



Designing an interactive EMG RMS envelope measurement application with the ASN Filter Designer

Application note (ASN-AN024)

January, 2016 (Rev 1)

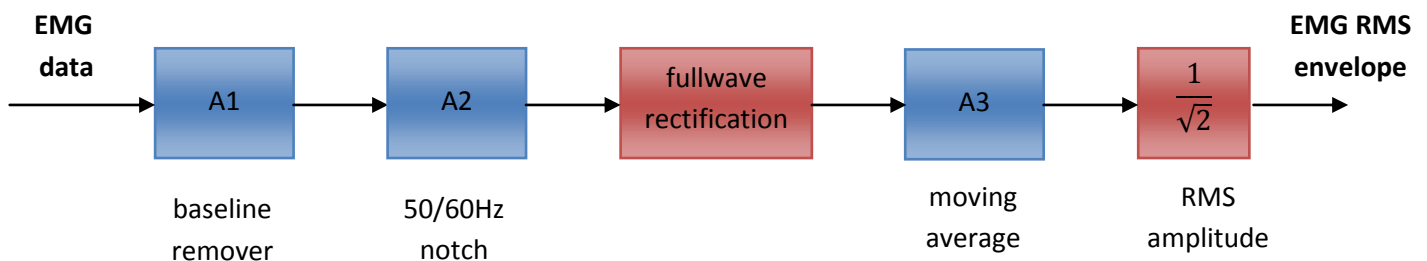
SYNOPSIS

Electromyography (EMG) is an experimental biomedical engineering technique for evaluating and recording the electrical activity produced by skeletal muscles. The EMG is performed using an instrument called an electromyograph, where the electromyograph detects the electrical potential generated by muscle cells when these cells are electrically or neurologically activated. The captured signals can be analysed by signal processing techniques in order to detect medical abnormalities and to analyse the biomechanics of human movement.

This application note demonstrates the design of a suitable EMG filtering chain using the ASN filter designer's Filter Script language and its interactive customisation for an RMS (root-mean-square) envelope measurement signal processing application.

INTRODUCTION

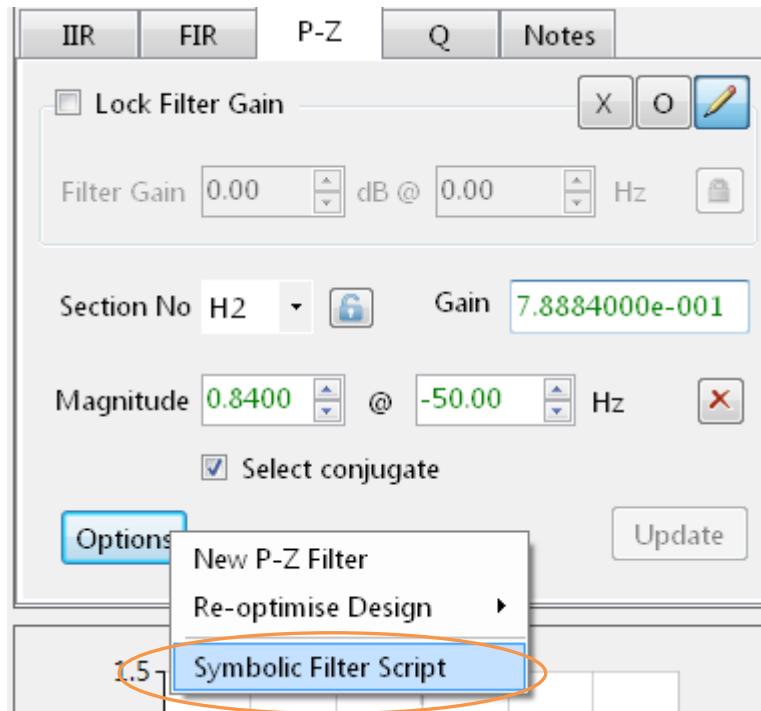
The standard building blocks required for an EMG RMS envelope measurement signal processing application are shown below:



Building block	Equation	Description
A1	$\frac{1 - z^{-1}}{1 - \alpha z^{-1}}$	A baseline removal filter (DC removal) can be realised with a simple first order pole-zero filter. Where, α is used to set the highpass filter's bandwidth.
A2	$\frac{1 - 2 \cos w_c z^{-1} + z^{-2}}{1 - 2r \cos w_c z^{-1} + r^2 z^{-2}}$	A notch filter. Where, $w_c = \frac{2\pi f_c}{f_s}$ controls the centre frequency, f_c (50 or 60Hz) of the notch, and r controls the bandwidth of the notch.
Fullwave rectification	abs()	Abs() function.
A3	$1 + z^{-1} + z^{-2} + \dots + z^{-M}$	An Mth order moving average filter.

