhörer mit einem Sinus, respektive mit einem Cosinus der gewünschten Frequenz multiplizierten. Das entspricht dem Prinzip der Amplitudenmodulation. Dabei hat es sich herausgestellt, dass es vom Zuhörer als unangenehm empfunden wird, wenn der Sinus resp. der Cosinus bis null aussteuert. Daher wurde die Amplitude des Modulationssignals auf 0.2 bis 1.0 festgelegt. Das modulierte Signal sieht folgendermassen aus:



Dies ist das Signal, z.B. auf dem linken Audiokanal. Das Signal auf dem rechten Kanal ist mit einem  $180^\circ$  Phasenverschobenen Sinus moduliert. Somit hat, wenn ein Kanal das Minimum erreicht, der andere gerade das Maximum.

Im Moment ist nicht vorgesehen, den Modulationsgrad online zu ändern. Er ist als konstante SINAMPL (entspricht  $V_{\text{peak}}$  des Modulationssignals) definiert in music.c (vgl. Anhang). Hat SINAMPL den Wert 0.5, entspricht dies einem Modulationsgrad von 1. Defaultmässig ist der Wert 0.4, was auch obiger Abbildung entspricht.

## 2. 8 Bedienung

---

Sowohl das m-File musicamedica3.m als auch music.c muss in einem dem Matlab bekannten Pfad sein. Im Matlab Command-Window ist der Befehl „musicamedica3“ einzugeben. Damit wird das GUI gestartet und der DSP initialisiert. Das GUI sollte ab diesem Zeitpunkt selbsterklärend sein.

## 2. 9 Probleme

---

### **2. 9. 1 Rauschsignal auf DSP Ausgängen**

Dies war wohl das zeitaufwendigste Problem. Auf allen Kanälen hatten wir verschiedenfrequente Schwingungen.

Erst verdächtigten wir eine fehlerhafte Implementation der Filteralgorithmen auf dem DSP. Bald stellten wir aber fest, dass das Problem auch bestehen blieb, wenn wir nur Samples einlesen und direkt wieder ausgaben.

Als nächstes vermuteten wir eine Einkopplung von PC-Prozessorsignalen. Abschirmen, kürzere Kabel, Einbau in einen anderen Slot sowie Ausbau aller überflüssigen Baugruppen schafften keine Abhilfe. Der Assistent aus dem Meo-Labor, Peter Roffler, baute daher die Karte in einen langsameren PC ein. Auch dieser Versuch war erfolglos, das Rauschen blieb bestehen.

Der Verdacht lastet nun auf der DSP-Karte selber, was jedoch nicht verifiziert werden konnte, da die Zeit, sie einzuschicken zu knapp war.

### **2. 9. 2 Defekt der DSP Karte**

Während der ganzen Dauer unserer Arbeit, stiessen wir immer mal wieder auf unerklärliche Ungereimtheiten im Zusammenhang mit der DSP-Karte. Manchmal liess sie uns im Stich, um kurz darauf wieder einwandfrei zu arbeiten. Am Tage unserer Vorführung der Studienarbeit verabschiedete sie sich schliesslich vollends. Sämtliche vorhin lauffähigen Testprogramme liefen nicht mehr. Bis zum jetzigen Zeitpunkt der Abgabe konnte kein Fehler gefunden werden. Verschiedene Tests, unter Mithilfe von André Rüegg, festigten der Verdacht, dass die Umgebung, sprich PC und Software in Ordnung ist und sich der Defekt auf der Karte selbst befindet.

### **2. 9. 3 Zeitprobleme / nicht Erreichtes**

Wie wahrscheinlich fast bei jeder Studienarbeit, fehlte auch hier die Zeit um alle Anforderungen des Pflichtenhefts zu erfüllen. Konkret war es bei uns die Beimischung eines Rauschen und die Beimischung eines Beats welche wir nicht implementieren konnten.

## **2. 10 Schlusswort**

---

Dank dieser Arbeit hatten wir erstmals die Möglichkeit, auf einem DSP eine grössere Arbeit durchzuführen. Ausserdem konnten wir die Filtertheorien, die in der Vorlesung behandelt wurden, anwenden und so vertiefen. Ebenfalls interessant war es, im Anfangsstadium der Arbeit mit der Veränderung von Musiksignalen zu experimentieren.

Etwas schade war die Tatsache, dass die gestellten Aufgaben signalverarbeitungstechnisch nicht sehr komplex waren. Auch war der zeitmässige Aufwand der grafischen Oberfläche auf Matlab relativ gross und nicht sehr lehrreich. Die Probleme mit der DSP-Karte frassen ebenfalls zu viel Zeit.

Motivierend war es wiederum, ein Problem von einem ‚richtigen Kunden‘ gestellt zu bekommen. Es zeigte sich aber auch, dass es nicht immer ganz einfach ist, eine einwandfreie ‚Schnittstelle‘ zwischen der Technik und dem Normalverbraucher darzustellen.

Danken möchten wir unserem Betreuer Andreas Ehrensperger für die unkomplizierte Betreu-

ung und die wertvollen Ratschläge. Auch an Herr Schüeli, Dozent für digitale Signalverarbeitung, André Rüegg, Assistent im Ds-Labor, und Peter Roffler, Assistent im Meo-Labor, sei ein Dank für Unterstützung bei allerlei kleineren und grösseren Problemen gerichtet.

Rapperswil, 6.Juli 1999

Christian Schläpfer

Stefan Heer

# III. Anhang

## 3. 1 Software

### 3. 1. 1 Quellcode

Name	Beschrieb
musicamedica3.m	GUI unter Matlab zur Steuerung des DSP
music.c	DSP Quellcode in C. Hat Interface zum GUI

### 3. 1. 2 Header File

Name	Beschrieb
fir.h	Implementiert laufzeitoptimiertes FIR Filter / by A. Rüegg
def.h	Ausgelageter Code von music.c, zwecks Übersichtlichkeit
bitsibb2.h	Boardspezifische definitionen / by BittWare Research Systems

### 3. 1. 3 Hilfs- und Interfacesoftware

Interface.m	Matlab Interface zur einfachen Kommunikation mit dem DSP Board
fabtast.m	Berechnet Koeffizienten in Frequenzabtasttechnik
dl.m	Lässt Koeffizienten auf dem DSP rechnen und vergleicht sie mit den in fabtast.m berechneten Koeffizienten.

Die aufgelisteten Dateien sind in dieser Reihenfolge im Anschluss an diese Seite zu finden.