AN3984
Application note

IIR filter design equations for Sound Terminal® devices

Introduction

The purpose of this document is to provide a tool to calculate the IIR filter coefficients to program the Sound Terminal® devices from STMicroelectronics.

For each filter the procedure and the formulas to calculate the coefficient will be described; the Matlab code is given in *Appendix A: Matlab code (functions) on page 24*.

A generalized set of equations can be formulated for the design of first-order low-pass and high-pass filters and of second-order filters.

A specialized set of equations is devised for designing parametric biquad EQ filters. As with any other filter design procedure, the desired characteristics of the filter are to be made available.

The parameters governing the characteristics of each filter are:

- fc: filter cutoff frequency which is the -3dB corner frequency or the midpoint frequency in a peak or notch filter
- fs: sampling frequency
- Q: quality factor (not applicable for low and high-shelf filters)
- Slope: applicable only for low and high-shelf filters
- Gain: the boost or the attenuation at f = fc

These parameters can be used to determine the coefficients of the digital filter transfer function.

www.st.com

Contents

List of figures

1 Overview

The transfer function for a first-order filter in the digital z-domain is:

Equation 1

$$H(z) = \frac{b_0 + b_1 \cdot z^{-1}}{a_0 + a_1 \cdot z^{-1}}$$

For a second-order filter (a biquad) the transfer function is:

Equation 2

$$H(z) = \frac{b_0 + b_1 \cdot z^{-1} + b_2 \cdot z^{-2}}{a_0 + a_1 \cdot z^{-1} + a_2 \cdot z^{-2}}$$

This equation can be modified normalizing the a_0 coefficient; the new equation is:

Equation 3

$$H(z) = \frac{(b_0/a_0) + (b_1/a_0) \cdot z^{-1} + (b_2/a_0) \cdot z^{-2}}{1 + (a_1/a_0) \cdot z^{-1} + (a_2/a_0) \cdot z^{-2}}$$

The most straightforward implementation form using *Equation 3* is:

Equation 4

$$y[n] = (b_0/a_0) \cdot x[n] + (b_1/a_0) \cdot x[n-1] + (b_2/a_0) \cdot x[n-2] - (a_1/a_0) \cdot y[n-1] - (a_2/a_0) \cdot y[n-2]$$

2 Filter stability

2.1 Definition

A filter is said to be stable in the z-domain if the roots (or poles) of the filter lie inside the unit circle.

This definition of stability can be translated in terms of the filter coefficients.

2.2 First-order filter

For a first-order filter, the stability condition that needs to be satisfied is that the pole of the filter lies within the unit circle.

In terms of the coefficients, the condition can be given as:

Equation 5

$$|a_1| < 1$$

2.3 Second-order filter

For a 2^{nd}-order filter, two conditions must be satisfied to ensure filter stability and translated in terms of the filter coefficients they are:

Equation 6

$$|a_2| < 1$$
$$|a_1| < (1 + a_2)$$

3 First-order filter design (LPF and HPF)

The preliminary step to obtain the coefficients for the first-order low-pass filter or high-pass filter is to define three constants obtained from the filter parameters:

Equation 7

$$\omega_c = 2 \cdot \pi \cdot f_c / f_s$$
$$K = \tan(\omega_c / 2)$$
$$\alpha = 1 + K$$

In a first-order filter both the coefficients a_2 and b_2 are null.

The denominator coefficients are identical for both an LPF and an HPF designed for the same cutoff frequency and they are computed as follows:

Equation 8

$$a_0 = 1$$
$$a_1 = -\frac{(1-K)}{\alpha}$$

The numerator for an LPF can be calculated as follows:

Equation 9

$$b_0 = \frac{K}{\alpha}$$
$$b_1 = \frac{K}{\alpha}$$

The numerator for an HPF can be calculated as follows:

Equation 10

$$b_0 = \frac{1}{\alpha}$$
$$b_1 = -\frac{1}{\alpha}$$

The coefficient used in APWorkbench can be calculated by applying these formulas:

Equation 11

$$Coefficient\ {}^{b_1}\!/_2 = \frac{(b_1/2)}{a_0}$$
$$Coefficient\ b_2 = \frac{(b_2)}{a_0}$$
$$Coefficient\ {}^{a_1}\!/_2 = \frac{(-a_1/2)}{a_0}$$
$$Coefficient\ a_2 = \frac{-a_2}{a_0}$$
$$Coefficient\ {}^{b_0}\!/_2 = \frac{(b_0/2)}{a_0}$$

4 Second-order filter design

4.1 Low-pass and high-pass filters

The preliminary step to obtain the coefficients for a second-order filter is the calculation of these coefficients obtained from the filter parameters:

Equation 12

$$\vartheta_c = 2 \cdot \pi \cdot \frac{f_c}{f_s}$$
$$K = \tan(\omega_c/2)$$
$$W = K^2$$
$$\alpha = 1 + K$$
$$DE = 1 + \frac{K}{Q} + W$$

The denominator coefficients are the same for both an LPF and an HPF if designed for the same cutoff frequency. They are computed as follows:

Equation 13

$$a_0 = 1$$
$$a_1 = 2 \cdot \frac{(W-1)}{DE}$$
$$a_2 = \frac{1 - \frac{K}{Q} + W}{DE}$$

4.1.1 Low-pass filter

The numerator coefficient for a second-order LPF can be calculated as follows:

Equation 14

$$b_0 = \frac{W}{DE}$$
$$b_1 = 2 \cdot \frac{W}{DE}$$
$$b_2 = \frac{W}{DE}$$

For a second-order LPF, the coefficients given in APWorkbench can be calculated as follows:

Equation 15

$$\begin{aligned}
&Coefficient\ {}^{b_1}\!/_{2} = \frac{W}{DE}\\
&Coefficient\ b_2 = \frac{W}{DE}\\
&Coefficient\ {}^{a_1}\!/_{2} = -1 \cdot \frac{W-1}{DE}\\
&Coefficient\ a_2 = -\frac{1-\frac{K}{Q}+W}{DE}\\
&Coefficient\ {}^{b_0}\!/_{2} = \frac{1}{2} \cdot \frac{W}{DE}
\end{aligned}$$

4.1.2 High-pass filter

The numerator coefficient for a second-order HPF can be calculated as follows:

Equation 16

$$\begin{aligned}
&b_0 = \frac{1}{DE}\\
&b_1 = -2 \cdot \frac{W}{DE}\\
&b_2 = \frac{1}{DE}
\end{aligned}$$

For a second-order HPF, the coefficients given in APWorkbench can be calculated as follows:

Equation 17

$$\begin{aligned}
&Coefficient\ {}^{b_1}\!/_{2} = -\frac{1}{DE}\\
&Coefficient\ b_2 = \frac{1}{DE}\\
&Coefficient\ {}^{a_1}\!/_{2} = -1 \cdot \frac{W-1}{DE}\\
&Coefficient\ a_2 = -\frac{1-\frac{K}{Q}+W}{DE}\\
&Coefficient\ {}^{b_0}\!/_{2} = \frac{1}{2} \cdot \frac{1}{DE}
\end{aligned}$$

4.2 Peak filters

The first step is the calculation of the constant gain obtained from the gain filter parameter (GdB is expressed in dB).

Equation 18

$$Gain = \exp(Gain_{dB} \cdot 0.115129254)$$

The filter coefficients are different if the gain is positive or negative.