SYMBOLIC MATH FILTER SCRIPTING LANGUAGE

As seen, there are three filters and two mathematical operations. Combining the A1 and A2 filters into one IIR filter (H2), we can implement the above operation in the ASN Filter designer with the Symbolic Filter Scripting language.



The scripting language supports over 40 scientific commands and provides designers with a familiar and powerful programming language, while at the same time allowing them to implement complex symbolic mathematical expressions for their filtering applications. The scripting language offers the unique and powerful ability to modify parameters on the fly with the so called interface variables, allowing for real-time updates of the resulting frequency response. This has the advantage of allowing the designer to see how the coefficients of the symbolic transfer function expression affect the frequency response and the filter's time domain dynamic performance.

Please refer to the ASN Filter Designer Filter Script user's guide ([ASN15-DOC002](#)) for more information.

FILTER SCRIPT

The `conv()` function merges the A1 and A2 filters¹ in order to produce a new filter, H2 (H1 is disabled for the implementation - see the [implementation summary](#)). The filter scripting language supports several constants, such as: `Twopi` (2π) and `fs` (the sampling rate), which ease the implementation.

```
// A1 filter
b={1,-1}; // define numerator coefficients
a={1,-alpha}; // define denominator coefficients

// A2 filter
wc=Twopi*fc/fs;
Num = {1,-2*cos(wc),1}; // define numerator coefficients
Den = {1,-2*r*cos(wc),r^2}; // define denominator coefficients

Num=conv(Num,b); // merge A1 and A2 into "one" new filter
Den=conv(Den,a);
```

The new filter now requires a new gain value in order to provide 0dB at Nyquist ($F_s/2$).

```
// set gain of filter to 0dB at Nyquist
Twiddles_at_nyquist=exp(-i*series(0,1,3)*pi);
NumGain=sum(Num.*Twiddles_at_nyquist); // dft on Num @ Fs/2
DenGain=sum(Den.*Twiddles_at_nyquist); // dft on Den @ Fs/2
Gain=abs(DenGain/NumGain);
```

The complete Filter Script becomes (including interface variables):

```
ClearH1; // clear primary filter from cascade
interface r = {0,1,0.01,0.9}; // radius range
interface fc = {0, fs/2,fs/100,fs/4}; // centre frequency range
interface alpha = {0.8, 1,0.01,.9};

Main()

// A1 filter
b={1,-1}; // define numerator coefficients
a={1,-alpha}; // define denominator coefficients

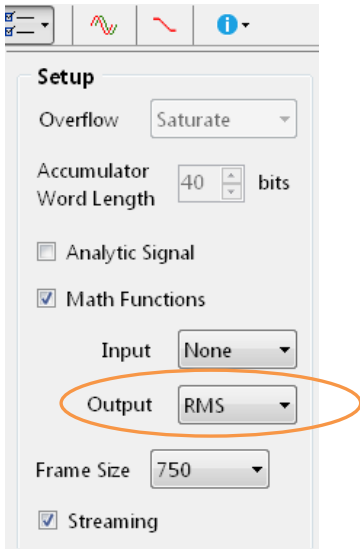
// A2 filter
wc=Twopi*fc/fs;
Num = {1,-2*cos(wc),1}; // define numerator coefficients
Den = {1,-2*r*cos(wc),r^2}; // define denominator coefficients

Num=conv(Num,b); // merge A1 and A2 into "one" new filter
Den=conv(Den,a);

// set gain of filter to 0dB at Nyquist
Twiddles_at_nyquist=exp(-i*series(0,1,3)*pi);
NumGain=sum(Num.*Twiddles_at_nyquist); // dft on Num @ Fs/2
DenGain=sum(Den.*Twiddles_at_nyquist); // dft on Den @ Fs/2
Gain=abs(DenGain/NumGain);
```

¹ `conv()` in this context is the same as performing an algebraic multiplication of the two filter polynomials.

