YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DSP TABANLI G/Ç KARTI ÜZERİNDE SES İŞARETİ İŞLEME UYGULAMALARI

Kontrol Bilgisayar Müh. Fatih BORAN

FBE Elektrik Mühendisliği Anabilim Dalı Kontrol ve Otomasyon Programında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı :Yrd. Doç. Dr. Şeref Naci ENGİN

Yrd. Doç. Dr. Ünal Küçük

Prof. Dr. Halil PASTACI

Yrd. Doç. Dr. Şeref Naci Engin

İSTANBUL, 2005

İÇİNDEKİLER

Sayfa

KISALTMA LİSTESİ

DSP	Digital Signal Processing ya da Processor – Dijital Sinyal İşleme ya da İşlemcisi
TDM	Time Division Multiplex – Zaman Bölmeli Çoğullama
I2S	Inter – IC Sound
SDRAM	Synchronous Dynamic Read Access Memory
EBIU	External Bus Interface Unit – Dış Taşıyıcı Arabirimi
DMA	Direct Memory Access – Doğrudan Bellek Erişimi
SPORT	Standard Port
RTC	Real Time Clock – Gerçek Zaman Saati
SPI	Serial Peripheral Interface – Seri İletişim Arabirimi
PPI	Parallel Peripheral Interface – Paralel İletişim Arabirimi
JTAG	Joint Test Action Group – Emülasyon Arabirimi
RS-232	Seri İletişim Arabirimi
PF	Peripheral Flag – Çevrebirimi Bayrağı
LDF	Linker Description File – Bağlayıcı Tanım Dosyası

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

ÖNSÖZ

Teknolojik gelişmeler, gün geçtikçe yeni tasarlanacak elektronik sistemlerin daha kolay taşınabilir, daha hızlı ve daha az güç tüketimli yapılar olmasını gerektirmektedir. Bununla birlikte gelen gelişmeler, elektronik sistemlerin günlük hayatın bir çok dalına artan bir şekilde girmesine sebep olmuştur. Özellikle gömülü sistemler olarak adlandırılan bu modeller üzerine yapılan çalışmalar yaygınlaştıkça, günlük işleri kolaylaştıran ya da çok farklı ihtiyaçlara cevap verebilen araçların geliştirilmesine de yol açılmaktadır.

Ülkemizde de bu yönde yapılacak Araştırma ve Geliştirme programlarına destek giderek artmakta ve bu sektörün ekonomik değerinin daha iyi anlaşılmakta olduğu düşünülmektedir. Donanımsal yazılım olarak da adlandırılabilecek bu sistemlerin nasıl üretilebileceği üzerine teori ve uygulamada katkı sağlanmak amacıyla yapılan bu projede değerli desteklerinden dolayı tez danışmanım Sayın Yrd. Doç. Dr. Şeref Naci Engin' e teşekkür ederim.

ÖZET

Bu çalışmada temel konu bir gerçek zamanlı dijital sinyal işleme uygulaması olan ses sistemleri için parametrik ekolayzır fonksiyonunun gerçeklenmesi olarak belirlenmiştir. Bu amaçla tasarlanacak sistemlerin bilgisayar üzerinde simülasyonu ve gerçek zamanlı gömülü sistem üzerindeki yapısı ve çalışması incelenmiştir.

Gömülü sistem üzerinde tasarlanmak üzere ikinci dereceden bikuad IIR filtre yapıları seçilmiştir. Seçilen bu filtre yapılarının teorik incelemesi yapılmış ve uygulamaya aktarılacak yazılım modelleri oluşturulmuştur. Simülasyon ile DSP kiti araçları tanıtılmış ve projede gerçeklenen yazılımın önemli noktaları üzerinde durulmuştur. Bununla birlikte sık karşılaşılan problemler ve bunların çözüm yolları irdelenmiştir.

Bu tez ile üzerinde durulan ana problem, sistem performansının geliştirilmesinde izlenecek optimizasyon yöntemleridir. Bu nedenle gerçek zamanda işaret işlemeyi sağlayacak sistem performansı ölçüm ve yazılımının geliştirilmesi üzerinde önemle durulmuştur.

Anahtar Kelimeler: Gerçek zaman dijital sinyal işleme, ses işareti uygulamaları, bikuad, IIR filtre

ABSTRACT

In this thesis, we propose a method for implementation of a parametric equalizer function on sound systems, which is a Real Time Digital Signal Processing application. In conjunction with this, we have developed a computer simulation of the system and examined the designed real time embedded system.

A second-order biquadratic filter type has been chosen for designing the embedded system. Theoretical study of the filter type has been focused and software models have been constituted which would have transferred to the application. The simulation and DSP tools which have been used in this project are introduced and then important points of developed software are pointed out. Moreover, we have studied carefully the most commonly encountered problems and suggested the ways of solving them.

The main problem in this work is developing optimization methods which will be used for the improvement of the performance of the system. For this reason we presented some points related to measurement of the system performance and optimization of the system software.

Keywords: Real-time digital signal processing, audio signal applications, biquad, IIR, filter

1 GİRİŞ

Sayısal işaret işleme konuları bilgisayar teknolojisinin hızla gelişmeye başladığı 1970'li yıllardan günümüze kadar yoğun bir şekilde gelişmiş ve temel teorileri üzerine geliştirilen yöntemler günümüzde oldukça yüksek karmaşıklılık, çeşitlilik ve hıza ulaşmıştır. Özellikle yazılımların hızlı ve güvenilir bir şekilde çalışma ihtiyacı donanım gereklerini ve mimarisini üst noktalara çekmeye zorlamıştır. Günümüz yükselen değerleri olan hız, güvenirlilik, maliyet düşüklüğü, fiziksel boyutlar ve yazılımsal ihtiyaçlar gibi konular geliştirilen donanım tasarımlarının en belirleyici kıstasları haline gelmiştir.

Halen kullanılan işaret işleme teorilerinin birçoğu analog dünyada kullanılmak amacıyla geliştirilmiştir. Bu nedenle dijital dünyaya geçişte karşılaşılan problemlerin çözümleri ve bu problemlerin donanıma yüklenmeden önce nasıl simüle edilerek giderilebileceği üzerinde önemle durulacaktır.

Bu tezde günümüz dijital dünyasının bazı uygulamaları üzerinde durulmuştur. Her ne kadar işaret işleme konuları ile gerçekleştirilebilecek projeler çok çeşitli gözükse de temelde işaret işleme teorilerinin özelleştirilmiş, karmaşıklığı ve hızı artırılmış biçimleridir. Bu nedenle DSP üzerinde gerçekleştirilecek herhangi bir uygulama diğer bir çok uygulama için de temel bilgi kaynağı olacağı düşünüldüğünden ele alınan gerçek zamanlı filtreleme uygulamasının sentezi ve analizinin bir çok farklı projeye ışık tutması beklenmektedir.

2 İKİNCİ DERECEDEN ANALOG IIR FİLTRELERİN ANALİZİ

2. dereceden analog filtreler elektronik tasarımda yoğun olarak kullanılan yapılardır. Bunun nedeni düşük maliyette evrensel bir filtre yapısı sağlayabilmesidir. Bununla birlikte eleman katsayılarının değişmesine karşı oldukça hassas yapıları olması gibi bir dezavantajı vardır. Özellikle elektronik elemanların yeterli miktarda bu hassasiyeti sağlayamaması ve zamanla sahip olduğu katsayıların değişebilmesi açısından önemlidir. Bu nedenle IIR filtrelerin sayısal olarak gerçeklenmesi daha güvenilir ve kolaydır.

Bu tür tasarımlarda genellikle ω_0 - merkez frekansı, Q – kalite faktörü belirtilir; "ω_0 'ı 10 kHz ve Q 'su 5 olan band geçiren filtre tasarlayınız" gibi...

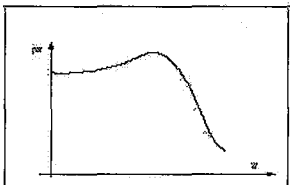
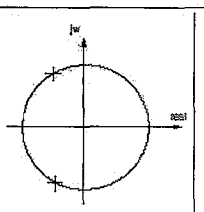
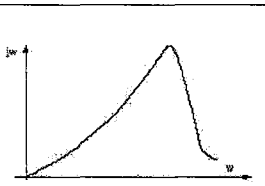
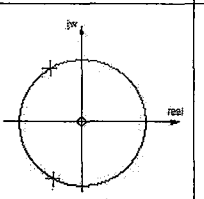
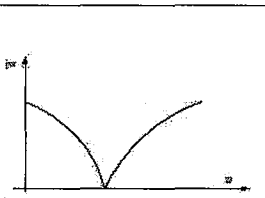
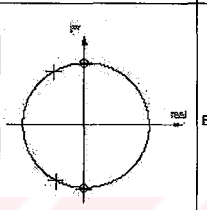
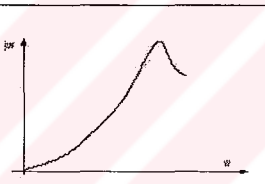
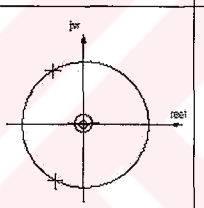
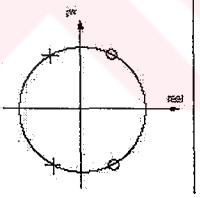
Q değeri ile band genişliği(BW) arasında çeşitli bağıntılar kurulabilmektedir. Bu nedenle tasarım belirli bir kazançtaki BW değeri verilerek de yapılabilmektedir. Q değeri ile BW ters orantılıdır.

Aktif filtrelemede çok kullanılan yapılardan biri de *bikuad* olarak adlandırılan sistemlerdir. Bu yapı 2.dereceden IIR sistemlerin özel halleri olup BW veya Q'ya dayalı filtrelerde hesaplama kolaylığı sağladığı için kullanılır. Genel transfer fonksiyonu:

$$T(s) = \frac{k_1.s^2 + k_2.(\frac{\omega_0}{Q}).s + k_3.{\omega_0}^2}{s^2 + \frac{\omega_0}{Q}s + {\omega_0}^2} \qquad (2.1)$$

şeklindedir. Bu yapılar parametrik ekolayzır sistemlerinin tasarlanmasında en çok kullanılan sistemlerdir.

Bazı 2.dereceden IIR filtre yapılarının frekans cevapları , kutup - sıfır yerleşimi Çizelge 1.1' de verilmiştir.