4.2.1 Peak filter - negative gain (cut)

The cut value is calculated with the following equation:

Equation 19

$$CutValue = 1 + K \cdot \left(\frac{Q}{Gain}\right) + W$$

The filter coefficient can be calculated as follows:

Equation 20

$$\begin{aligned}
a_0 &= 1 \\
a_1 &= 4 \cdot \frac{(W-1)}{CutValue} \\
a_2 &= \left(\frac{1 - \frac{Q}{Gain} + W}{CutValue}\right) \\
b_0 &= \frac{\left(1 + \frac{K}{Q} + W\right)}{CutValue} \\
b_1 &= 2 \cdot \frac{(W-1)}{CutValue} \\
b_2 &= \frac{(1 - \frac{K}{Q} + W)}{CutValue}
\end{aligned}$$

The coefficients in the APWorkbench are consequently calculated as follows:

Equation 21

$$Coefficient\ {}^{b_1}\!/_{2} = \frac{W-1}{CutValue}$$

$$Coefficient\ b_2 = \frac{(1-\frac{K}{Q}+W)}{CutValue}$$

$$Coefficient\ {}^{a_1}\!/_{2} = \frac{1-W}{CutValue}$$

$$Coefficient\ a_2 = -\frac{1-\frac{Q}{Gain}\cdot K+W}{CutValue}$$

$$Coefficient\ {}^{b_0}\!/_{2} = \frac{1}{2}\cdot\frac{1+\frac{K}{Q}+W}{CutValue}$$

4.2.2 Peak filter - positive gain (boost)

The boost value is calculated with the following equation:

Equation 22

$$BoostValue = 1+\frac{K}{Q}+W$$

The filter coefficient can be calculated as follows:

Equation 23

$$a_0 = 1$$

$$a_1 = \frac{(1+K\cdot\frac{Gain}{Q}+W)}{BoostValue}$$

$$a_2 = \frac{\left(1-\frac{K}{Q}+W\right)}{BoostValue}$$

$$b_0 = 2\cdot\frac{(1+K\cdot\frac{Gain}{Q}+W)}{BoostValue}$$

$$b_1 = 2\cdot\frac{W-1}{BoostValue}$$

$$b_2 = \frac{(1-K\cdot\frac{Gain}{Q}+W)}{BoostValue}$$

The coefficients in the APWorkbench are consequently calculated as follows:

Equation 24

$$Coefficient\ b_1/2 = \frac{W-1}{BoostValue}$$

$$Coefficient\ b_2 = \frac{(1-\frac{Gain}{Q}\cdot K+W)}{BoostValue}$$

$$Coefficient\ a_1/2 = \frac{1-W}{BoostValue}$$

$$Coefficient\ a_2 = -\frac{1-\frac{K}{Q}+W}{BoostValue}$$

$$Coefficient\ b_0/2 = \frac{1}{2}\cdot\frac{1+\frac{Gain}{Q}\cdot K+W}{BoostValue}$$

4.3 Shelf filters

The coefficient gain is defined in *Equation 25*.

Equation 25

$$Gain = 10^{\left(Gain_{dB}/40\right)}$$

The coefficients α and β are calculated as follows:

Equation 26

$$\alpha = \frac{\sin(\vartheta_c)}{2}\cdot\sqrt{\left(\left(Gain+\left(\frac{1}{Gain}\right)\right)\cdot\left(\frac{1}{S}-1\right)+2\right)}$$

$$\beta = 2\cdot\alpha\cdot\sqrt{Gain}$$

4.3.1 Low-shelf filter

The coefficients for an LSF can be calculated as follows:

Equation 27

$$a_0 = (Gain+1)+(Gain-1)\cdot\cos\vartheta_c+\beta$$

$$a_1 = -2\cdot(Gain-1)+(Gain+1)\cdot\cos\vartheta_c$$

$$a_2 = (Gain+1)+(Gain-1)\cdot\cos\vartheta_c-\beta$$

$$b_0 = Gain\cdot((Gain+1)-(Gain-1)\cdot\cos\vartheta_c+\beta)$$

$$b_1 = 2\cdot Gain\cdot((Gain-1)-(Gain+1)\cdot\cos\vartheta_c)$$

$$b_2 = Gain\cdot((Gain+1)-(Gain-1)\cdot\cos\vartheta_c-\beta)$$

The coefficient to load in APWorkbench can be calculated by applying the calculation already shown in *Equation 11*.

4.3.2 High-shelf filter

The coefficients for an HSF can be calculated as follows:

Equation 28

$$
\begin{aligned}
a_0 &= (Gain+1)-(Gain-1)\cdot\cos\vartheta_c+\beta\\
a_1 &= 2\cdot(Gain-1)-(Gain+1)\cdot\cos\vartheta_c\\
a_2 &= (Gain+1)-(Gain-1)\cdot\cos\vartheta_c-\beta\\
b_0 &= Gain\cdot((Gain+1)+(Gain-1)\cdot\cos\vartheta_c+\beta)\\
b_1 &= -2\cdot Gain\cdot((Gain-1)-(Gain+1)\cdot\cos\vartheta_c)\\
b_2 &= Gain\cdot((Gain+1)+(Gain-1)\cdot\cos\vartheta_c-\beta)
\end{aligned}
$$

The coefficient to load in APWorkbench to program a HSF can be computed by applying the formulas shown in *Equation 11*.

4.4 Notch filter

The first step is to define the constant α[a]:

Equation 29

$$\alpha=\frac{\sin(\vartheta_c)}{2\cdot Q}$$

The coefficients for a notch filter can be calculated as follows:

Equation 30

$$
\begin{aligned}
a_0 &= 1+\alpha\\
a_1 &= -2\cdot\cos\vartheta_c\\
a_2 &= 1-\alpha\\
b_0 &= 1\\
b_1 &= -2\cdot\cos\vartheta\\
b_2 &= 1
\end{aligned}
$$

The coefficients to load in APWorkbench can be calculated using *Equation 11*.

a. ϑ_c is defined in *Equation 12*

4.5 All-pass filter

Equation 29 allows calculating the constant α.

The coefficients for an APF can be calculated as follows:

Equation 31

$$
\begin{aligned}
a_0 &= 1 + \alpha \\
a_1 &= -2 \cdot \cos \vartheta_c \\
a_2 &= 1 - \alpha \\
b_0 &= 1 - \alpha \\
b_1 &= -2 \cdot \cos \vartheta_c = a_1 \\
b_2 &= 1 + \alpha
\end{aligned}
$$

The coefficients to load in APWorkbench can be calculated using *Equation 11*.

4.6 Band-pass filter

Equation 29 allows calculating the constant α while *Equation 32* is used to calculate the normalized gain.

Equation 32

$$NormGain = 10^{\left(\frac{Gain_{dB}}{20}\right)}$$

The coefficients for a BPF can be calculated as follows[b]:

Equation 33

$$
\begin{aligned}
a_0 &= 1 + \alpha \\
a_1 &= -2 \cdot \cos \vartheta_c \\
a_2 &= 1 - \alpha \\
b_0 &= \alpha \cdot NormGain \\
b_1 &= 0 \\
b_2 &= -b_0 = -\alpha \cdot NormGain
\end{aligned}
$$

The coefficients to load in APWorkbench can be calculated using *Equation 11*.

b. α is defined in *Equation 29*, ϑ_c is defined in *Equation 12*.

5 Examples

5.1 1st-order low-pass filter

Input data:

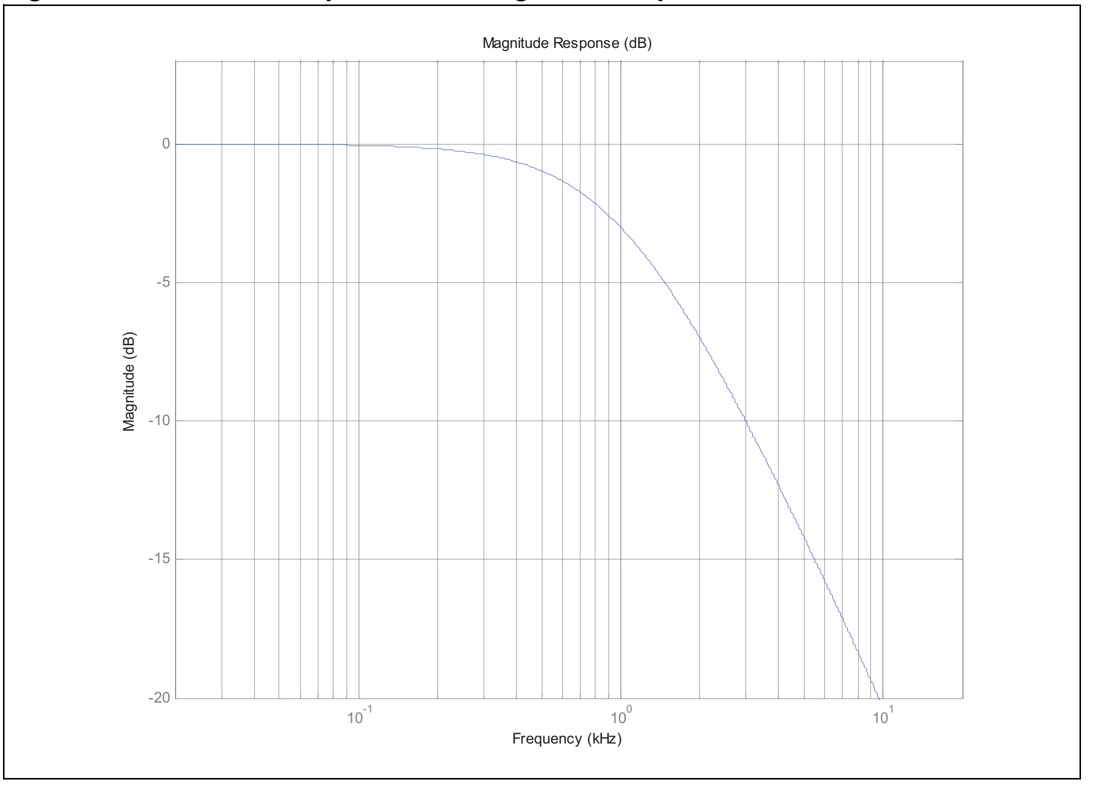
- Cutoff freq: 1 kHz
- Coefficient range: 4
- Processing frequency: 96 kHz