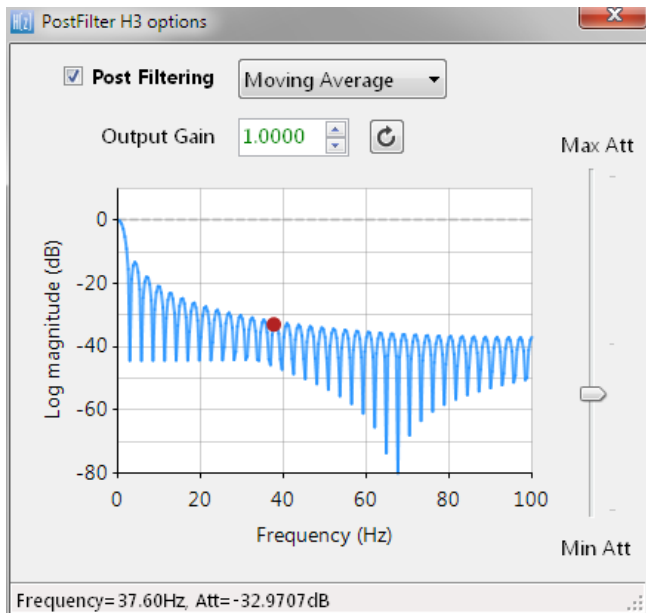
MATHEMATICAL FUNCTIONS




The fullwave rectification and $\frac{1}{\sqrt{2}}$ mathematical operations can be summarised as one RMS () operation, i.e. $\frac{abs()}{\sqrt{2}}$ and implemented via the Output Math Function in the signal analyser. This scaling trick is made possible as H3 is a linear filter (see [post filtering](#)), although this operation is equivalent to using abs () with an output Gain of 0.707.

The tool supports the following mathematical functions: abs, angle, ln, RMS, sqr and sqrt.

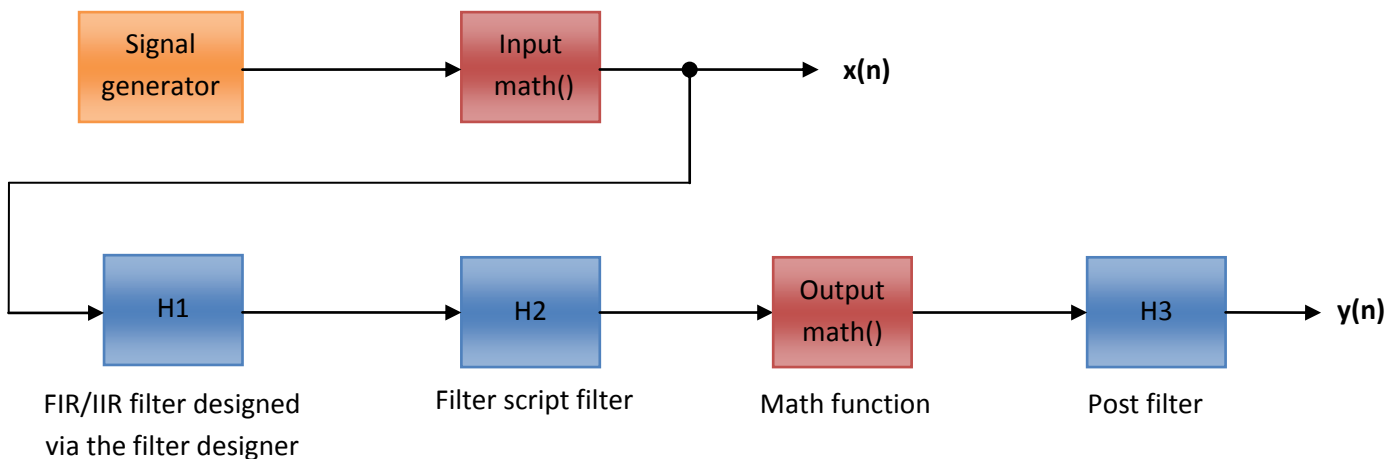
POST FILTERING



The ASN filter designer's signal analyser implements an extra post filter, H3. Unlike the H1 and H2 filters, the H3 filter is *always lowpass* and is preceded by an optional mathematical function operation ([see above](#)).

For the application considered herein, the H3 moving average filter can be implemented in the signal analyser's H3 post filter, .

The ASN filter designer's signal analyser complete filtering chain is shown below together with the signal generator and the input/output math function blocks.



The Input math () and H1 filter blocks are disabled for the application considered herein.