	Frekans Cevabı	Kutup/Sıfır	Türü
$T_{LP} = \dfrac{\omega_0^{\,2}}{s^2 + \dfrac{\omega_0}{Q}s + \omega_0^{\,2}}$			Alçak Geçiren
$T_{BP} = \dfrac{\dfrac{\omega_0}{Q}s}{s^2 + \dfrac{\omega_0}{Q}s + \omega_0^{\,2}}$			Band Geçiren
$T_{BS} = \dfrac{s^2 + \omega_0^{\,2}}{s^2 + \dfrac{\omega_0}{Q}s + \omega_0^{\,2}}$			Band Durduran(Çentik)
$T_{HP} = \dfrac{s^2}{s^2 + \dfrac{\omega_0}{Q}s + \omega_0^{\,2}}$			Yüksek Geçiren
$T_{AP} = \dfrac{s^2 - \dfrac{\omega_0}{Q}s + \omega_0^{\,2}}{s^2 + \dfrac{\omega_0}{Q}s + \omega_0^{\,2}}$			Tüm Geçiren

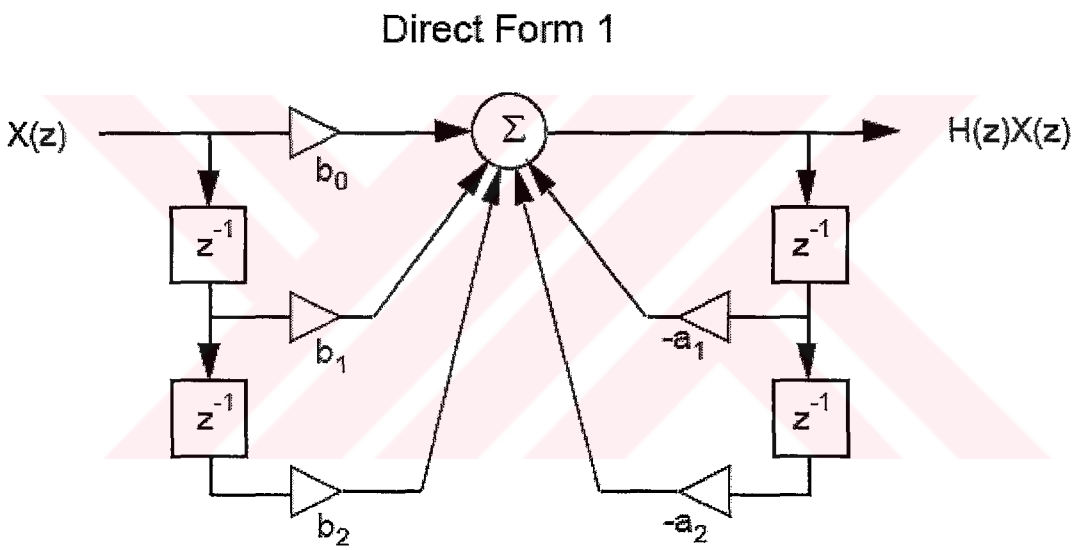
Çizelge 2.1 İkinci dereceden bazı sistemlerin frekans cevapları

2.1 İkinci Dereceden Parametrik IIR Filtrelerin Katsayılarının Bulunması

Genel olarak 2.dereceden dijital rekürsif filtreler(bir başka deyişle IIR filtreler) aşağıda verilen transfer fonksiyonunu yapısındadır.

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \tag{2.2}$$

Bu transfer fonksiyonunu gerçekleyen blok şeması, doğrudan programlamayla (direct programming, direct form 1) aşağıda verildiği gibidir.



Şekil 2.1 Direkt form - I grafı

Verilen transfer fonksiyonun 5 bilinmeyeni bulunmaktadır. Parametrik bir band geçiren filtrenin tanım denklemleri aşağıdaki gibi verilmiştir. Bu fonksiyon ile elde edilebilecek band geçiren filtre modelleri Şekil 2-5 'de verilmiştir.

Aşağıda verilen 4 tanım denkleminden bilinmeyen 5 katsayıyı bulabilmek için BW ya da Q olarak tanımlanan bant genişliği ya da kalite faktörünün verilmesi yeterlidir.

Başlangıç ve bitiş frekansları 1 olmalıdır.

$$H(e^{j0}) = H(1) = 1$$
$$H(e^{j\pi}) = H(-1) = 1 \tag{2.3}$$

Tepe noktasındaki 1. ve 2. türevleri 0 olmalıdır.

$$\frac{\partial}{\partial\Omega}\left|H(e^{j\Omega})\right|_{\Omega=\Omega_0} = 0 \tag{2.4}$$

$$\frac{\partial}{\partial\Omega}\left|H(e^{j\Omega})\right|^2_{\Omega=\Omega_0} = 0. \tag{2.5}$$

Belirli bir noktadaki kazanç

$$\left|H(e^{j\Omega_0})\right| = 10^{G/20\text{dB}} \equiv K \tag{2.6}$$

Bu fonksiyonlar ile verilecek BW değerleri ile ilgili değişik tanımlamalar mevcuttur. Bunlardan bazıları:

- Tepe noktasının -3 dB altındaki
- G/2 noktasındaki

BW'nin G/2 noktasındaki kazancı için verilen tanımlama hesap kolaylığı sağlamaktadır. Bu nedenle bu tanımlama kullanılacaktır.

2.1 IIR Transfer Fonksiyonunun Belirlenmesi

Katsayıların belirlenebilmesi için analog filtre modelinden yararlanılarak dijital modele bilineer dönüşüm(Tustin dönüşümü) kullanılarak geçilir. Bununla beraber bilineer dönüşümden kaynaklanan eğrilmenin giderilmesi için yapılan düzenlemede yeni bir BW değeri kullanılır. Aşağıda kullanılacak model fonksiyonu *s*-domeninde verilmiştir.

$$\hat{H}(s) = \frac{s^2 + 2K\alpha\omega_0 s + \omega_0^2}{s^2 + 2\alpha\omega_0 s + \omega_0^2} \tag{2.7}$$

Bu model ile ilgili bilinen analog filtre modeli fonksiyonları aşağıda verilmiştir.

$$\hat{H}(j0) = \hat{H}(j\infty) = 1, \quad \hat{H}(j\omega_0) = K, \tag{2.8}$$

$$\frac{\partial}{\partial\omega}\left|\hat{H}(j\omega)\right|^2_{\omega=\omega_0} = 0. \tag{2.9}$$

G/2 noktasındaki kazanç değeri aşağıda verildiği gibi tanımlanmıştır.

$$\left|\hat{H}(j\omega)\right| = 10^{\frac{G/2}{20\text{dB}}} = \sqrt{K} \tag{2.10}$$

ya da

$$\left|\hat{H}(j\omega)\right|^2 = K \tag{2.11}$$

Bu denklemler (2.8)-(2.9) 'da kullanılırsa aşağıdaki açısal frekans denklemi elde edilir.

$$\omega^2 = \omega_0^2\left[1 + 2K\alpha^2 \pm 2\alpha\sqrt{K(K\alpha^2 + 1)}\right]. \tag{2.12}$$

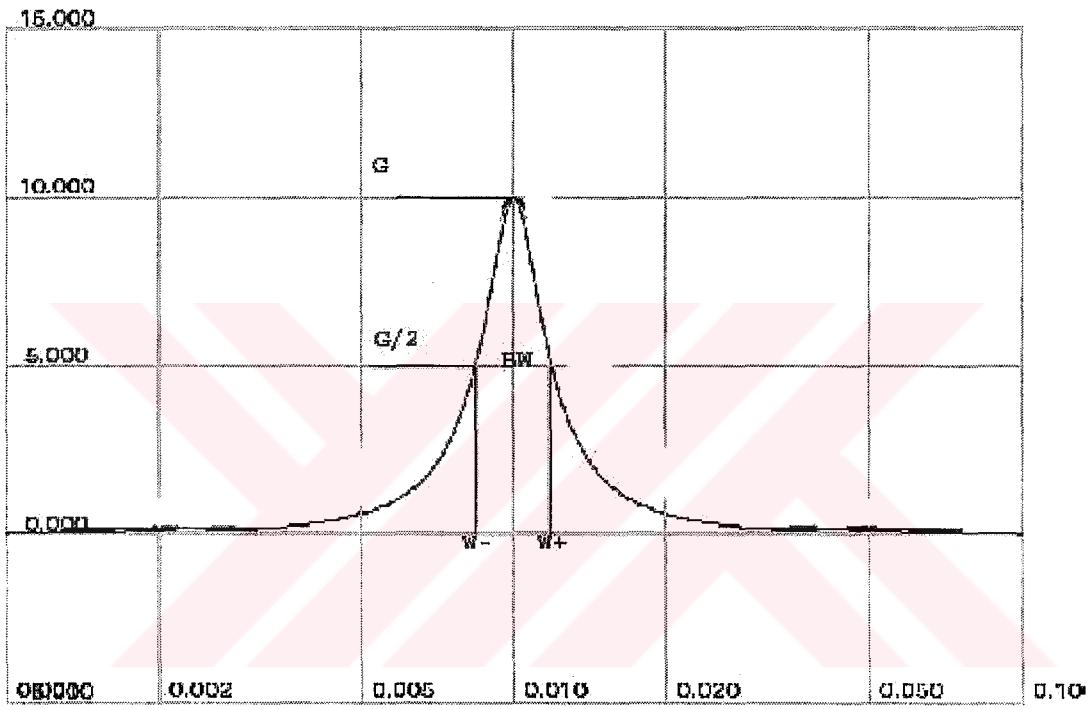
Buradan da band genişliği için tanımlanmış olan alt ve üst band için açısal frekans

Üst Band

$$\omega_+^2 = \omega_0^2\left[1 + 2K\alpha^2 + 2\alpha\sqrt{K\left(K\alpha^2 + 1\right)}\right] \tag{2.13}$$

Alt Band

$$\omega_-^2 = \omega_0^2\left[1 + 2K\alpha^2 - 2\alpha\sqrt{K\left(K\alpha^2 + 1\right)}\right]. \tag{2.14}$$



Şekil 2.2 BW'si G/2 de tanımlanmış parametrik band geçiren filtre

Alt band üst band cinsinden şu şekilde ifade edilebilir.

$$\omega_+ = \omega_- 2^{bw} = \omega_- e^{\beta} \qquad \beta \equiv \ln(2)bw. \tag{2.15}$$

$$\omega_+^2 e^{-\beta} = \omega_-^2 e^{\beta} \tag{2.16}$$

$$\alpha^2 = \frac{1}{K}\left[-1 \pm \sqrt{1 + \tfrac{1}{4}\left(e^{\beta} - e^{-\beta}\right)^2}\right] \tag{2.17}$$

α^2 değeri pozitif olacağından

$$\alpha = \frac{1}{\sqrt{K}} \sinh\left(\frac{\beta}{2}\right) = \frac{1}{\sqrt{K}} \sinh\left(\frac{\ln(2)}{2} bw\right) \tag{2.18}$$

Elde edilen analog filtre transfer fonksiyonu aşağıdaki verilmiştir.

$$\hat{H}(s) = \frac{s^2 + 2\sqrt{K}\sinh\left(\frac{\ln(2)}{2} bw\right)\omega_0 s + \omega_0^2}{s^2 + \frac{2}{\sqrt{K}}\sinh\left(\frac{\ln(2)}{2} bw\right)\omega_0 s + \omega_0^2}. \tag{2.19}$$

Bu noktadan sonra bilineer dönüşüm uygulanabilir.

$$s \leftarrow \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}} \quad \Rightarrow \quad H(z) = \hat{H}(s)\Big|_{s=\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}} \tag{2.20}$$