Output data:

Filter coefficients

'Coeff 1: b1/2'	'Coeff 2: b2'	'Coeff 3: -a1/2'	'Coeff 4: -a2'	'Coeff 5: b0/2'
'0081d6'	'000000'	'0efc52'	'000000'	'0081d6'

Figure 1. 1st-order low-pass filter - magnitude response



5.2 1st-order high-pass filter

Input data:

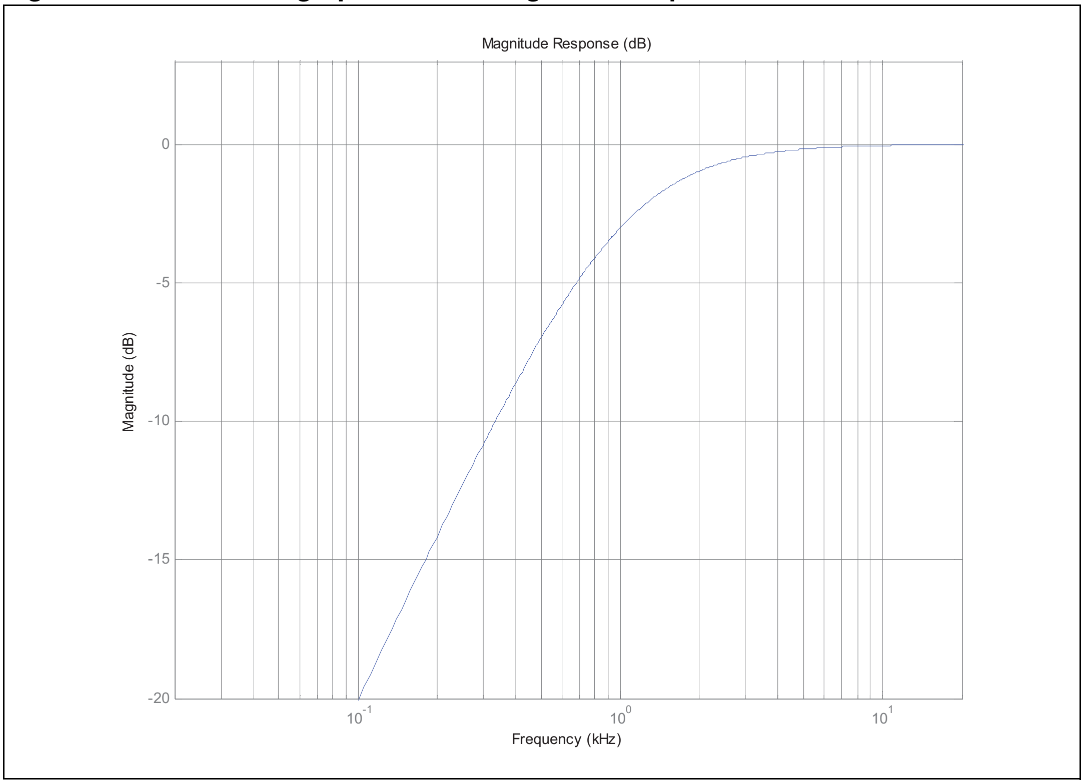
- Cutoff freq: 1 kHz
- Coefficient range: 4
- Processing frequency: 96 kHz

Output data:

Filter coefficients

'Coeff 1: b1/2'	'Coeff 2: b2'	'Coeff 3: -a1/2'	'Coeff 4: -a2'	'Coeff 5: b0/2'
'f081d6'	'000000'	'0efc52'	'000000'	'0f7e29'

Figure 2. 1st-order high-pass filter - magnitude response



5.3 2nd-order low-pass filter

Input data:

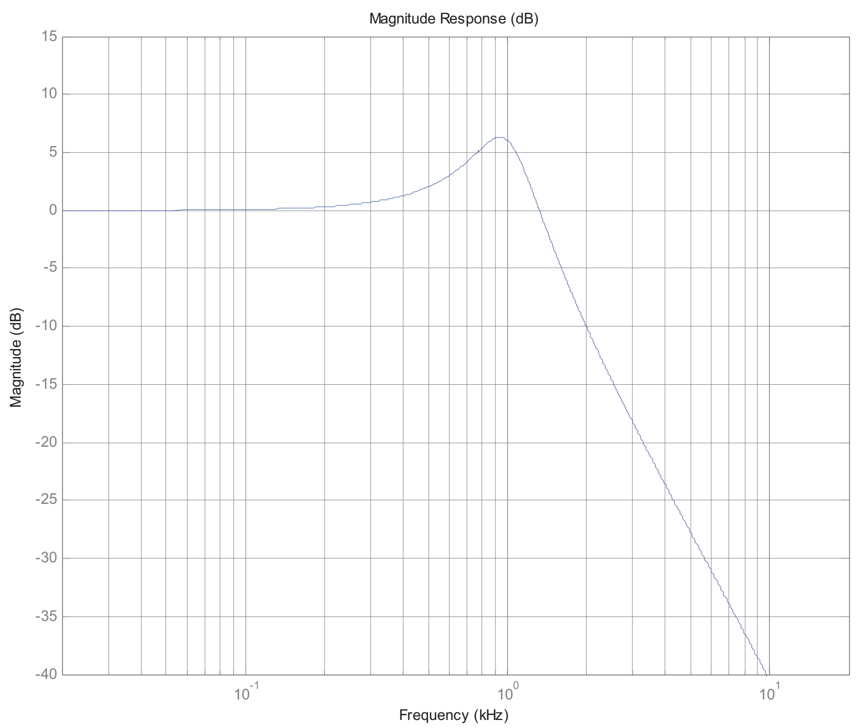
- Cutoff freq: 1 kHz
- Coefficient range: 4
- Quality factor (Q): 2
- Processing frequency: 96 kHz

Output data:

Filter coefficients

'Coeff 1: b1/2'	'Coeff 2: b2'	'Coeff 3: -a1/2'	'Coeff 4: -a2'	'Coeff 5: b0/2'
'0008a0'	'0008a0'	'1f6af3'	'e10794'	'000450'

Figure 3. 2nd-order low-pass filter - magnitude response



5.4 2nd-order high-pass filter

Input data:

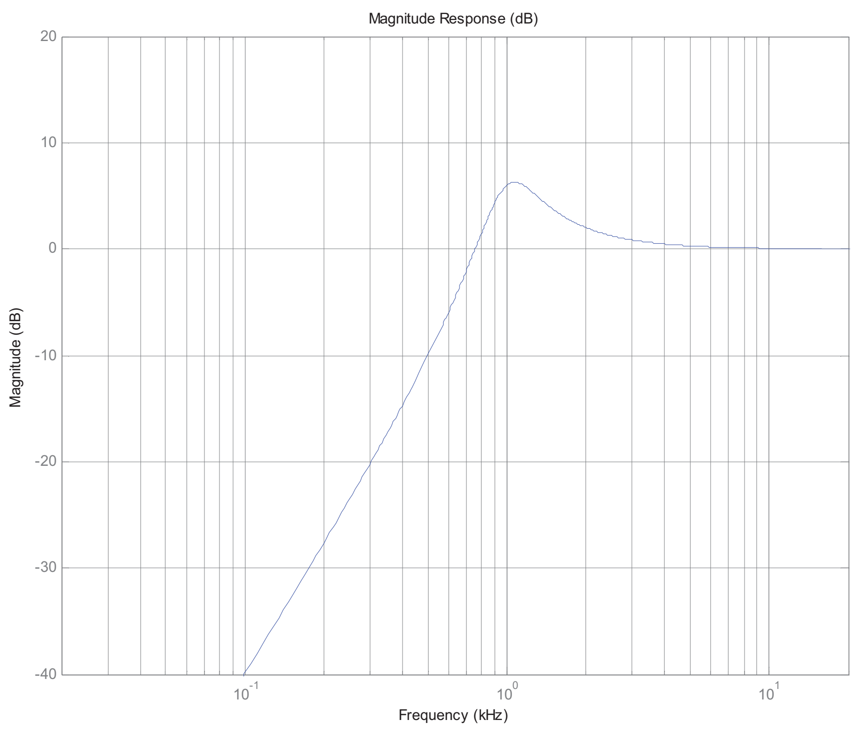
- Cutoff freq: 1 kHz
- Coefficient range: 4
- Quality factor (Q): 2
- Processing frequency: 96 kHz