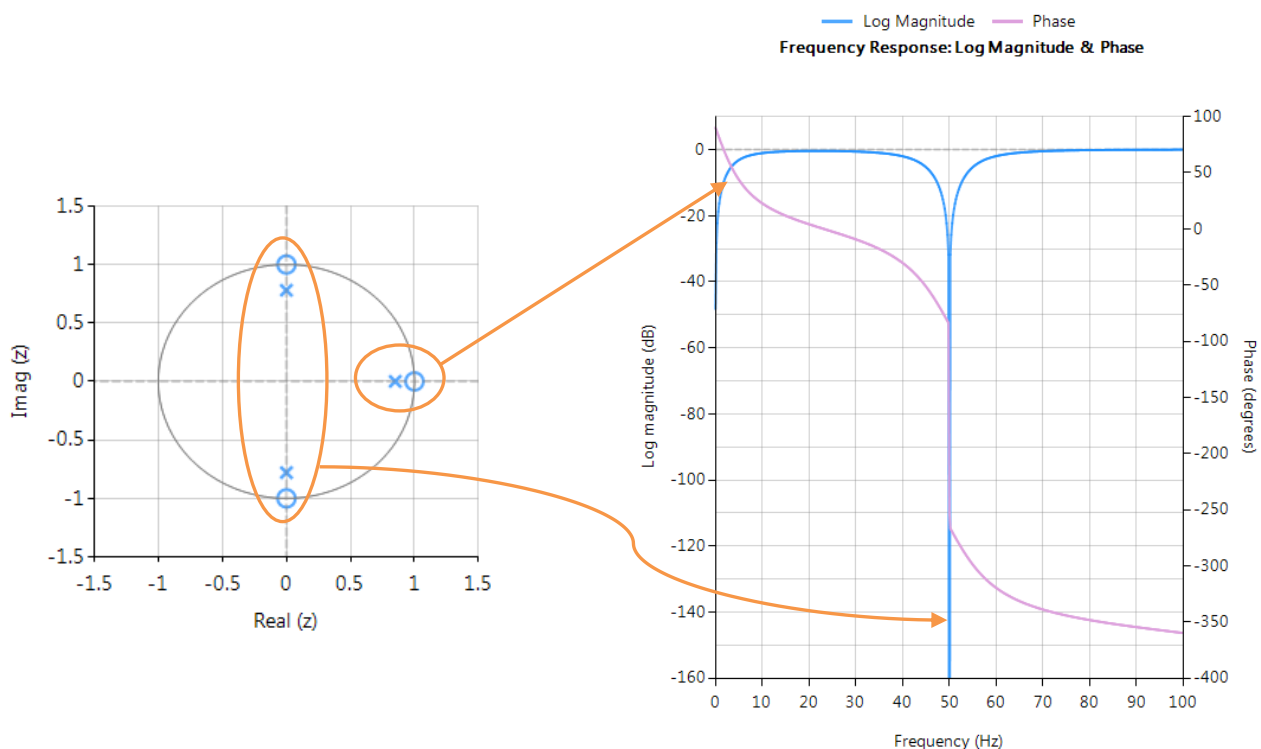
IMPLEMENTATION SUMMARY


Implementation Summary

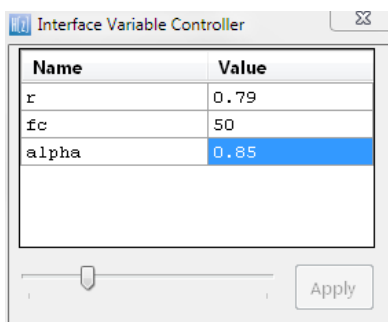
- Sampling Rate (F_s) = 200Hz
- Filter Arithmetic = Double Precision
- Input Math Function = None
- H1 (---) = Disabled
- H2 (IIR) = Direct Form II Transposed
- Output Math Function = RMS
- H3 (FIR) = Transposed Direct Form
- H3 (Output Gain) = 1

P-Z CHART AND FREQUENCY RESPONSE

Upon running the Filter Script, we obtain the following P-Z chart and frequency response.