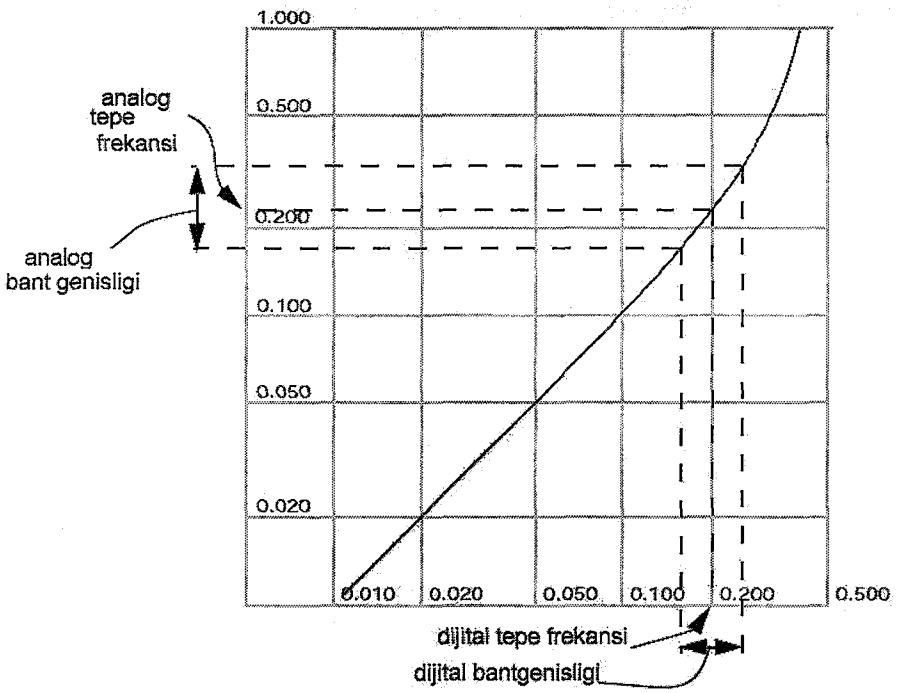
$$H(z) = \frac{\left(1+\left(\omega_0 T/2\right)^2 + 2K\alpha\left(\omega_0 T/2\right)\right) - 2\left(1-\left(\omega_0 T/2\right)^2\right)z^{-1} + \left(1+\left(\omega_0 T/2\right)^2 - 2K\alpha\left(\omega_0 T/2\right)\right)z^{-2}}{\left(1+\left(\omega_0 T/2\right)^2 + 2\alpha\left(\omega_0 T/2\right)\right) - 2\left(1-\left(\omega_0 T/2\right)^2\right)z^{-1} + \left(1+\left(\omega_0 T/2\right)^2 - 2\alpha\left(\omega_0 T/2\right)\right)z^{-2}}. \tag{2.21}$$

Yukarıda elde edilen H(z) transfer fonksiyonunda bilineer dönüşümden kaynaklanan eğrilmeler olmaktadır.

$$s = j\omega = \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}} = \frac{2}{T}\frac{z-1}{z+1} = \frac{2}{T}\frac{e^{j\Omega}-1}{e^{j\Omega}+1} = j\frac{2}{T}\tan\left(\frac{\Omega}{2}\right) \tag{2.22}$$

olmasından dolayı frekansta bir kayma meydana gelir.

Ω_0 frekansı $2\arctan\left(\frac{\omega_0 T}{2}\right)$ değerine eşit olur. Bu kayma aşağıdaki grafikte gösterilmiştir.



Şekil 2.3 Bilineer dönüşüm nedeniyle gerçekleşen kayma

Oluşan bu kaymayı engellemek için band genişliğinin ön kompanzasyonu yöntemi kullanılır. Bu yöntemde log. analog açısal frekansın log. dijital açısal frekansa göre kısmi türevini alınarak bulunur.

$$\omega = \frac{2}{T}\tan\left(\frac{\Omega}{2}\right) \tag{2.23}$$

Her iki tarafın logaritması alınıp

$$\ln(\omega) = \ln\left(\frac{2}{T}\right) + \ln\left(\tan\left(\frac{e^{\ln(\Omega)}}{2}\right)\right) \tag{2.24}$$

Kısmi türevi elde edilirse

$$\frac{\partial(\ln(\omega))}{\partial\ln(\Omega)} = \frac{\Omega}{\sin(\Omega)} \tag{2.25}$$

Burada elde edilen değer yeni band genişliği değeridir.

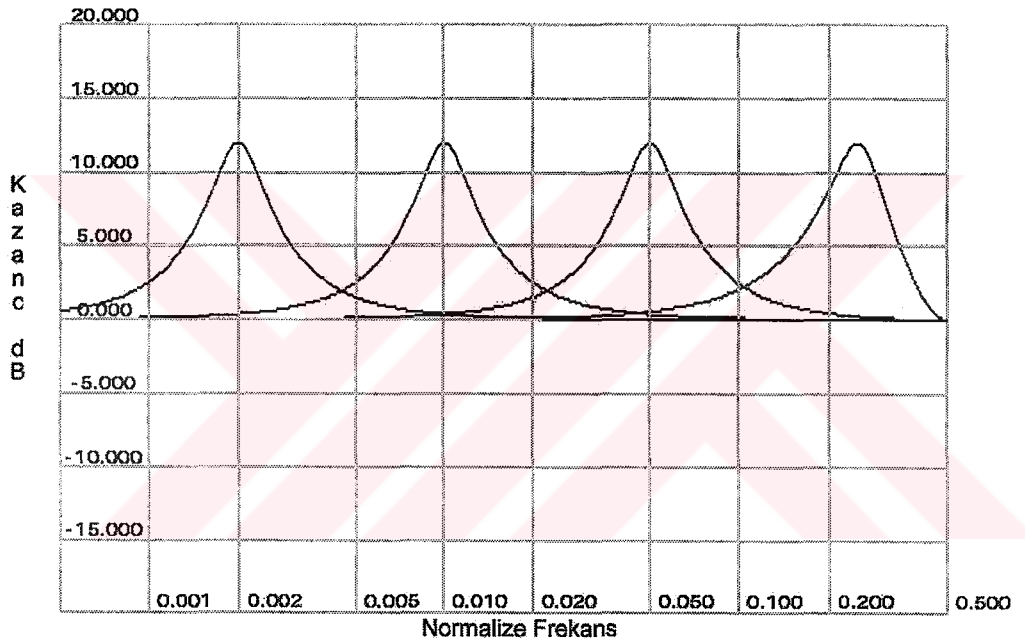
$$bw \leftarrow \frac{\Omega_0}{\sin(\Omega_0)}\, bw \tag{2.26}$$

Sonuçta elde edilen nihai transfer fonksiyonu aşağıda verilmiştir.

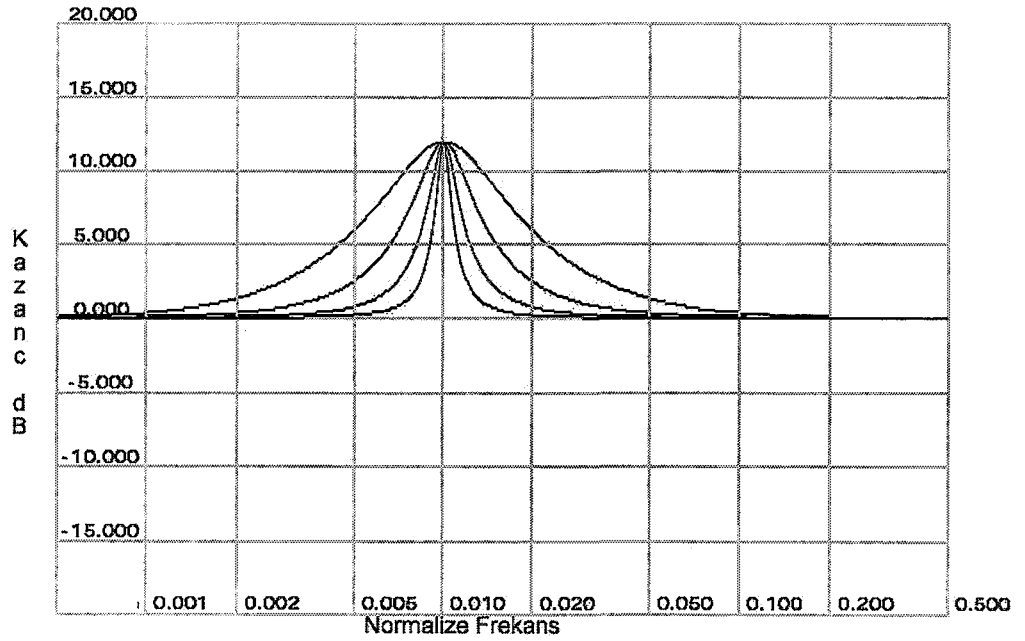
$$H(z) = \frac{\left(1+\gamma\sqrt{K}\right) - 2\cos(\Omega_0)z^{-1} + \left(1-\gamma\sqrt{K}\right)z^{-2}}{\left(1+\gamma/\sqrt{K}\right) - 2\cos(\Omega_0)z^{-1} + \left(1-\gamma/\sqrt{K}\right)z^{-2}} \tag{2.27}$$

$$\gamma = \sinh\left(\frac{\ln(2)}{2} bw \frac{\Omega_0}{\sin(\Omega_0)}\right)\sin(\Omega_0) \tag{2.28}$$

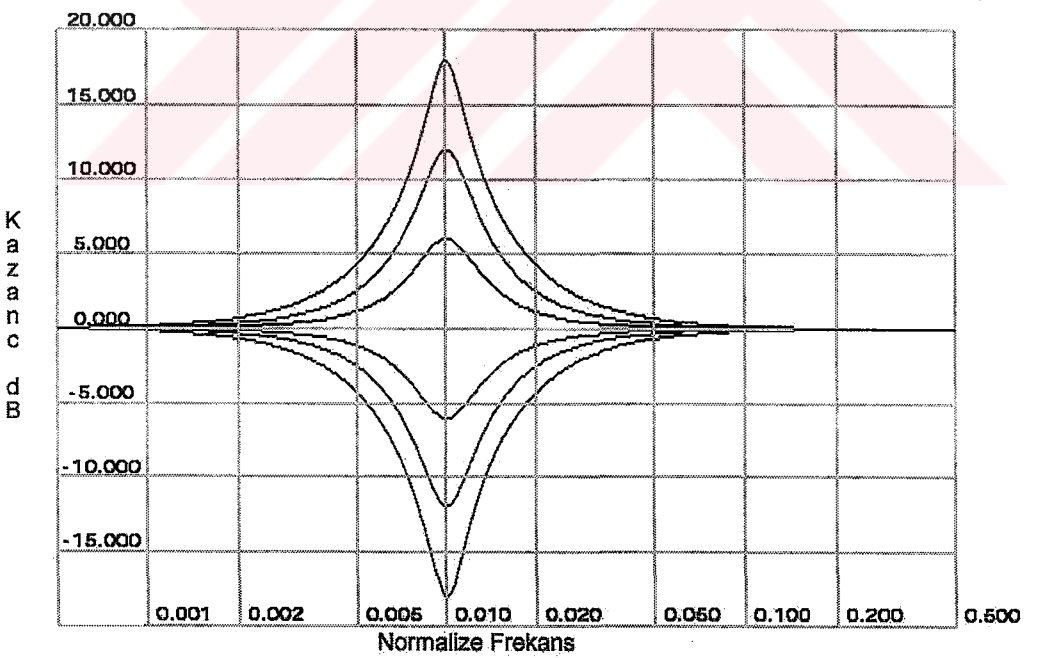
Aşağıda çeşitli durumlar için frekans – kazanç grafikleri verilmiştir.