Output data:

Filter coefficients

'Coeff 1: b1/2'	'Coeff 2: b2'	'Coeff 3: -a1/2'	'Coeff 4: -a2'	'Coeff 5: b0/2'
'e08c6b'	'1f7394'	'1f6af3'	'e10794'	'0fb9ca'

Figure 4. 2nd-order high-pass filter - magnitude response



5.5 Low-shelf filter

Input data:

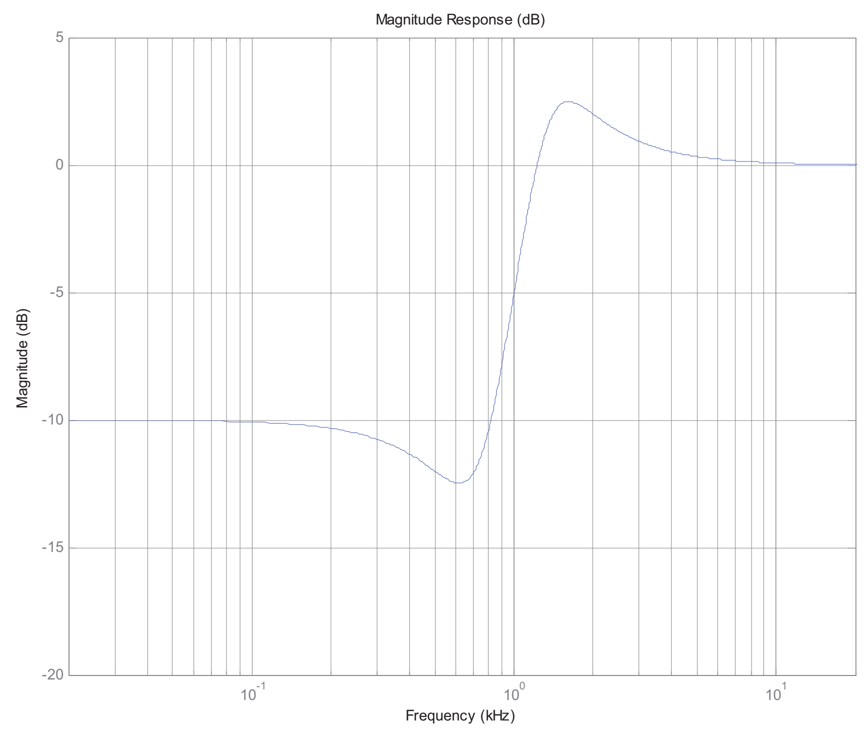
- Cutoff freq: 1 kHz
- Gain: -10 dB
- Coefficient range: 4
- Slope: 2
- Processing frequency: 96 kHz

Output data:

Filter coefficients

'Coeff 1: b1/2'	'Coeff 2: b2'	'Coeff 3: -a1/2'	'Coeff 4: -a2'	'Coeff 5: b0/2'
'e0f9f2'	'1e8e49'	'1efbb2'	'e1cc06'	'0fc87d'

Figure 5. Low-shelf filter - magnitude response



5.6 High-shelf filter

Input data:

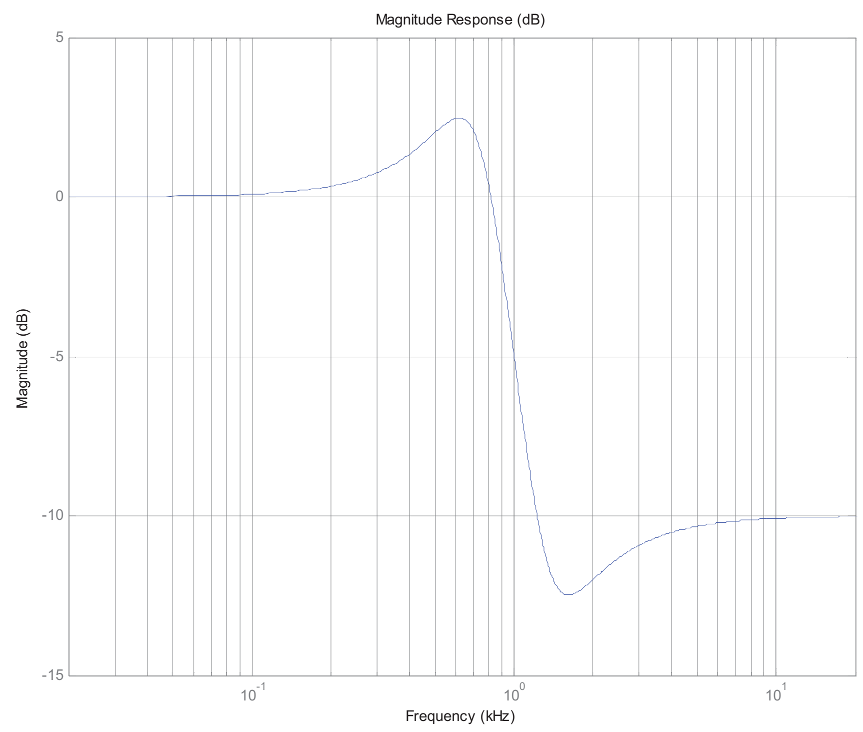
- Cutoff freq: 1 kHz
- Gain: -10 dB
- Coefficient range: 4
- Slope: 2
- Processing frequency: 96 kHz

Output data:

Filter coefficients

'Coeff 1: b1/2'	'Coeff 2: b2'	'Coeff 3: -a1/2'	'Coeff 4: -a2'	'Coeff 5: b0/2'
'f61151'	'09aea8'	'1f732a'	'e1063e'	'052110'

Figure 6. High-shelf filter - magnitude response



5.7 Notch filter

Input data:

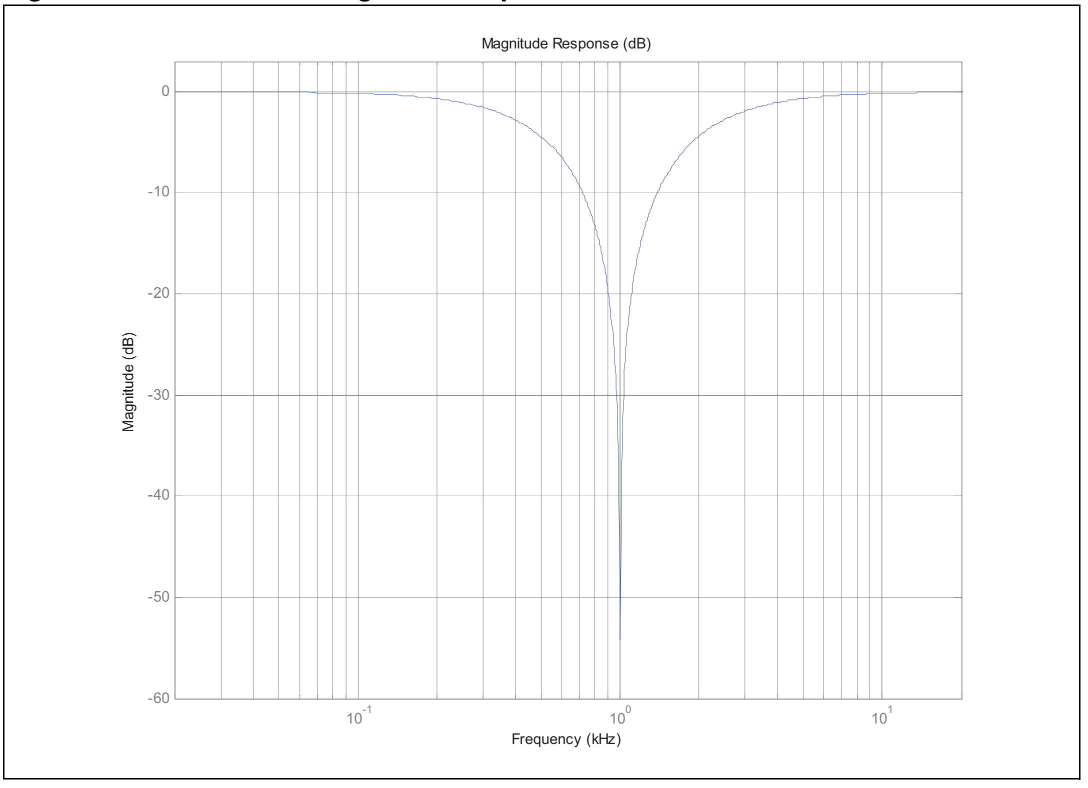
- Cutoff freq: 1 kHz
- Quality factor: 0.5
- Coefficient range: 4
- Processing frequency: 96 kHz

