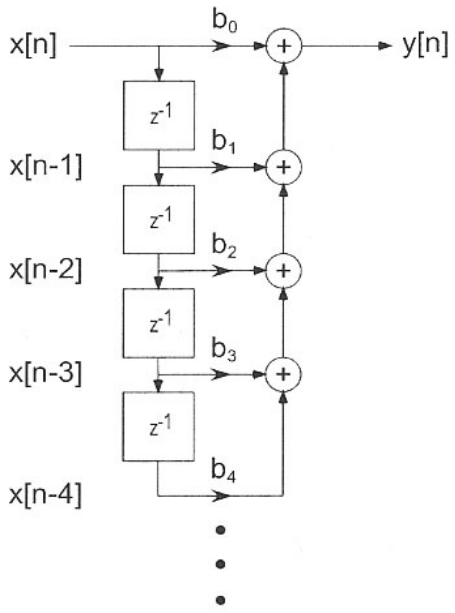
Übungseinheit 3

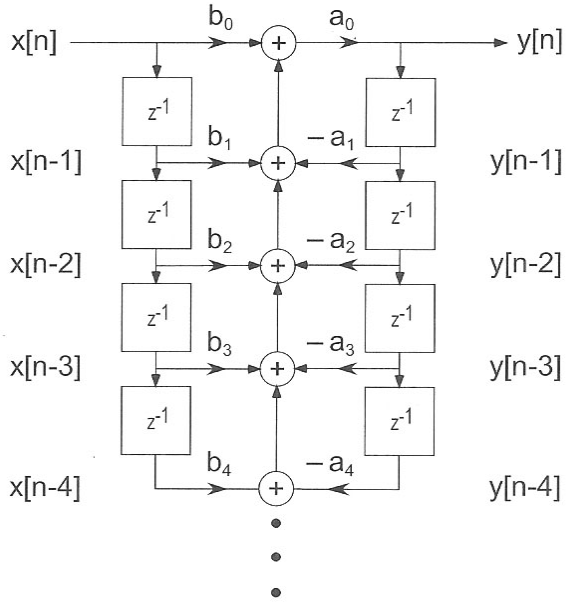
FIR und IIR Filter

In dieser Übungseinheit sollen verschiedene Effekte mittels FIR (finite impulse response) und IIR (infinite impulse response) Filter implementiert werden.

FIR Filter sind zeitdiskrete Filter. Ein FIR Filter liefert, wie der Name schon sagt, eine endliche Impulsantwort. Wird beispielsweise ein FIR Filter n-ter Ordnung mit einem Testsample und anschließend Samples mit dem Wert null angeregt, so erhält man n+1 Ausgaben bevor alle Ausgabewerte den Wert null annehmen. Daher ist die Impulsantwort endlich.
Ein FIR Filter n-ter Ordnung besitzt n+1 Filterkoeffizienten. Die Berechnung des Ausgabesignals erfolgt durch Multiplikation der verzögerten Eingabesignale mit den Filterkoeffizienten und anschließender Aufsummierung nach folgendem Blockdiagramm:

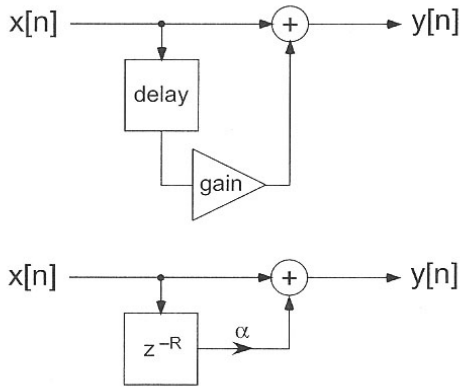


IIR Filter haben eine unendliche Impulsantwort. Mathematisch betrachtet wird die Ausgabe niemals den Wert null erreichen, wenn wiederum ein Testsample und anschließend Samples mit dem Wert null das System anregen.
FIR Filter können einfach implementiert werden und sind stabil. IIR Filter müssen hingegen nicht zwingend stabil sein. Für bestimmte Anwendungen jedoch können diese Vorteile gegenüber FIR Filtern haben und sollten in diesen Fällen bevorzugt werden.
Um spezielle Filterkriterien zu erfüllen könnte ein FIR Filter sehr hoher Ordnung benötigt werden. In den meisten Fällen kann ein IIR Filter mit einer sehr viel niedrigeren Ordnung dieselben Filterkriterien erfüllen.
Weiters sind bereits viele Filterdesigninformationen von analogen Filtern bekannt. Analoge Filter sind IIR Filter, deshalb bietet sich an, bestehendes Wissen zu Nutzen um digitale IIR Filter zu implementieren.
Im Gegensatz zu einem FIR Filter werden zusätzlich die verzögerten und gefilterten Ausgabesignale mit Filterkoeffizienten multipliziert und aufsummiert dem aktuellen Ausgabesignal hinzugefügt:


x[n]
b0
a0
y[n]
z-1
z-1
x[n-1]
b1
– a1
y[n-1]
z-1
z-1
x[n-2]
b2
– a2
y[n-2]
z-1
z-1
x[n-3]
b3
– a3
y[n-3]
z-1
z-1
x[n-4]
b4
– a4
y[n-4]

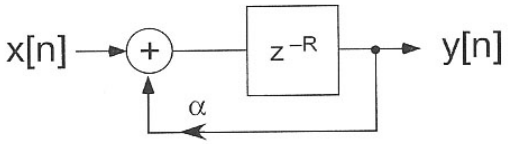
Effekte

Ein *einmalig auftretendes Echo* kann durch einen einfachen FIR Filter implementiert werden. Dabei kann der Filterkoeffizient als Lautstärke verstanden werden, mit der das Signal nach einer gewissen Verzögerung auftritt:



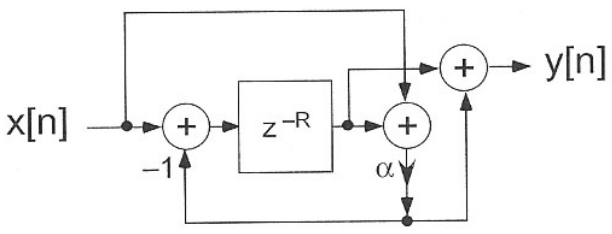
Das eingehende Signal wird mit einem skalierten, verzögerten Signal addiert. Die Verzögerung ist konstant und ergibt in Relation mit der Abtastrate die Zeit nach der das Echo auftritt.

Ein *multiples Echo* wird hingegen mit einem einfachen IIR Filter implementiert.

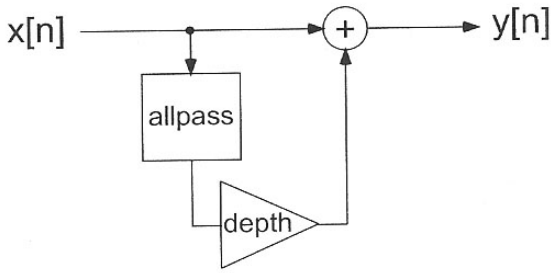


Hierbei wird nicht das verzögerte Eingangssignal für die Berechnung herangezogen, sondern das verzögerte Ausgangssignal. Um Stabilität zu gewährleisten sollte der Verstärkungsfaktor kleiner als 1.0 gewählt werden.

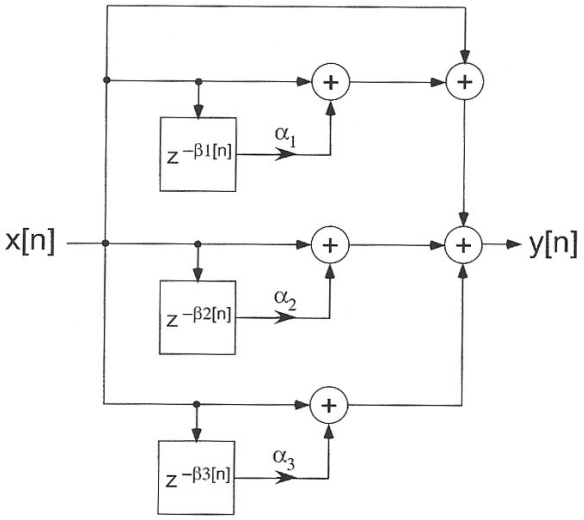
Der *Phaser Effekt* wird mithilfe eines allpass Filters erzeugt. Eine Implementierung eines solchen allpass Filters sieht folgendermaßen aus:



Für den Phaser Effekt muss jedoch eine sich langsam verändernde Verzögerungszeit verwendet werden. Eine Sinusfunktion mit einer Frequenz von 1 Hz scheint angebracht zu sein. Dieser allpass Filter kann nun für den Phaser Effekt verwendet werden. Die Ausgabe des allpass Filters wird dabei skaliert zum Eingangssignal hinzuaddiert:

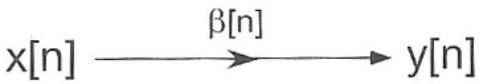


Der *Chorus Effekt* lässt einen Musiker so klingen, als ob mehrere Musiker dieselben Noten spielen würden. Dazu wird das Eingangssignal durch mehrere FIR Filter gefiltert. Die jeweiligen Ausgabesignale werden zum Eingangssignal wiederum addiert:



Wie beim Phaser Effekt werden auch hier die Verzögerungszeiten variiert. Für einen tollen Sound sollten die Verzögerungen und Verstärkungen der einzelnen FIR Filter unabhängig voneinander gewählt werden.

Der *Tremelo Effekt* variiert die Amplitude des Ausgangssignals. Dies kann durch Multiplikation mit einer Sinusfunktion, oder aber auch durch eine Dreiecksfunktion erfolgen. Es sollte dabei eine Frequenz von weniger als 20 Hz gewählt werden.



Übungen

1) Implementieren sie einen FIR Filter um ein einfaches Echo zu erzeugen

2) Implementieren sie einen IIR Filter um ein multiples Echo zu erzeugen.

3) Implementieren sie einen Tremelo Effekt.

4) Implementieren sie einen Phaser Effekt.

5) Implementieren sie einen Chorus Effekt.

6) Kombinieren sie ihre Lösungen mit denen aus Übungseinheit 2.

7) Implementieren sie eine Steuerung um die Filter und Effekte im laufenden Betrieb ein- und auszuschalten. Dabei sollten die DIP switches und LEDs auf dem Board verwendet werden, alternativ können auch GEL Files verwendet werden um die Verarbeitung vom Host PC aus zu steuern.

Tipps:

- Beachten sie, dass der Linker versucht, alle Variablen und Arrays im internen RAM des Prozessors abzulegen. Bei den Filtern werden jedoch zum Teil sehr große Arrays verwendet, für die das interne RAM zu klein ist. Durch folgende Präprozessordirektive kann eine Variable oder Array in einem anderen Speicherbereich abgelegt werden:

  ```
  #pragma DATA_SECTION (<Variable>, <Speicherbereich>);
  ```

  Legen sie die Arrays im SDRAM auf dem Board ab. Wie der entsprechende Speicherbereich heißt, können sie dem Linker Command File entnehmen.

- Vermeiden sie Divisionen (dazu gehören auch modulo Berechnungen), da diese vom DSP Prozessor nur sehr langsam ausgeführt werden können.

- Führen sie alle Effekte mit Lautstärkeregelung (u.a. Balance und Fader) erst nach den Berechungen der Filter und Effekte durch, um für diese Berechnungen die volle Amplitude zur Verfügung zu haben.

- Achten sie darauf, die Buffer zunächst mit Datensamples zu füllen, bevor sie mit der Bearbeitung, d.h. den Filteroperationen beginnen.

- Verwenden sie Sinustabellen für die Verzögerungswerte der Effekte Phaser, Chorus und Tremelo.