Şekil 2.4 12 dB kazanç , 1 oktav bant genişliği , değişken tepe frekansı



Şekil 2.5 12 dB kazanç, değişken band genişliği, 0.01 tepe frekansı



Şekil 2.6 Değişken kazanç , 1 oktav band genişliği , 0.01 tepe frekansı

2.3 Filtre Katsayılarının Belirlenmesi

(2.27) 'nin tekrar düzenlenmesi ile payda fonksiyonunun ilk katsayısı 1 olacak şekilde aşağıdaki katsayılar elde edilebilir.

$$b_0 = \frac{1+\gamma\sqrt{K}}{1+\gamma/\sqrt{K}} \tag{2.29}$$

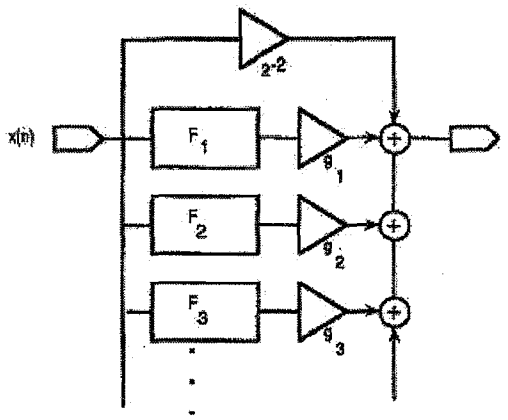
$$b_1 = a_1 = \frac{-2\cos(\Omega_0)}{1+\gamma/\sqrt{K}} \tag{2.30}$$

$$b_2 = \frac{1-\gamma\sqrt{K}}{1+\gamma/\sqrt{K}} \tag{2.31}$$

$$a_2 = \frac{1-\gamma/\sqrt{K}}{1+\gamma/\sqrt{K}} \tag{2.32}$$

3 İKİ BAND PARAMETRİK EKOLAYZIR MATLAB SİMÜLASYONU

Bir IIR parametrik band geçiren filtre akış şeması aşağıdaki şekilde verilmiştir. Burada ileri yönde sabit 0.25 kazanç değeri vardır. Diğer kazanç değerleri ise band geçiren filtrelere aittir.



Şekil 3.1 IIR parametrik bant geçiren filtreme akış şeması

Band geçiren filtrelere ait MATLAB fonksiyonu aşağıda verilmiştir.

```
function [b] = peq(dbGain,freq)
close all
format('long','g');
srate = 44100;
bandwidth = 0.3;
M_PI = 3.14159265358979323846;
M_LN2 = 0.69314718055994530942;
   A = 10^(dbGain / 40);
   omega = 2 * M_PI * freq /srate;      %radyan/sample
   sn = sin(omega);
   cs = cos(omega);
   alpha = sn * sinh((M_LN2 /2) * bandwidth * (omega /sn));
%kalite faktoru
%Q = 1/(2*sinh(M_LN2/2*bandwidth*omega/sn));
b0 = 1 + (alpha * A);
b1 = -2 * cs;
b2 = 1 - (alpha * A);
```

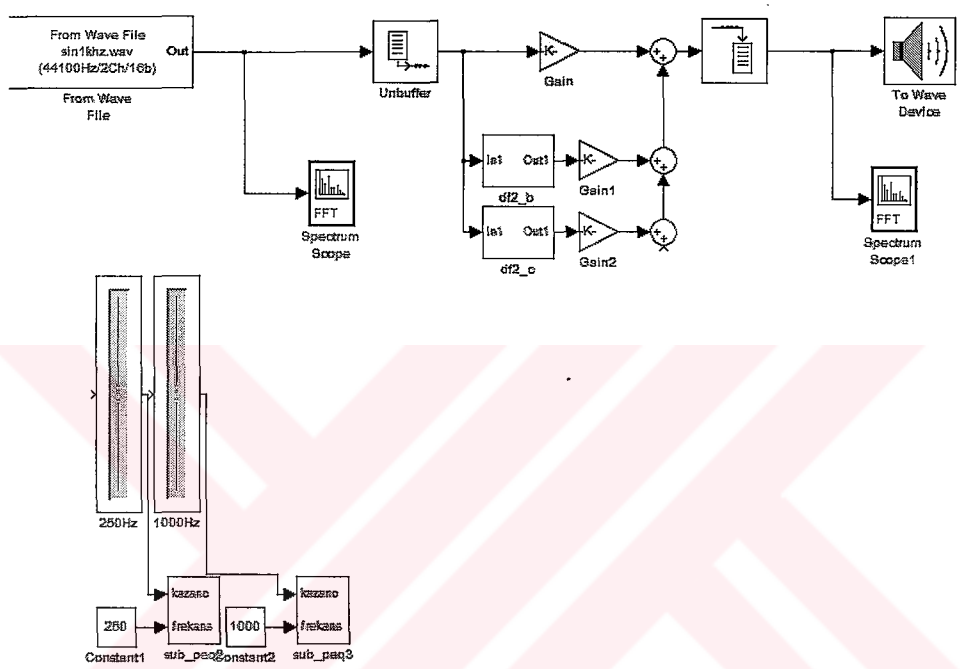
a0 = 1 + (alpha /A);

a1 = -2 * cs;

a2 = 1 - (alpha /A);

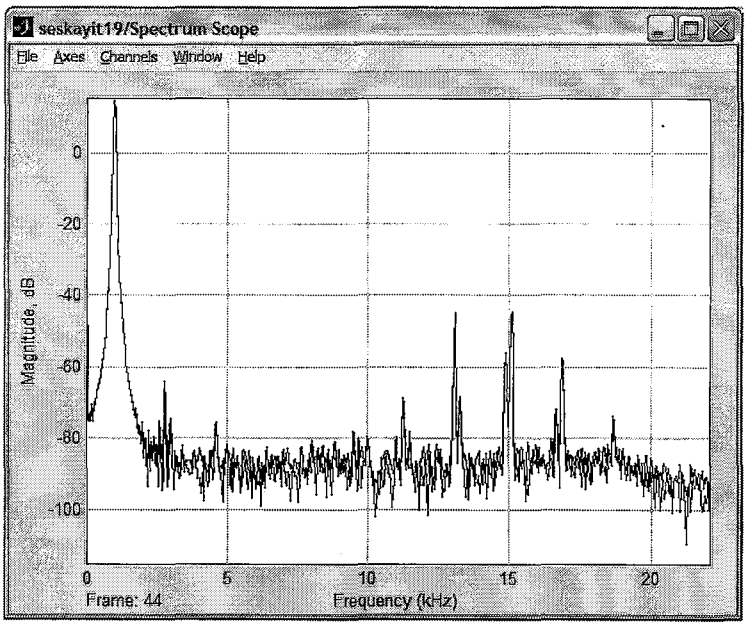
b = [b0, b1, b2, a0, a1, a2];

Aşağıdaki şekilde de verilen simülasyonda 1kHz'lik 44100 Hz örnekleme frekansına ve 16 bit kuantalama aralığına sahip bir sinüs işareti içeren WAV dosyası kullanılmıştır.

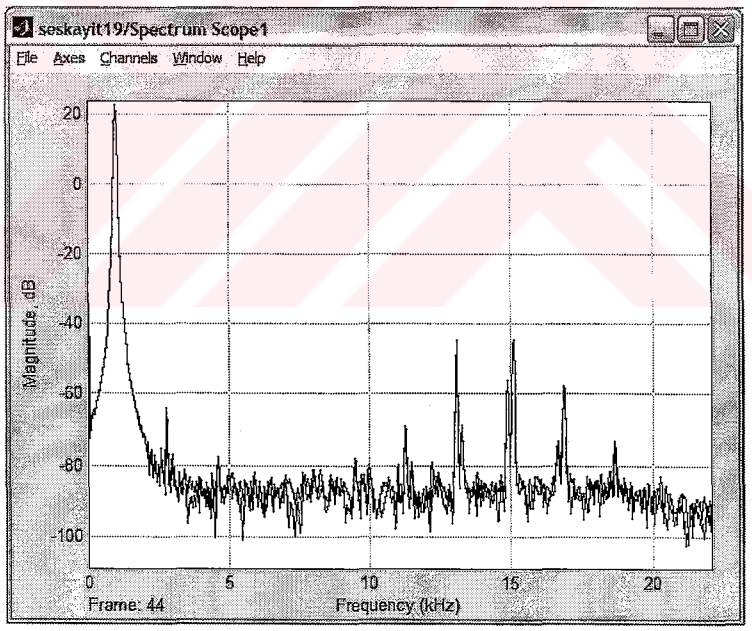


Şekil 3.2 İki bant parametrik ekolayzır simulink simülasyonu

1000 Hz'lik filtrenin kazanç değeri +14 dB'ye getirilince Şekil 10' daki frekans spektrumu elde edilmiştir. +14 dB'ye getirilmesi ise spektrumu etkilememektedir, çünkü 250 Hz filtrenin band genişliği 1000 Hz'i kapsamamaktadır.