Output data:

Filter coefficients

'Coeff 1: b1/2'	'Coeff 2: b2'	'Coeff 3: -a1/2'	'Coeff 4: -a2'	'Coeff 5: b0/2'
'e2075a'	'1e091b'	'1df8a5'	'e3edc8'	'0f048d'

Figure 7. Notch filter - magnitude response



5.8 All-pass filter

Input data:

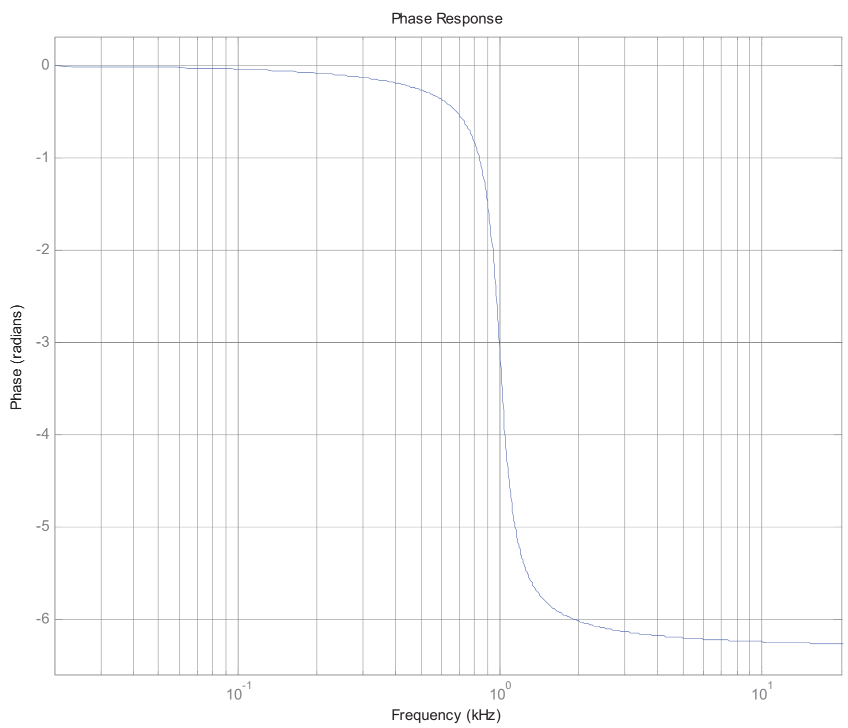
- Cutoff freq: 1 kHz
- Quality factor: 5
- Coefficient range: 4
- Processing frequency: 96 kHz

Output data:

Filter coefficients

'Coeff 1: b1/2'	'Coeff 2: b2'	'Coeff 3: -a1/2'	'Coeff 4: -a2'	'Coeff 5: b0/2'
'e046a7'	'200000'	'1fb958'	'e06a75'	'0fcac5'

Figure 8. All-pass filter - phase response



5.9 Band-pass filter

Input data:

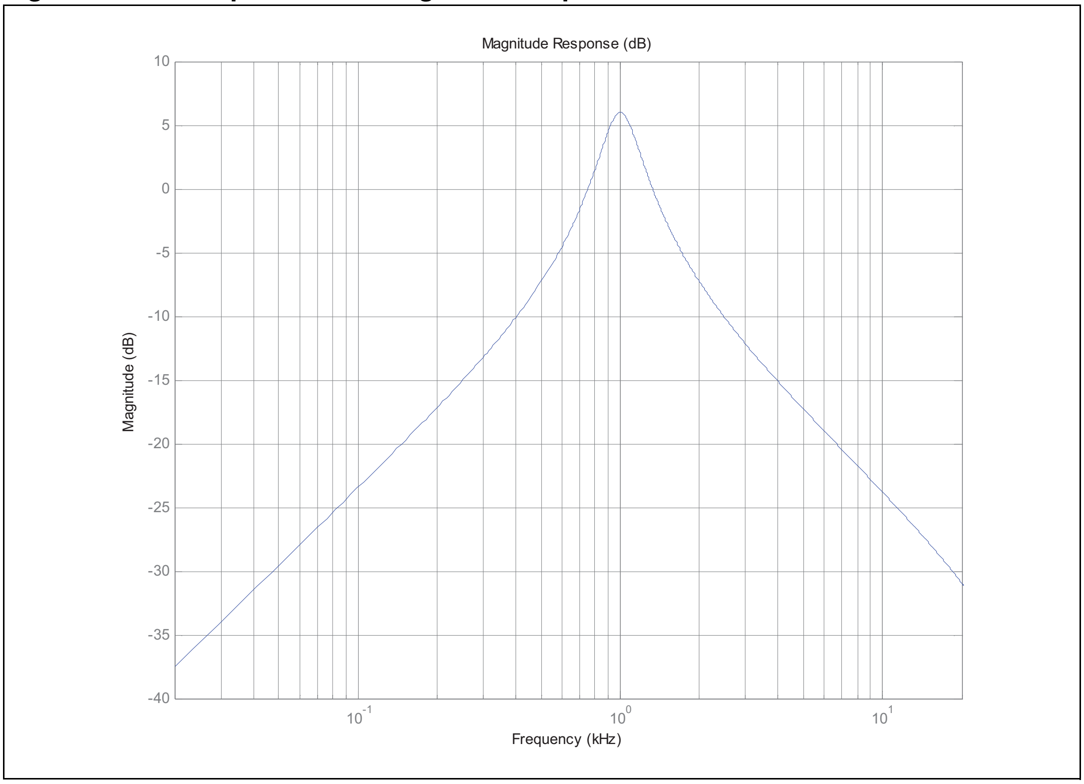
- Cutoff freq: 1 kHz
- Gain: +6 dB
- Quality factor: 3
- Coefficient range: 4
- Processing frequency: 96 kHz

Output data:

Filter coefficients

'Coeff 1: b1/2'	'Coeff 2: b2'	'Coeff 3: -a1/2'	'Coeff 4: -a2'	'Coeff 5: b0/2'
'000000'	'ff4fc0'	'1f9650'	'e0b0ab'	'00581f'