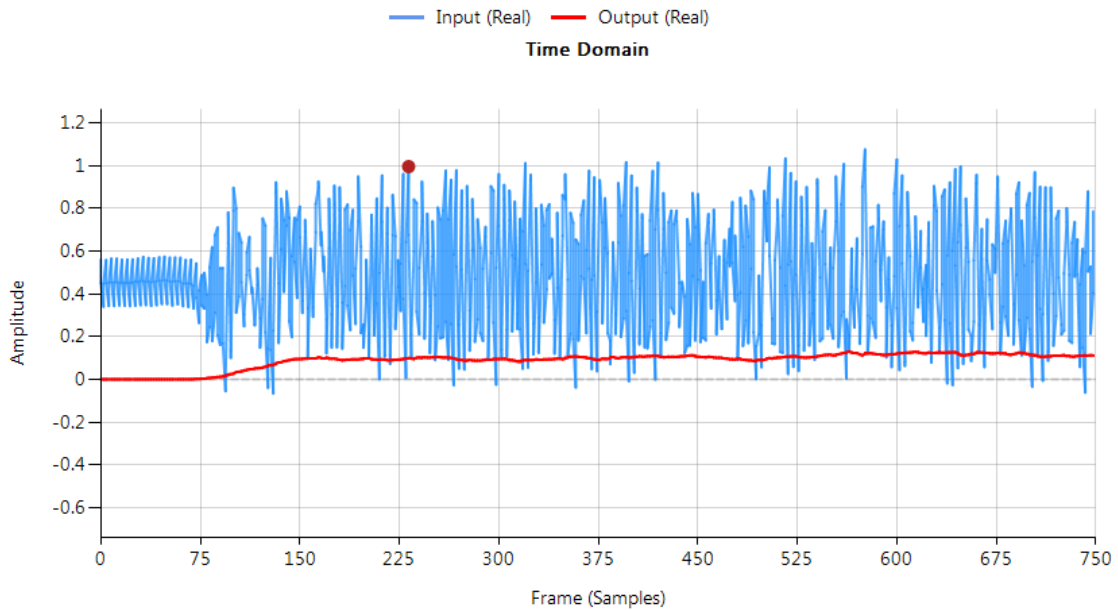
As seen, the two filters have been combined correctly. You may alter the filter parameters interactively by using the signal analyser's Interface Variable Controller UI, 



EXAMPLE PROJECT

An example project file (`\Projects\emg_ex.afd`) is provided with the tool. This project file implements the concepts described herein and uses an example EMG datafile (`\Datafiles\EMGSensorData.txt`, which may be loaded into the signal generator under the `Data File` option) with an additive 50Hz sinusoid. Depending on your installation directory structure, it may be necessary to find the `\Datafiles` directory manually and re-load the datafile.

The chart shown below demonstrates the complete signal processing operation. Where, it can be seen that the filtering operation has nicely eliminated the baseline offset and the effects of the 50Hz interference and produced an accurate estimate of the RMS amplitude of the EMG signal's envelope.



EXPORT TO MATLAB AND SCILAB

```
Filter Summary
Matlab/Octave
ASNFD.Arithmetic = 'Floating Point (Double Precision)';
ASNFD.Architecture = 'IIR';
ASNFD.Structure = 'Direct Form II Transposed';
ASNFD.Response = 'Lowpass';
ASNFD.Method = 'User Defined';
ASNFD.Biquad = 'Yes';
ASNFD.Stable = 'Yes';
ASNFD.Fs = 2.000000e+002;
ASNFD.Order = 5;
ASNFD.SOS=[];
ASNFD.H2Numerator = [ 0.9135460000,-0.9135460000, 0.9135460000,-0.9135460000, 0.9135460000];
ASNFD.H2Denominator = [ 1.0000000000,-0.9400000000, 0.8836000000,-0.8836000000, 1.0000000000];
ASNFD.H3Enabled = 'Yes';
ASNFD.H3PreMathFunction = 'RMS';
ASNFD.H3Numerator = [ 0.01408450704, 0.01408450704, 0.01408450704, 0.01408450704, 0.01408450704];
ASNFD.H3Denominator = [ 0.01408450704, 0.01408450704, 0.01408450704, 0.01408450704, 0.01408450704];
```

The complete design may be exported to Matlab or Scilab for further evaluation with the 3rd party frameworks. A version for both platforms of the example considered herein is available for evaluation:

`\Matlab\EMGDataDemo.m`

`\Scilab\EMGDataDemo.sce`

Please refer to the ASN filter designer's user guide ([ASN15-DOC001](#)) for details pertaining to the Matlab and Scilab frameworks.

Document Revision Status

Rev.	Description	Date
1	Document reviewed and released.	21/01/2016

Support and product details

- ▶ ASN Filter Designer product home page: http://www.advsolned.com/asn_filter_designer.html
- ▶ ASN Technical support: support@advsolned.com