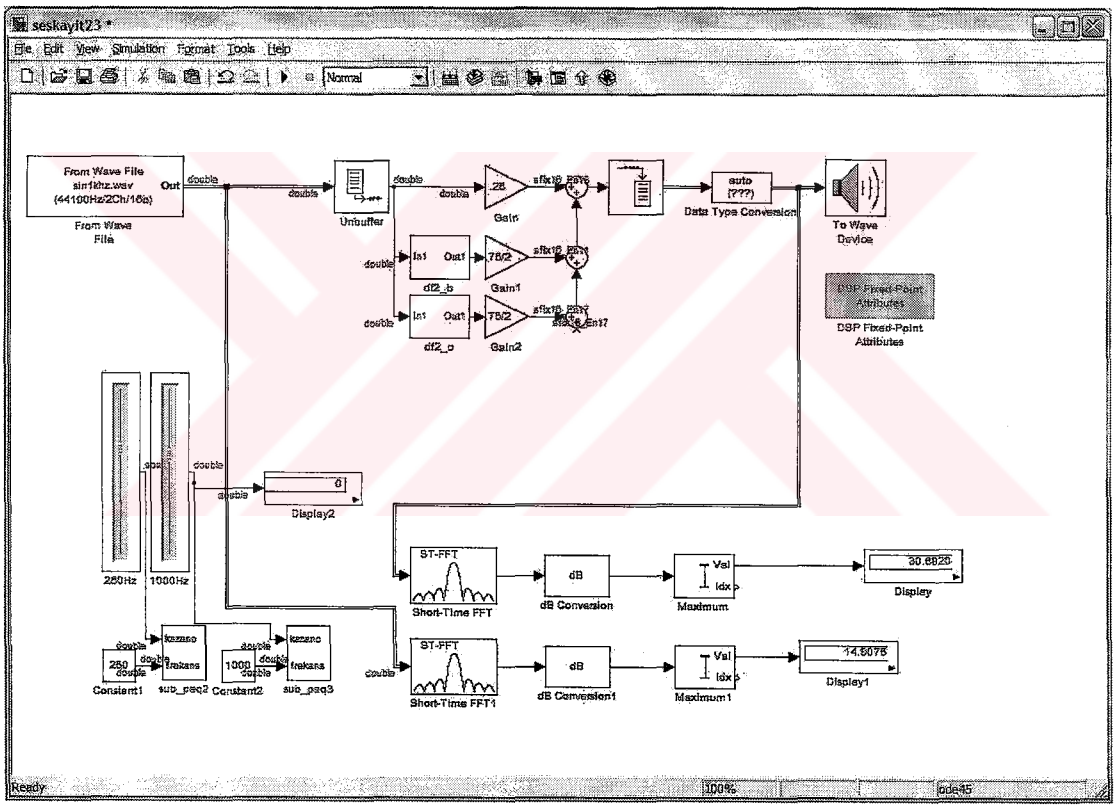
Şekil 3.3 Giriş işareti spektrumu ~16dB maks. – 1kHz



Şekil 3.4 Çıkış işareti spektrumu ~22dB maks. – 1kHz

3.1 Sabit Noktalı Simülasyon

MATLAB ve Simulink Yazılımları ile sabit noktalı işlemcilere yönelik simülasyonlar yapılabilmektedir. Bunun için öncelikle sistem kayan sayı modeline göre tasarlanır. Daha sonra matematiksel işlem blokları *Fixed-Point Toolbox* olarak adlandırılan araç kiti altındaki modeller ile değiştirilir. Bununla beraber sistemde oluşabilecek sabit sayı pozitif ve negatif taşması ve doyma gibi bu sisteme has hata durumları simüle edilebilmektedir. Şekil 3.5' de 2 Band Ekolayzır sisteminin Sabit sayı sistemine göre nasıl simüle edildiği görülmektedir. Ayrıca *Real Time Toolbox* olarak adlandırılan araç kiti ile de standart C kodu ya da çeşitli platformlara özgü kod üretilebilmektedir.



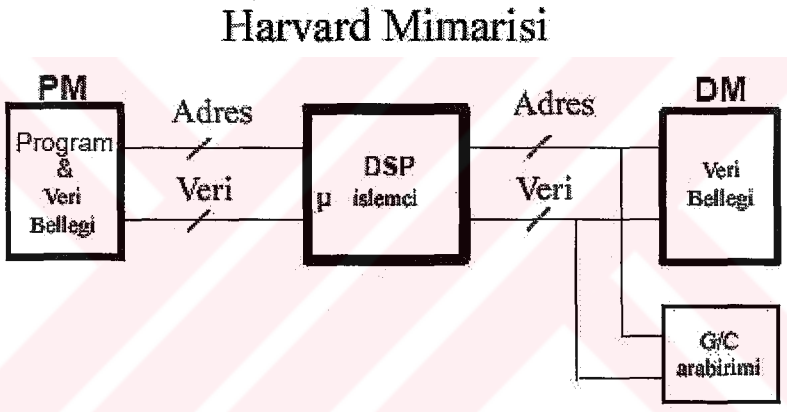
Şekil 3.5 Simulink üzerinde sabit noktalı simülasyon

4 DSP ARABİRİMİNİN İNCELENMESİ ve ÖZELLİKLERİ

DSP donanım yapısı bazı özellikleri ile standart bilgisayar sistemlerinden ayrılmaktadır. Bunların başlıcaları:

1. MAC: Çarpma – Toplama Birimi → 1 işlemci devrinde çarpma ile toplama işlemini ard arda yapabilen birim.

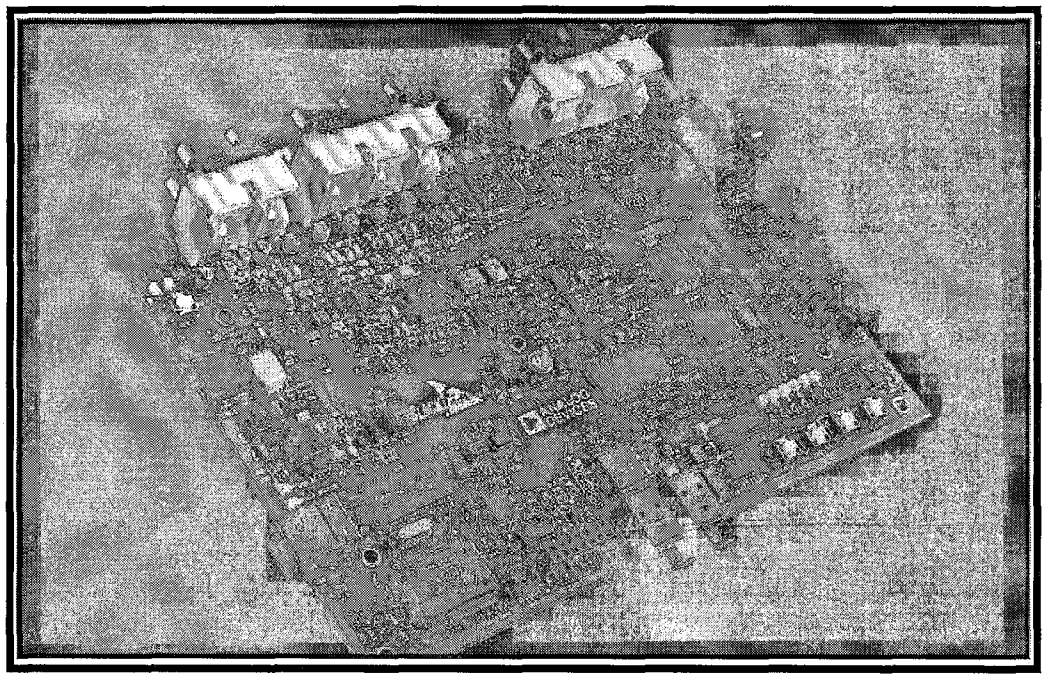
2. Harvard Mimarisi: Program belleği ile veri belleğinin ayrı olduğu mimari yapısına denir. Aynı işlemci devrinde hem program kodu işlenir hem de veri belleğinden veri çekilebilir.



Şekil 4.1 DSP mimarisi

3. Donanımsal dairesel tampon desteği
4. Donanımsal döngü optimizasyonları
5. Blok veri işleyebilme kapasitesi
6. Temel işaret sentezleyici tablosu oluşturma(Sinüs,Kare Dalga,Üçgen Dalga … gibi)

Projede kullanılan donanım sistemi Şekil 4.2'de verilmiştir.

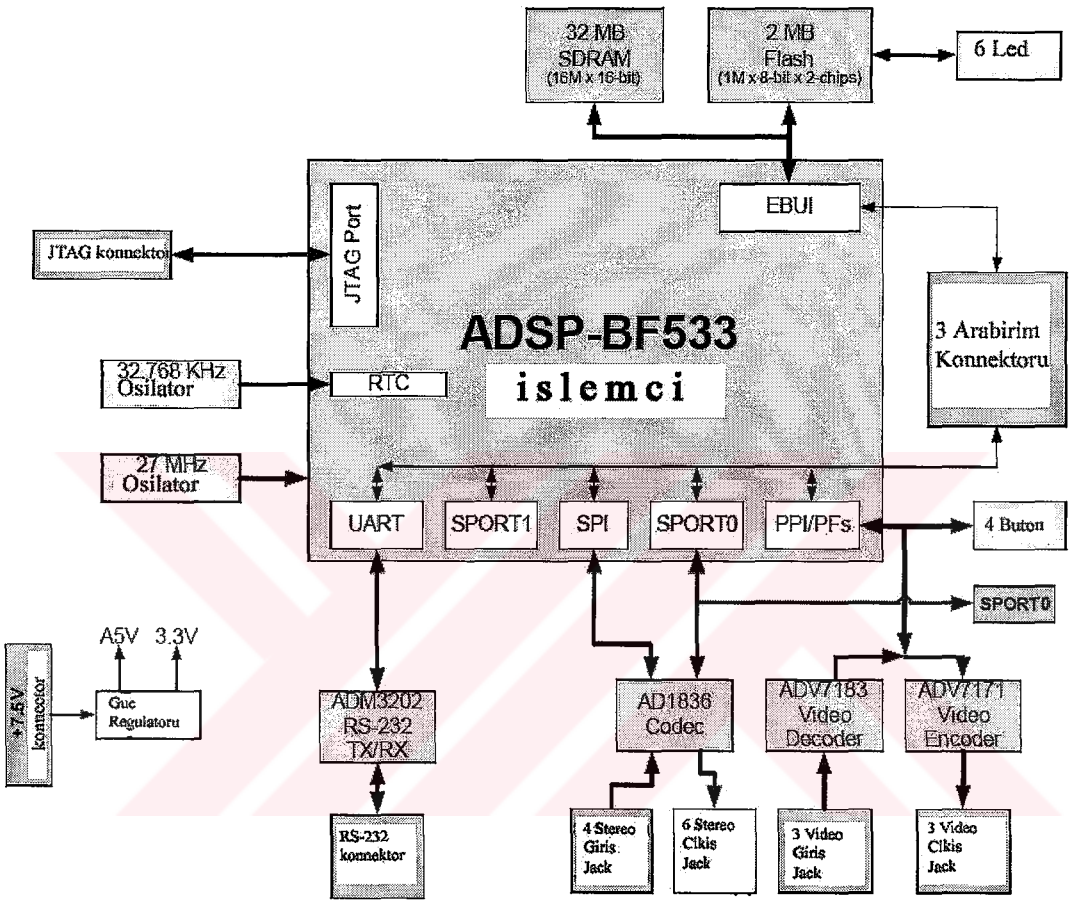


Şekil 4.2 ADSP – BF533 EZKIT donanım kiti

Çok yönlü kullanıma sahip bu setin özellikleri aşağıda verilmiştir :

- ADSP-BF533 Blackfin DSP işlemcisi Model - **ADSP-BF533SKBC750 – 750 MHz**
- 32 MB (16M x 16-bit) SDRAM
- 2 MB (512K x 16-bit x 2) FLASH bellek
- AD1836 96 kHz ses kodek, 4 giriş, 6 çıkış. RCA jak
- ADV7183 video kod çözücü 3 giriş, RCA jak
- ADV7171 video kodlayıcı 3 çıkış, RCA jak
- VisualDSP++ yazılım seti

Bu kit ses ve görüntü işleme özelliklerini barındırmaktadır. Fakat bu projede video işleme özellikleri kullanılmamıştır.