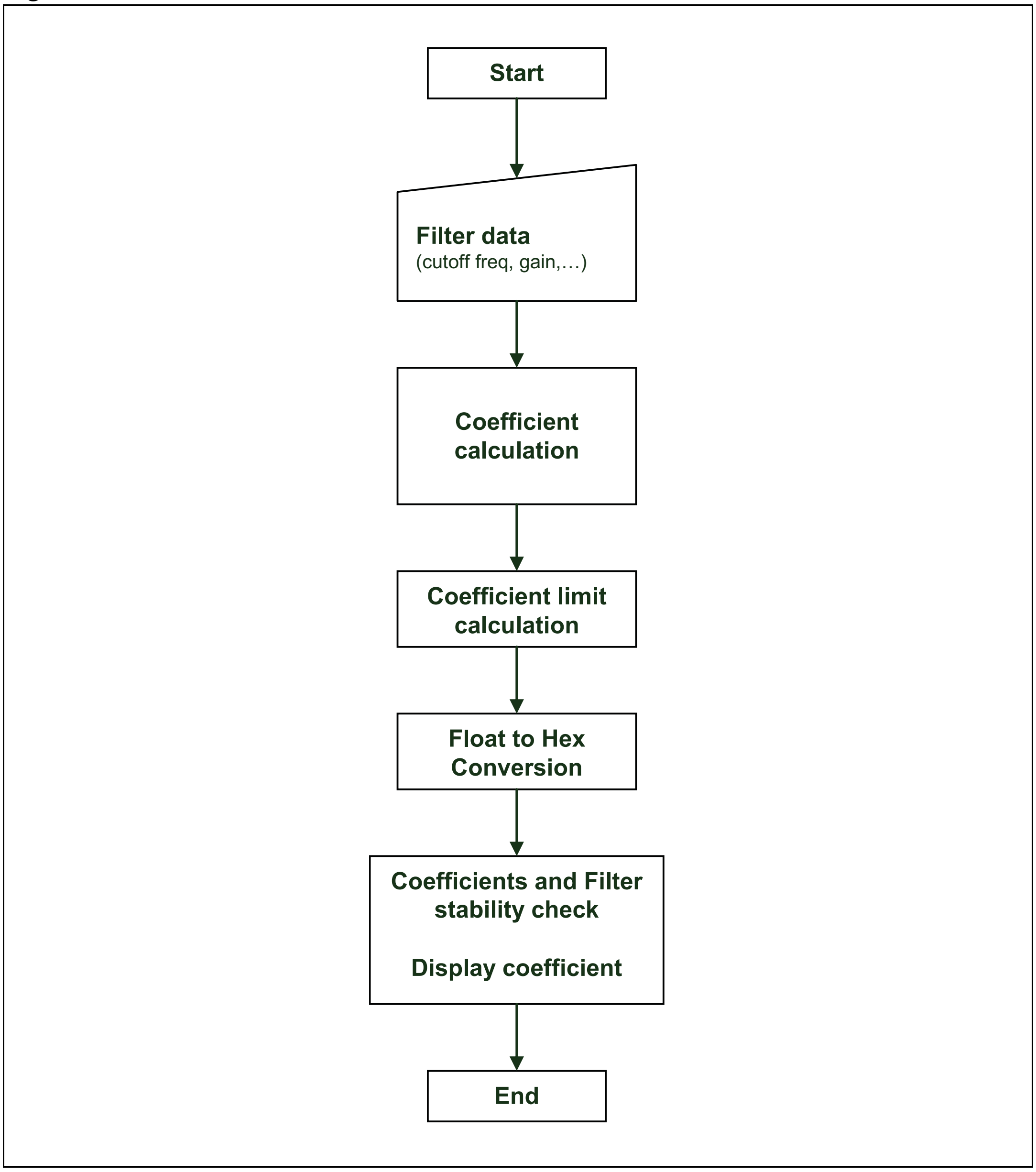
Figure 9. Band-pass filter - magnitude response



Appendix A Matlab code (functions)

A.1 Code structure

Figure 10. Code structure



A.2 Peak filter (PeakFilterAPW.m)

```
%------------------------------------------------------------------------%
%  function [Coeff_Hex, CoeffAPW, LimitVal] = PeakFilterAPW(Fc, Gain, Q,
%                                         CoeffRange, Fs)
%
%       Args:Fc -> Cutoff Frequency
%               Gain -> Gain
%               Q -> Quality factor
%               CoeffRange -> Coefficient Range (1, 2 or 4)
%               Fs -> Sample frequency
%       Return: Coeff_Hex -> APW filter Coeff  - Hex
%               CoeffAPW -> APW filter Coeff  - Floating Point
%               LimitVal -> Limit coeff value
%       Description: Generates the APWorkbench coeff for a Peak Filter
%
%   STMicroelectronics - Agrate (ITALY)
%   MSH - Audio & Sound BU
%   Revision: 1.1
%   Date: 23 June 2011
%------------------------------------------------------------------------%
%% Function code
function [Coeff_Hex, CoeffAPW, LimitValue] = PeakFilterAPW(Fc, Gain, Q, ...
    CoeffRange, Fs)
format long

if (nargin <5)
    Fs = 96000;
end

Teta = (2*pi*Fc)/Fs;    %Angle from frequency
K = tan(Teta/2);
W = K*K;

%% Process Gain

Gain = Gain* 0.115129254;
NormGain = exp(Gain);
```

```
%% Coefficint Calculation

if NormGain<1
%    Negative NormGain - Cut
    fCutValue = 1+(1/NormGain/Q)*K+W;                  % Boost/NormGain
    Coeff_4 = ((1+(1/Q)*K+W)/fCutValue)/2.0;           % b0/2
    Coeff_0 = (W-1)/fCutValue;                         % b1/2
    Coeff_1 = (1-(1/Q)*K+W)/fCutValue;                 % b2
    Coeff_3 = ((1-(1/NormGain/Q)*K+W)/fCutValue)*-1.0; % -a2
    Coeff_2 = (Coeff_0)*-1.0;                          % -a1/2
else
    %    Positive NormGain - Boost
    fBoostValue = 1+(1/Q)*K+W;                         % Boost/NormGain
    Coeff_4 = ((1+(NormGain/Q)*K+W)/fBoostValue)/2.0;  % b0/2
    Coeff_0 = (W-1)/fBoostValue;                       % b1/2
    Coeff_1 = (1-(NormGain/Q)*K+W)/fBoostValue;        % b2
    Coeff_3 = ((1-(1/Q)*K+W)/fBoostValue)*-1.0;        % -a2
    Coeff_2 =(Coeff_0)*-1.0;                           % -a1/2

end

%% Coefficient Matrix
CoeffAPW = [Coeff_0 Coeff_1 Coeff_2 Coeff_3 Coeff_4];

%% Coefficient Limit Value
LimitValue = LimitVal(CoeffRange);

%% Coefficient Matrix - Hex format
Coeff_Hex = myFloat2Hex(CoeffAPW, CoeffRange);
```

A.3 Low-pass and high-pass filter (LHPassFilterAPW.m)

```
%-------------------------------------------------------------------------%
%  function [Coeff_Hex, CoeffAPW] = LHPassFilterAPW(CutOff_Freq, Q,
%                                       FType, Order, CoeffRange, Fc)
%       Args:Fc -> Cutoff Frequency
%               Q -> Quality factor
%               FType -> 0->LowPassFilter; 1->HighPass Filter
%               Order -> 1=1st order; 2=2nd order
%               CoeffRange -> Coefficient Range (1, 2 or 4)
%               Fc -> Sample frequency
%       Return: Coeff_Hex -> APW filter Coeff  - Hex
%               CoeffAPW -> APW filter Coeff  - Floating point
%               LimitVal -> Limit coeff value
%       Description: Generates the APWorkbench coeff for a LHPassFilter
%
%   STMicroelectronics - Agrate (ITALY)
%   MSH - Audio & Sound BU
%   Revision: 1.1
%   Date: 23 June 2011
%-------------------------------------------------------------------------%
%% Function code

function [Coeff_Hex, CoeffAPW, LimitValue] = LHPassFilterAPW(Fc, Q, ...
    FType, Order, CoeffRange, Fs)
format long

if (nargin <6)
    Fs = 96000;
end

Teta = (2*pi*Fc)/Fs;    %Angle from frequency
K = tan(Teta/2);
alpha = 1+K;

a2 = 0;
b2 = 0;
a0 = 1.0;
a1 = -(1-K)/alpha;
```

```
if Order == 1 %1st Order Filter
    if FType ==0 % Low Pass Filter
        b0 = K/alpha;
        b1 = b0;
        Coeff_0 = (b1/2.0)/a0;
        Coeff_1 = (b2)/a0;          % Always =0!!!
        Coeff_2 = (-a1/2.0)/a0;
        Coeff_3 = (-a2)/a0;         % Always =0!!!
        Coeff_4 = (b0/2.0)/a0;
    else
        % High Pass Filter
        b0 = 1/alpha;
        b1 = -b0;
        Coeff_0 = (b1/2.0)/a0;
        Coeff_1 = (b2)/a0;          % Always =0!!!
        Coeff_2 = (-a1/2.0)/a0;
        Coeff_3 = (-a2)/a0;         % Always =0!!!
        Coeff_4 = (b0/2.0)/a0;
    end


else % 2nd Order Filter
    Teta = (2*pi*Fc)/Fs;    %Angle from frequency
    K = tan(Teta/2);
    W = K*K;
    DE = 1+(1/Q)*K+W;

   Coeff_3 = ((1-(1/Q)*K+W)/DE)*-1.0; % -a2
    Coeff_2 = ((W-1)/DE)*-1.0;          % -a1/2

     if FType ==0 % Low Pass Filter 2nd Order
        Coeff_4 = (W/DE)/2.0;           % b0/2
        Coeff_0 = W/DE;                 % b1/2
        Coeff_1 = W/DE;                 % b2
    else
        % High Pass Filter 2nd Order
        Coeff_4 = (1/DE)/2.0;           % b0/2
        Coeff_0 = -1/DE;                % b1/2
```

```
        Coeff_1 = 1/DE;                 % b2
     end
end


%% Coefficient Matrix
CoeffAPW = [Coeff_0 Coeff_1 Coeff_2 Coeff_3 Coeff_4];

%% Coefficient Limit Value
LimitValue = LimitVal(CoeffRange);

%% Coefficient Matrix - Hex format
Coeff_Hex = myFloat2Hex(CoeffAPW, CoeffRange);
```

A.4 Low and high-shelf filter (ShelfFilterAPW.m)

```
%-----------------------------------------------------------------------%
%  function [CoeffAPW] = ShelfFilterAPW(Fc, Gain, Slope, FType,
%                                       CoeffRange, Fs)

%       Args:Fc -> Cutoff Frequency
%               Gain -> Gain
%               Slope -> Slope
%               FType -> Filter type (Low or High Shelf)
%               CoeffRange -> Coefficient Range (1, 2 or 4)
%               Fs -> Sample frequency
%       Return: Coeff_Hex -> APW filter Coeff  - Hex
%               CoeffAPW -> APW filter Coeff  - Floating Point
%               LimitVal -> Limit coeff value
%       Description: Generates APWorkbench coeff for a Low or a High
%                   Shelf Filter
%
%   STMicroelectronics - Agrate (ITALY)
%   MSH - Audio & Sound BU
%   Revision: 1.1
%   Date: 23 June 2011
%-----------------------------------------------------------------------%
%% Function code
function [Coeff_Hex, CoeffAPW, LimitValue] = ShelfFilterAPW(Fc, Gain, ...
    Slope, FType,...
    CoeffRange, Fs)

format long
%   if FType=0 =>LowShelf
%   if FType=1 =>HighShelf

if (nargin < 6)
    Fs = 96000;
end


Teta = (2*pi*Fc)/Fs;    %Angle from frequency
SinTeta = sin(Teta);
```

```
CosTeta = cos(Teta);


%   Normalized Gain
NormGain = 10^(Gain/40);

%   alpha and beta
alpha = (SinTeta/2)*sqrt((NormGain+(1/NormGain))*(1.0/Slope-1.0)+2.0);
beta = 2*sqrt(NormGain)*alpha;



%% Coefficient Calculation

if FType == 0
%    FType = 0 => LowShelf
    b0 = NormGain*((NormGain+1)-(NormGain-1)*CosTeta + beta);
    b1 = 2*NormGain*((NormGain-1)-(NormGain+1)*CosTeta);
    b2 = NormGain*((NormGain+1)-(NormGain-1)*CosTeta - beta);
    a0 = (NormGain+1)+(NormGain-1)*CosTeta + beta;
    a1 = -2*((NormGain-1)+(NormGain+1)*CosTeta);
    a2 = (NormGain+1)+(NormGain-1)*CosTeta-beta;
else
%    FType = 1 => HighShelf
    b0 = NormGain*((NormGain+1)+(NormGain-1)*CosTeta + beta);
    b1 = -2*NormGain*((NormGain-1)+(NormGain+1)*CosTeta);
    b2 = NormGain*((NormGain+1)+(NormGain-1)*CosTeta - beta);
    a0 = (NormGain+1)-(NormGain-1)*CosTeta + beta;
    a1 = 2*((NormGain-1)-(NormGain+1)*CosTeta);
    a2 = (NormGain+1)-(NormGain-1)*CosTeta-beta;

end

%   APW Coefficient - Reworked coefficient
Coeff_0 = (b1/2.0)/a0;
Coeff_1 = (b2)/a0;
Coeff_2 = (-a1/2.0)/a0;
Coeff_3 = (-a2)/a0;
Coeff_4 = (b0/2.0)/a0;
```

```
%% Coefficient Matrix
CoeffAPW = [Coeff_0 Coeff_1 Coeff_2 Coeff_3 Coeff_4];

%% Coefficient Limit Value
LimitValue = LimitVal(CoeffRange);

%% Coefficient Matrix - Hex format
Coeff_Hex = myFloat2Hex(CoeffAPW, CoeffRange);
```

A.5 Notch filter (NotchFilterAPW.m)

```
%-------------------------------------------------------------------------%
%  function [Filter_Coeff, CoeffAPW] = NotchFilterAPW(Fc, Q, CoeffRange,
%                                            Fs)
%       Args:Fc -> Cutoff Frequency
%               Gain -> Gain
%               Q -> Quality factor
%               CoeffRange -> Coefficient Range (1, 2 or 4)
%               Fs -> Sample frequency
%       Return: Coeff_Hex -> APW filter Coeff  - Hex
%               CoeffAPW -> APW filter Coeff  - Floating Point
%               LimitVal -> Limit coeff value
%
%       Description: Generates the APWorkbench coeff for a Notch Filter
%
%   STMicroelectronics - Agrate (ITALY)
%   MSH - Audio & Sound BU
%   Revision: 1.1
%   Date: 23 June 2011
%-------------------------------------------------------------------------%
%% Function code
function [Coeff_Hex, CoeffAPW, LimitValue] = NotchFilterAPW(Fc, Q, ...
    CoeffRange, Fs)
format long

if (nargin == 3)
    Fs = 96000;
end

Teta = (2*pi*Fc)/Fs;    %Angle from frequency
SinTeta = sin(Teta);
CosTeta = cos(Teta);
alpha = SinTeta/(2*Q);

%% Coefficint Calculation

b0 = 1;
b1 = -2*CosTeta;
```

```
b2 = 1;
a0 = 1+alpha;
a1 = -2*CosTeta;
a2 = 1-alpha;

%   APW Coefficient - Reworked coefficient
Coeff_0 = (b1/2.0)/a0;
Coeff_1 = (b2)/a0;
Coeff_2 = (-a1/2.0)/a0;
Coeff_3 = (-a2)/a0;
Coeff_4 = (b0/2.0)/a0;

%% Coefficient Matrix
CoeffAPW = [Coeff_0 Coeff_1 Coeff_2 Coeff_3 Coeff_4];

%% Coefficient Limit Value
LimitValue = LimitVal(CoeffRange);

%% Coefficient Matrix - Hex format
Coeff_Hex = myFloat2Hex(CoeffAPW, CoeffRange);
```

A.6 All-pass filter (AllPassFilterAPW.m)

```
%------------------------------------------------------------------------%
%  function [Coeff_Hex, CoeffAPW] = AllPassFilterAPW(Fc, Q, CoeffRange,
%                                                        Fs)
%       Args:Fc -> Cutoff Frequency
%               Q -> Quality factor
%               CoeffRange -> Coefficient Range (1, 2 or 4)
%               Fs -> Sample frequency
%       Return: Coeff_Hex -> APW filter Coeff  - Hex
%               CoeffAPW -> APW filter Coeff  - Floating Point
%               LimitVal -> Limit coeff value
%
%       Description: Generates the APWorkbench coeff for a All Pass Filter
%
%   STMicroelectronics - Agrate (ITALY)
%   MSH - Audio & Sound BU
%   Revision: 1.1
%   Date: 23 June 2011
%------------------------------------------------------------------------%
%% Function code
function [Coeff_Hex, CoeffAPW, LimitValue] = AllPassFilterAPW(Fc, Q, ...
    CoeffRange, Fs)
format long

if (nargin < 4)
    Fs = 96000;
end

Teta = (2*pi*Fc)/Fs;    %Angle from frequency
SinTeta = sin(Teta);
CosTeta = cos(Teta);
alpha = SinTeta/(2*Q);

%% Coefficint Calculation

b0 = 1-alpha;
b1 = -2*CosTeta;
b2 = 1+alpha;
```

```
a0 = 1+alpha;
a1 = b1;
a2 = 1-alpha;

%   APW Coefficient - Reworked coefficient
Coeff_0 = (b1/2.0)/a0;
Coeff_1 = (b2)/a0;
Coeff_2 = (-a1/2.0)/a0;
Coeff_3 = (-a2)/a0;
Coeff_4 = (b0/2.0)/a0;

%% Coefficient Matrix
CoeffAPW = [Coeff_0 Coeff_1 Coeff_2 Coeff_3 Coeff_4];

%% Coefficient Limit Value
LimitValue = LimitVal(CoeffRange);

%% Coefficient Matrix - Hex format
Coeff_Hex = myFloat2Hex(CoeffAPW, CoeffRange);
```