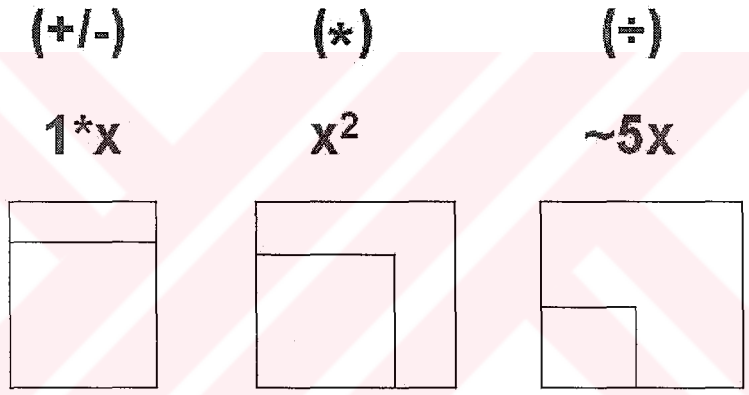
Şekil 4.3 DSP kiti donanım diyagramı

4.1 Nokta İşlemine Göre DSP Türleri

DSP işlemcileri günümüzde 2 ana matematiksel grup altında toplanmıştır. Bunlar

1. Kayan Noktalı İşlem
2. Sabit Noktalı İşlem

Hesaplama hassasiyeti ve kolaylığı açısından kayan noktalı işlemciler her zaman sabit noktalı işlemcilere göre daha üstündür. Fakat hız, maliyet düşüklüğü, güç tüketimi ve donanım boyutları açısından sabit noktalı işlemciler daha üstün olduğu için bir çok geliştirici tarafından hala geliştirilmekte ve destek verilmektedir. Kayan nokta işlemcilerinin sabit nokta işlemcilere göre donanım boyutlarındaki artışı ile ilgili olarak aşağıdaki şekil verilebilir.



Şekil 4.4 Sabit noktalı işlemcilerin kayan noktalı işlemcilere göre donanım büyüklüğü karşılaştırması

4.2 DSP Yazılım Setinin İncelenmesi ve Özellikleri

DSP işlemcisi VisualDSP++ arabirimi kullanılarak programlanabilmektedir. Bu arabirim ile Analog Devices firmasına ait Assembly dili ile beraber, standart C , C++ dilleri ile de programlama yapılabilmektedir. Projede kullanılan yazılım dili diğer platformlara kolay aktarılabilmesi amacıyla ANSI C ile gerçeklenmiştir. Fakat performans açısından gerekli görüldüğü durumlarda zaman zaman Assembly dili kullanımı ihtiyaç haline gelmektedir.

Şekil 4.5 Visual DSP++ arabirimi

4.2.1 DSP Yazılım setinin üzerinde ekolayzır sisteminin gerçeklenmesi

DSP kiti üzerinde herhangi bir yazılımın koşturulabilmesi için öncelikli olarak kart üzerindeki arabirimlerin başlangıç durumlarına getirilmesi gerekmektedir. Bu nedenle kart üzerindeki donanımların genel çalışma durumlarını gözden geçirmek gerekmektedir.

Şekil 4.3'de kart ile ilgili donanım diyagramı verilmiştir. Ana program rutini içerisinde aşağıdaki fonksiyonlar çağrılmaktadır.

```
Init_Flags();
Init_EBIU();
Init_Flash();
Init1836();
Init_Sport0();
Init_DMA();
Init_Interrupts();
Init_SDRAM();
Enable_DMA_Sport0();
```

Bu fonksiyonlar şu amaçlarla kullanılmaktadır.

- Init_Flags fonksiyonu işlemci üzerinde bulunan programlanabilir pinlerin –ki bu pinler kart üzerinde bulunan butonlara bağlıdır, başlangıç durumlarını(aktif, pasif, yön) ve kesme şartlarını ayarlar.

- Init_EBIU fonksiyonu EBIU olarak adlandırılan arabirim üzerindeki Flash A, ya da Flash B asenkron bellek banklarına erişebilmek için kullanılır.

Flash A üzerindeki bazı pinler *port* olarak kullanılabilmektedir. Bu portlar diğer donanımların bazı kontrol pinlerine bağlı olup Flash A üzerinden kontrol edilebilir. AD1836 kodek reset pini de bunlardan biridir.

- Init_Flash fonksiyonu Flash A üzerindeki Port A.1' i çıkış pini olarak ayarlar.

- Init1836 fonksiyonu AD1836 codec ini SPI portunu ayarlamak suretiyle başlangıç durumuna getirilir.

AD1836 kodeği 2 farklı modda çalışabilir. Bunlardan ilki I2S modu olup Phillips firması tarafından standartlaşmış bir veri iletişim standardıdır. Diğeri ise TDM modu olup haberleşme teorisinde standart olarak kullanılan metotlardan birisidir.

- Init_Sport0 fonksiyonu AD1836 codec inin Sport0 portunu I2S moduna getirir.

- Init_DMA fonksiyonu Sport0 üzerinden doğrudan bellek erişimine izin vererek ses örneklerini otomatik tampona almayı sağlar. DMA1 otomatik alma, DMA2 otomatik gönderme amacıyla kullanılır.

- Init_Interrupts fonksiyonu genel olarak sisteme eklenecek olan donanım kesmelerini ve kesme rutinlerini bağlamak amacıyla kullanılmıştır. Bu örnekte AD1836 ya bağlı olan Sport0 ile LED' lere bağlı olan FlagA' nın kesme öncelik sırasını belirlemek ve kesme rutini bağlamak amacıyla kullanılır.

- Init_SDRAM fonksiyonu kart üzerinde bulunan 32MB belleği aktif duruma getirir.

- Enable_DMA_Sport fonksiyonu Sport0 üzerinden ses örneklerini alma ve gönderme işlemlerini başlatır.

Herhangi bir andaki kesmeye ne şekilde yanıt verileceği kesme rutinleri ile belirlenir. Bu rutinler ANSI C dilinde programa INTERRUPT_HANDLER(.) makrosu ile bağlanır. Ses işaretlerinin alma – gönderme işlemi Sport0_RX_ISR işareti altında temsil edilir. Bu makro altında her bir ses işareti örneğinin işleneceği Process_Data() fonksiyonu çağrılmaktadır. Bu bölümde kullanılacak işlem bikuad IIR filtreleme olduğu için Process_Data fonksiyonu altında biquad.c ve biquadCall.c deki fonksiyonlar kullanılmaktadır.

DSP ile yapılacak işlemler gerçek zamanlı olacağından dolayı her bir işaretin işlenme süresi iki örnekleme arasındaki süre ve işlemcinin hızı ile doğrudan ilişkilidir.

BiQuad_new fonksiyonu ile girilen parametreye göre 7 değişik 2.derecede IIR filtre tanımlanabilmektedir. Bunlar:

LPF - Alçak Geçiren
HPF – Yüksek Geçiren
BPF – Band Geçiren

NOTCH – Çentik

PEQ – Parametrik Ekolayzır

LSH – Low Shelf

HSH – High Shelf

Katsayı hesaplamaları aşağıdaki şekilde yapılmaktadır.

```
.
.
switch (type) {
   case LPF:
      b0 = (1 - cs) /2;
      b1 = 1 - cs;
      b2 = (1 - cs) /2;
      a0 = 1 + alpha;
      a1 = -2 * cs;
      a2 = 1 - alpha;
      break;
   case HPF:
      b0 = (1 + cs) /2;
      b1 = -(1 + cs);
      b2 = (1 + cs) /2;
      a0 = 1 + alpha;
      a1 = -2 * cs;
      a2 = 1 - alpha;
      break;
   case BPF:
      b0 = alpha;
      b1 = 0;
      b2 = -alpha;
      a0 = 1 + alpha;
      a1 = -2 * cs;
      a2 = 1 - alpha;
      break;
   case NOTCH:
```

```
    b0 = 1;
    b1 = -2 * cs;
    b2 = 1;
    a0 = 1 + alpha;
    a1 = -2 * cs;
    a2 = 1 - alpha;
    break;
case PEQ:
    b0 = 1 + (alpha * A);
    b1 = -2 * cs;
    b2 = 1 - (alpha * A);
    a0 = 1 + (alpha /A);
    a1 = -2 * cs;
    a2 = 1 - (alpha /A);
    break;
case LSH:
    b0 = A * ((A + 1) - (A - 1) * cs + beta * sn);
    b1 = 2 * A * ((A - 1) - (A + 1) * cs);
    b2 = A * ((A + 1) - (A - 1) * cs - beta * sn);
    a0 = (A + 1) + (A - 1) * cs + beta * sn;
    a1 = -2 * ((A - 1) + (A + 1) * cs);
    a2 = (A + 1) + (A - 1) * cs - beta * sn;
    break;
case HSH:
    b0 = A * ((A + 1) + (A - 1) * cs + beta * sn);
    b1 = -2 * A * ((A - 1) + (A + 1) * cs);
    b2 = A * ((A + 1) + (A - 1) * cs - beta * sn);
    a0 = (A + 1) - (A - 1) * cs + beta * sn;
    a1 = 2 * ((A - 1) - (A + 1) * cs);
    a2 = (A + 1) - (A - 1) * cs - beta * sn;
    break;
default:
    free(b);
    return NULL;
```

```
/* paydanın ilk katsayısı 1 olacak şekilde tekrara hesapla */
b->a0 = b0 /a0;
b->a1 = b1 /a0;
b->a2 = b2 /a0;
b->a3 = a1 /a0;
b->a4 = a2 /a0;
.
.
```

Parametrik Ekolayzır dışındaki filtreler konumuz dışı olduğundan teorik hesaplanmasına girilmemiştir ve sadece uygulamada deneme amaçlı kullanılmaktadır.