A.7 Band-pass filter (BandPassFilterAPW.m)

```
%-------------------------------------------------------------------------%
%  function [Filter_Coeff, CoeffAPW] = BandPassFilterAPW(Fc, Q, CoeffRange,
%                                           Fs)
%       Args:Fc -> Cutoff Frequency
%               Gain -> Gain
%               Q -> Quality factor
%               CoeffRange -> Coefficient Range (1, 2 or 4)
%               Fs -> Sample frequency
%       Return: Coeff_Hex -> APW filter Coeff  - Hex
%               CoeffAPW -> APW filter Coeff  - Floating Point
%               LimitVal -> Limit coeff value
%
%       Description: Generates the APWorkbench coeff for a Band Pass Filter
%
%   STMicroelectronics - Agrate (ITALY)
%   MSH - Audio & Sound BU
%   Revision: 1.1
%   Date: 23 June 2011
%-------------------------------------------------------------------------%
%% Function code
function [Coeff_Hex, CoeffAPW, LimitValue] = BandPassFilterAPW(Fc, Gain,...
    Q, CoeffRange, Fs)
format long

if (nargin == 3)
    Fs = 96000;
end

Teta = (2*pi*Fc)/Fs;    %Angle from frequency
SinTeta = sin(Teta);
CosTeta = cos(Teta);
alpha = SinTeta/(2*Q);

NormGain = 10^(Gain/20);

%% Coefficint Calculation
```

```
b0 = alpha*NormGain;
b1 = 0;
b2 = -b0;
a0 = 1+alpha;
a1 = -2*CosTeta;
a2 = 1-alpha;

%   APW Coefficient - Reworked coefficient
Coeff_0 = (b1/2.0)/a0;
Coeff_1 = (b2)/a0;
Coeff_2 = (-a1/2.0)/a0;
Coeff_3 = (-a2)/a0;
Coeff_4 = (b0/2.0)/a0;

%% Coefficient Matrix
CoeffAPW = [Coeff_0 Coeff_1 Coeff_2 Coeff_3 Coeff_4];

%% Coefficient Limit Value
LimitValue = LimitVal(CoeffRange);

%% Coefficient Matrix - Hex format
Coeff_Hex = myFloat2Hex(CoeffAPW, CoeffRange);
```

A.8 Float to hex conversion (myFloat2Hex.m)

```
%----------------------------------------------------------------------%
%  function [floatN] = myFloat2Hex(hexN, range)
%       Args:hexN -> hexadecimal number to be converted in string format
%                       without the 0x, i.e. 0x123456 => '123456' (24 bits
%                       only)
%               range -> coefficients range 4, 2, 1
%       Return: floatN -> floating point notation number
%
%       Description: converts a fixed point hexadecimal number into a
%                    floating point one
%
%   STMicroelectronics - Agrate (ITALY)
%   MSH - Audio & Sound BU
%   Revision: 1.1
%   Date: 23 June 2011
%----------------------------------------------------------------------%


function [hexN] = myFloat2Hex(floatN, range)

format long

quantizerSetup.mode = 'fixed';
% quantizerSetup.roundmode = 'nearest';
quantizerSetup.roundmode = 'ceil';
quantizerSetup.overflowmode = 'saturate';

%Quantizer to translate from hex to num
if(range == 1)
    quantizerquantizerSetup.format = ([24 23]);
elseif(range == 2)
    quantizerquantizerSetup.format = ([24 22]);
elseif(range == 4);
    quantizerquantizerSetup.format = ([24 21]);
end

q = quantizer(quantizerquantizerSetup);
```

```
hexN = num2hex(q,floatN);
```

A.9 Max coefficient limit value calculator (LimitVal.m)

```
%----------------------------------------------------------------------%
%  function [LimitValue] = LimitVal(CoeffRange)
%
%       Args:CoeffRange -> APW filter Coeff Range
%
%       Return: LimitValue -> APW filter limit value
%
%       Description: From the CoeffRange it calculates the LimitValue
%
%   STMicroelectronics - Agrate (ITALY)
%   MSH - Audio & Sound BU
%   Revision: 1.1
%   Date: 23 June 2011
%----------------------------------------------------------------------%
%% Function code

function [LimitValue] = LimitVal(CoeffRange)
format long
switch CoeffRange
    case 1  % Coefficient +/- 1
        LimitValue = 0.99999;

   case 2  % Coefficient +/- 2
        LimitValue = 1.99999;

    case 4  % Coefficient +/- 4
        LimitValue = 3.99999;
end
```

A.10 Display coefficient and error messages (Display_Coeff.m)

```
%-----------------------------------------------------------------------%
%  function []=Display_Coeff(Filter_Coeff, CoeffAPW, LimitValue)
%       Args:Filter_Coeff -> APW filter Coeff  - Hex
%               CoeffAPW -> APW filter Coeff  - Floating
%               LimitVal -> Limit coeff value
%       Return: Display -> APW filter Coeff  - Hex
%
%       Description: Display Filter Coefficient (HEX)
%
%   STMicroelectronics - Agrate (ITALY)
%   MSH - Audio & Sound BU
%   Revision: 1.1
%   Date: 23 june 2011
%-----------------------------------------------------------------------%

function  []=Display_Coeff(Filter_Coeff, CoeffAPW, LimitValue)

a0 = 1;
a1 =-2*CoeffAPW(3);
a2 = -CoeffAPW(4);
b0 =2*CoeffAPW(5);
b1 =2*CoeffAPW(1);
b2 = CoeffAPW(2);

Coeff = [b0 b1 b2 a0 a1 a2];

%% Check for stability and Limit
Error = 0;

if ((abs(b0)>=LimitValue) || (abs(b1)>=LimitValue)|| (abs(b2)>=LimitValue))
   Error = 1;
end

if abs(a2)>1 && (abs(a1)>1+a2)
    Error = 2;
end
```

```
counter = 1;
while counter<=5
    if imag(CoeffAPW(counter))~=0
        Error = 3;
    end
   counter = counter+1;
end



% Filter coefficiners or Error message.
switch Error
    case 0  % No error
        h=fvtool(Coeff(1:3),Coeff(4:6));

        disp('   ');
        disp('   ');

disp('*****************************************************************************
*****************');
        disp('                                  Filter Coefficients');

disp('*****************************************************************************
*****************');
        Label = {'Coeff 1: b1/2', 'Coeff 2: b2', 'Coeff 3: -a1/2', 'Coeff 4: -a2',
'Coeff 5: b0/2'};
        TABLE_data = {Filter_Coeff(1,:)  Filter_Coeff(2,:) Filter_Coeff(3,:) ...
            Filter_Coeff(4,:) Filter_Coeff(5,:)};
        % TABLE_data = num2cell(TABLE_data);
        TABLE = [Label; TABLE_data];
        disp (TABLE);

    case 1  % The coefficient range must be changed
        disp('   ');
        disp('   ');

disp('*****************************************************************************
*****************');
        disp('                                  Error!!!');
        disp('                        The coefficient range must be increased');

disp('*****************************************************************************
*****************');
```

```
    case 2  % The filter is not stable
        disp('   ');
        disp('   ');
disp('******************************************************************************
*****************');
        disp('                      Error!!! The Filter is not stable!');
        disp('                      Please check the filter parameters');
disp('******************************************************************************
*****************');

    case 3  % A CoeffAPW coeff is not real
        disp('   ');
        disp('   ');
disp('******************************************************************************
*****************');
        disp('                                  Error!!!');
        disp('                      Please check the filter parameters');
disp('******************************************************************************
*****************');
end
```

Appendix B Abbreviations and acronyms

The abbreviations and acronyms used throughout this application note are defined as follows:

- fc: cutoff frequency
- fs: sampling frequency
- Q: filter quality factor
- G: gain
- LPF: low-pass filter
- HPF: high-pass filter
- LSF: low-shelf filter
- HSF: high-shelf filter
- APF: all-pass filter
- BPF: band-pass filter

6 Revision history

Table 1. Document revision history

Date	Revision	Changes
26-Sep-2011	1	Initial release.