Fonksiyon parametreleri olarak

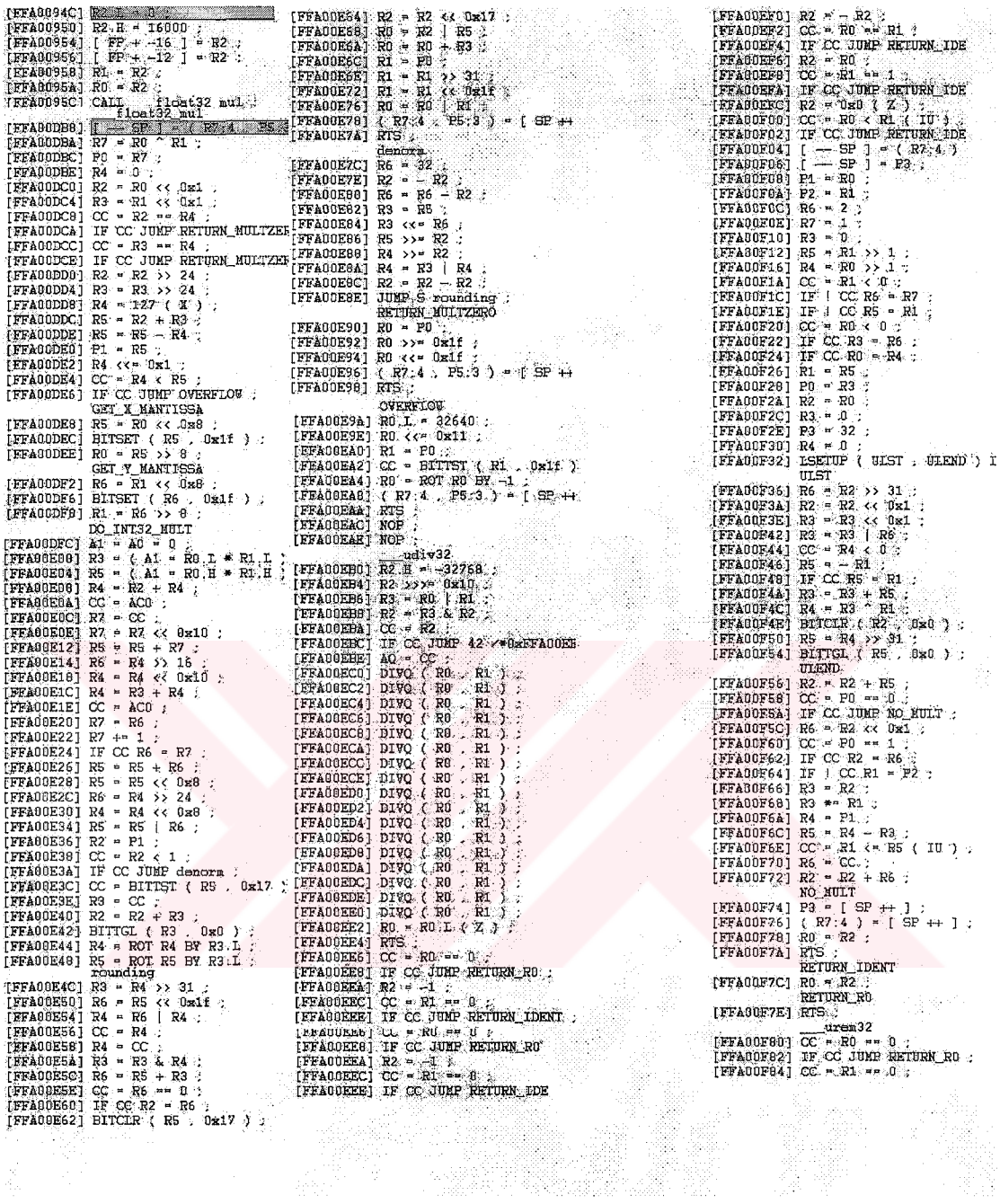
int type
smp_type dbGain
smp_type freq
smp_type srate
smp_type bandwidth

değerleri girilmektedir. Burada smp_type kayan noktalı işlemciler için double, sabit noktalı işlemciler için 1.15 formatını kullanan fract16' yı temsil eder.

4.2.2 Yazılım optimizasyonları

VisualDSP++ 16 bit versiyonu double formatını işlemci üzerinde emülasyon ile elde eder. fract16 yada fract32 olarak temsil edilen sabit sayı formatını ise doğrudan işlemci üzerinde işleme girer. VisualDSP++ 32 bit versiyonunda ise tam tersi bir durum geçerlidir. Yani kayan sayı işlemleri işlemci üzerinde doğrudan hesaplanabilmekte buna karşın sabit nokta işlemleri emülasyon ile elde edilmektedir. Aşağıdaki örnekte kayan noktalı hesaplama için üretilen assembly kodu ile sabit noktalı hesaplama için üretilen assembly kodu arasındaki fark verilmiştir.

```
float a,b;
a=0.25;
b=0.25;
a=a*b;          // Emülasyon yapılan çarpma işlemi
```



Şekil 4.6 Kayan sayı formatında çarpma işleminin emülasyonu için üretilen assembly kodu

---

fract16 x=0x2000; //1.15 formatında 0.25

fract16 y=0x2000; //1.15 formatında 0.25

x=mult_fr1x16(x, y); //1.15 formatında çarpma işlemi yapan fonk.

```
[FFA00962] R3 = 8192 ( X ) ;
[FFA00966] W [ FP + -8 ] = R3 ;
[FFA0096A] W [ FP + -4 ] = R3 ;
[FFA0096E] R1 = R3.L ( X ) ;
[FFA00970] R0 = W [ FP + -8 ] ( X )
[FFA00974] CALL mult_fr1x16 ;
           mult_fr1x16
[FFA00928] LINK 0x0 ;
[FFA0092C] [ FP + 0x8 ] = R0 ;
[FFA0092E] [ FP + 0xc ] = R1 ;
[FFA00930] R1 = R1.L ( X ) ;
[FFA00932] R0 = R0.L ( X ) ;
[FFA00934] R3.H = R0.H * R1.H , R3.I
[FFA00938] R0 = R3.L ( X ) ;
[FFA0093A] JUMP.S 2 /*0xFFA0093C*/ ;
[FFA0093C] P0 = [ FP + 0x4 ] ;
[FFA0093E] UNLINK ;
```

Şekil 4.7 Sabit sayı formatında çarpma işlemi için üretilen assembly kodu

Aşağıda Visual DSP++ üzerinde optimize edilmiş 1.15 formatındaki desteklenen matematiksel fonksiyon listesi verilmiştir.

fract16 add_fr1x16(fract16, fract16);

fract16 sub_fr1x16(fract16, fract16);

fract16 mult_fr1x16(fract16, fract16);

fract16 multr_fr1x16(fract16, fract16);

fract32 mult_fr1x32(fract16, fract16);

fract16 abs_fr1x16(fract16);

fract16 min_fr1x16(fract16, fract16);

fract16 max_fr1x16(fract16, fract16);

fract16 negate_fr1x16(fract16);

fract16 shl_fr1x16(fract16, int);

fract16 shr_fr1x16(fract16, int);

fract32 add_fr1x32(fract32, fract32);

fract32 sub_fr1x32(fract32, fract32);

fract32 mult_fr1x32x32(fract32, fract32) ;

fract32 abs_fr1x32(fract32);

fract32 min_fr1x32(fract32, fract32);

```
fract32 max_fr1x32(fract32, fract32);
fract32 negate_fr1x32(fract32);
fract32 shl_fr1x32(fract32, int);
fract32 shr_fr1x32(fract32, int);
fract16 sat_fr1x32(fract32);
fract16 round_fr1x16(fract32);
int norm_fr1x32(fract32);
int norm_fr1x16(fract16);
```

Bunun dışında yapılması gereken optimizasyonlar **Bölüm 3'** de verilmiştir.

4.2.3 Yazılım parametreleri

İlgili yazılım parametreleri aşağıda verilmiştir.

type : LPF , HPF,BPF,NOTCH,PEQ, LSH,HSH
dbGain : desibel cinsinden kazanç $\pm smp\_type$ boyutunda olmalı (10 dB gibi)
freq : Merkezi geçirme ya da söndürme frekansı (1000 Hz gibi)
srate : Örnekleme frekansı (44100 Hz gibi)
bandwidth : G/2 deki oktav olarak band genişliği (0.1 gibi)

Ayrıca fonksiyona band genişliği yerine Q-kalite faktörü cinsinden de parametre girilebilir. Çizelge 4.1'de çeşitli band genişliklerine karşılık düşen Q değerleri verilmiştir. Q 'nun 1 değerine yakınsadığı band genişliği değeri olan 1.40 tasarımcılar için çoğu zaman referans olarak alınmaktadır.

Blackfin serisi DSP'lerde maks. 148K byte işlemci üzerinde bellek bulunmaktadır. Bu bellek ile aşağıdaki oranlarda ayrılmıştır.

16K byte Kod SRAM/ Önbellek
64K byte Kod SRAM
32K byte Veri SRAM/ Önbellek
32K byte Veri SRAM
4K byte Geçici SRAM

Bu nedenle, modeline göre değişmekle beraber maks. 64K'nın üzerinde oluşturulan kodlar için kart üzerinde bulunan 32 MB SD-RAM bellek kullanılmalıdır. Fakat bu bellek kullanımı performans düşüklüğüne sebebiyet verir. Çünkü dış bellekten alınan veri ile iç bellekten alınan veri arasında hız farkı vardır. Bu dış belleği kullanıma açmak için aşağıdaki makro, proje LDF başlık dosyasında tanımlanmalıdır.

```
#define USE_CACHE 1
```

BW	Q	BW	Q	BW	Q
0,02	72,13	1,40	0,99	1,00	1,39
0,03	48,09	1,60	0,86	1,10	1,27
0,04	36,07	1,80	0,75	1,20	1,17
0,05	28,85	1,90	0,71	1,30	1,08
0,06	24,04	2,20	0,60	1,40	1,01
0,07	20,61	2,40	0,54	1,50	0,94
0,08	18,03	2,60	0,49	1,60	0,89
0,09	16,03	2,80	0,44	1,70	0,84
0,10	14,42	3,00	0,40	1,80	0,79
0,20	7,21	0,50	2,54	1,90	0,75
0,30	4,80	0,55	2,35	2,00	0,71
0,40	3,60	0,60	2,19	3,00	0,48
0,50	2,87	0,65	2,04	4,00	0,36
0,60	2,39	0,70	2,00	5,00	0,29
0,70	2,04	0,75	1,80	6,00	0,24
0,80	1,78	0,80	1,70	8,00	0,18
0,90	1,58	0,85	1,61	10,00	0,14
1,00	1,41	0,90	1,53	20,00	0,07
1,20	1,17	0,95	1,46	30,00	0,05

Çizelge 4.1 Çeşitli band genişliklerine karşılık düşen Q değerleri

Kart üzerinde bulunan AD1836 ses kodeği üzerindeki ADC çeşitli bit değerlerinde örnekleme yapabilmektedir. Bu değerler 16, 20, 24' dür. Projede en yüksek değer olan 24 bit kullanılmıştır. 24 bit ile oldukça yüksek bir ses kalitesi elde edilebilmektedir. Rasyonel olarak 105 dB SNR ve Dinamik Aralık elde edilmektedir. Kesme rutini içerisinde yer alan ses örneklerini alan fonksiyonda gelen örnekler 32 bit integer formatında gelir. Bunu double - kayan sayı formatına çevirebilmek ve (-1,1) aralığında ölçeklemek için aşağıda verilen kod kullanılır.

```
if(sample > 0x007fffff)
{
        isample = 0x01000000 - sample;
        stotal = (-1)*((double)(isample) / (1 << 23));
}
else
        stotal=((double)(sample)) / (1 << 23);
```

Burada örnekleme değeri 16 olsaydı sample değişkeni 15 bit sola ötelenmesi gerekirdi. Ayrıca çıkışa aktarabilmek için yukarıda verilen işlemin tersi yapılır. Yani

```
i = (int)(stotal * (1 << 23));
i = i  & 0x00ffffff;
```

kodu kullanılmıştır.

Ayrıca hesaplamada kayan sayı formatı (-1,1) aralığında olması gerektiğinden satürasyon edilmesi gerekmektedir. Bunun için de aşağıdaki kod bloğu kullanılmıştır. Satürasyon bloğu eklenmediği takdirde ses çıkışı bozulacaktır.

```
if(stotal >= 1)
        stotal = 0.9999;
else if(stotal <= -1)
        stotal = -0.9999;
```

Programın dış dünyadan mesaj alabilmesi için kart üzerinde bulunan 4 adet programlanabilir buton kullanılmıştır. Bunlara sırasıyla aşağıdaki atamalar yapılmıştır.

1. Ekolayzır fonksiyonu kapalı – Gelen işareti aynen çıkışa aktarır.
2. Ekolayzır fonksiyonu açık – dB kazanç değerleri 0
3. Ekolayzır fonksiyonu açık – Pozitif Bas
4. Ekolayzır fonksiyonu açık – Negatif Bas

Burada 2. fonksiyonun kullanılmasının sebebi ilk durumla aynı etkiyi verip vermediğinin kontrolünün yapılmasıdır. Gerçektende 2. durum çoğu zaman 1.durumla aynı ses düzeyinde elde edilmiştir. Ayrıca başlık dosyasında tanımlanan BAND10 değişkeni 1 yapıldığı durumda 10 band ekolayzır aktif hale getirilmiş olur.

5 SİSTEM PERFORMANSININ ÖLÇÜLMESİ

Bir donanım sistemi üzerinde çalışan yazılımın performans ölçümü önemli bir konudur. Teorikte geliştirilen bir algoritmanın pratikte donanım üzerinde yeterince hızlı çalışıp çalışmadığı test edilmelidir. Bu testlerle ilgili çok değişik yöntemler geliştirilmiş ve bazı standartlar ortaya koyulmuştur. Konumuz bu standartlar olmamakla beraber bu sistem üzerinde gerçeklenebilecek 2 performans ölçüm mantığı üzerinde durulacaktır.

1. İşaret işlemenin gerçekleştiği sistem kesme rutin yada rutinleri içerisinde kullanılan fonksiyonların aldığı maks. işlemci süresini hesaplamak gerekmektedir. Bu nedenle fonksiyonlardan üretilen assembly kodunun toplam süreleri bulunur. 48 kHz örnekleme yapan bir sistemde kesme rutini içerisine kullanılacak fonksiyonları maks süresi $\frac{1}{48000} = 2{,}083.10^{-5} sn$ olmalıdır. 750 MHz lik bir işlemcide bir komutun bir işlemci döngüsüne eşit olduğu varsayımı ile $\frac{1}{750.000.000} = 1{,}33.10^{-9} sn$ sürmektedir. Böylece kesme rutini içerisinde kullanılabilecek maks. komut sayısı $\frac{750.000.000}{48000} \approx 15000$ dir. Normal şartlar altında çok hassas durumlar dışında bu tür hesaplamalar yapılmaz. Çünkü yazılım boyutlarının zamanla büyümesi, her kod değişikliği için assembly kod uzunluğunun tekrar hesaplanması gerekliliği gibi … Bunun yerine 2. bir yöntem olarak Gerçek Zaman Saati(RTC)'nden yararlanılabilir.

2. Bilindiği üzere 48 kHz'lik işaret işleyen bir kesme rutini içerisine saniyede 48000 defa girilmesi gerekmektedir. Projede kullanılan işlemci dışarıdan 32.768 kHz kristal bağlamak suretiyle RTC fonksiyonlarını sağlayabilir. Aşağıdaki kod bloğu donanım kiti üzerinde bulunan RTC nin aktif hale getirilmesini sağlar.

```
void Init_RTC(void){
    int x;
    //kesme rutinine 1 sn de kaç defa girdiğini sayacak değişken
    iTickCount = 0;
    //Bütün saat kesmeleri pasif
```

```
*pRTC_ICTL = 0x0000;
/* Write Complete için bekleme */
do{
        x=(*pRTC_ISTAT);
        x=x&0x00008000;
}while(x!=0x00008000);
//RTC yi 32Khz moduna al
*pRTC_PREN = 0x0000;
//Saat olayı  ve yazma işleminin tamamlanmasını bekle
do{
        x=(*pRTC_ISTAT);
} while((x<0x0000803c) || (x > 0x0000c000));

// RTC yi 1 Hz moduna al
*pRTC_PREN = 0x0001;
/* Write Complete için bekleme */
do{
        x=(*pRTC_ISTAT);
        x=x&0x00008000;
}while(x!=0x00008000);
//1 sn lik kesme ayarı
*pRTC_ICTL = 0x0004;
/* Write Complete için bekleme */
do{
        x=(*pRTC_ISTAT);
        x=x&0x00008000;
}while(x!=0x00008000);
//Saat olaylarını temizle
*pRTC_ISTAT = 0xffff;
```

iTickCount değişkeni kesme rutini içerisine kaç defa girildiğini göstermektedir. Aşağıdaki tabloda bazı durumlar için değişken değeri verilmiştir.

	iTickCount
a. Tek Kanal	47996
b. Tek Kanal 2 Band Filtre	47996
c. Çift Kanal 2 Band Filtre	32676
d. Tek Kanal 10 Band Filtre	17102
e. Çift Kanal 10 Band Filtre	8630

Çizelge 5.1 Optimize olmayan kod ile elde edilen 1sn'lik kesme rutini giriş sayısı

Tablodan da görüldüğü üzere 1sn'de 48000 defa giriş işlemini başarabilen tek sistem **Tek Kanal 2 Band Filtre'**de elde edilmiştir. Bunun nedeni **Bölüm 3** ve **3.2.2** de belirtilen optimizasyonların yapılmamasıdır. Bu nedenle **b, c, d, e** deki sistemler kullanılamaz. Gerçekte de **a**. da ki sistem dışındaki modellerde ses işaretinde gittikçe daha belirginleşen ses bozulmaları meydana geldiği gözlenmiştir.

6 SONUÇ

Bu projede bir DSP üzerinde ses işareti işlemeye dayalı bir sistemin nasıl kurulacağı ve bilgisayar platformu üzerinde nasıl simüle edileceği üzerinde durulmuştur. Ayrıca güncel DSP işlemci türleri arasındaki önemli farklara değinilmiş ve bunlar arasında nasıl geçiş yapılabileceği konu edilmiştir.

İkinci dereceden bikuad IIR filtreleme en temel işaret işleme teorilerinden biri olması nedeniyle bu projede kullanılmak üzere seçilmiştir. Filtre sistemi bilgisayar platformu üzerinde simüle edilerek, DSP üzerinde gerçek zamanlı yapının nasıl kurulabileceği hususunda fikir verilmiştir. DSP üzerinde genel olarak ANSI C ile girilen yazılımın sistem performansı açısından yeterli gelmediği görülmüştür. Bu açıdan belirtilen optimizasyonların gerçekleştirilmesinin önemi kavranmıştır.

Bu sisteminin sonraki tezlerde kullanılmak üzere geliştirilebilecek şekilde yazılımları kodlanmıştır. Özellikle sonraki aşamalarda adaptif filtreleme konularına değinilebileceği düşünüldüğünden herhangi bir algoritmanın gerçek zamanlı koşturulabileceği şekilde bir yapı oluşturulmuştur. Ayrıca, sayısal kontrolörlerin de bir çeşit sayısal filtre oldukları düşünülürse, kurulan gerçek zamanlı çalışabilen, yüksek performanslı platformun, dijital kontrol uygulamaları için de çok uygun bir temel oluşturacağı umulmaktadır. Sistemin üstün yetenekleri, yüksek hız ve işlem kapasitesi gerektiren gelişmiş işaret işleme ve otomatik kontrol algoritmalarının gerçek zamanlı, hassas ve doğru çalışabilmelerini mümkün kılacaktır.

Akademik çalışmalara ışık tutması dileğiyle...

KAYNAKLAR

[1] DENBIGH Philip, (1998), "System Analysis and Signal Processing",
Addison Wesley Longman Ltd.

[2] OPPENHEIM A., SCHAFFER R. , (2001), "Discrete Time Signal Processing - Second Edition ",
Prentice Hall

[3] KUO Benjamin. C ., (1992), "Digital Control Systems - Second Edition ",
Oxford University Press

[4] VAN VALCENBERK M.E., (1982), "Analog Filter Design",
HRW Series in Electrical and Computer Engineering
CBS College Publishing

[5] "Digital Stereo 10-Band Graphic Equalizer Using the DSP56001", (1998)
Motorola Inc.

INTERNET KAYNAKLARI

[6]AGUADO Axel, BISHOP James, RADDATZ Chad, WAITE Tom, (2004), "Active Noise Control Final Report",
http://seniord.ee.iastate.edu/may0421/ANC_final_report.pdf

[7]BALCZARCZYK T. - GULDENER R., (2003), "Real-Time Audio Equalizer mittels Matlab und Simulink Studienarbeit WS02/03 Hochschule Rapperswil HSR",
http://www.medialab.ch/archiv/pdf_studien_diplomarbeiten/1sa03/audio_equalizer_mittels_simulink.pdf

[8] "Blackfin® Embedded Processor ADSP-BF531/ADSP-BF532/ADSP-BF533 Data Sheet", (2004),
Analog Devices, Inc.
http://www.analog.com/UploadedFiles/Data_Sheets/144127970ADSP_BF531_2_3_a.pdf

[9] "Filter Design Equations", (2004)
Apogee Technology Inc.,
http://www.apogeeddx.com/AN-06.pdf

[10]GAN Woon S. - KU Sen M., (2004), "Transition from Simulink to MATLAB in Real – Time Digital Signal Processing Education",
http://eeeweba.ntu.edu.sg/DSPLab/geq/26_Gan.pdf

[11]JOHNSON - BRISTOW Robert, (2004), "The Equivalence of Various Methods of Computing Biquad Coefficients for Audio Parametric Equalizers" ,
Wave Mechanics, Inc., Burlington VT
http://www.harmony-central.com/Effects/Articles/EQ_Coefficients/EQ-Coefficients.pdf

[12]JOHNSON - BRISTOW Robert, (2003), "Cookbook Formulae for Audio Equalization Biquad Filter Coefficients",
Wave Mechanics, Inc., Burlington VT
http://www.harmony-central.com/Computer/Programming/Audio-EQ-Cookbook.txt

[13]TOMARAKOS John - LEDGER Dan Using, (1998), "The Low-Cost, High Performance ADSP-21065L Digital Signal Processor for Digital Audio Applications",
Analog Devices, Inc.
http://www.analog.com/UploadedFiles/Application_Notes/7056820721065L_Audio_Tutorial.pdf

ÖZGEÇMİŞ

Doğum tarihi 17.06.1979
Doğum yeri Malatya
Ortaokul 1990-1994 Malatya Anadolu Lisesi
Lise 1994-1997 Malatya Fen Lisesi
Lisans 1997-2002 İstanbul Teknik Üniversitesi Elektrik Elektronik Fakültesi Kontrol ve Bilgisayar Mühendisliği Bölümü

Çalıştığı kurumlar

2002-2003 Netvizyon Yazılım Ltd.
2003-2004 İkomSistem Ltd.
2004-2005 Satko Telekomünikasyon A.Ş