ASN Filter Designer v5.x
Filter designer UI user's guide

April 2024
ASN21-DOC015, Rev. 10

For public release

Legal notices

Advanced Solutions Nederland B.V.
www.advsolned.com
support@advsolned.com

Technical documentation feedback

If you would like to make a suggestion or report an error in our documentation, please feel free to contact us. You are kindly requested to provide as much information as possible, including a full description of the error or suggestion, the page number and the document number/description. All suggestions or errors may be sent to: support@advsolned.com

1. Introduction

Thank you for your interest in the ASN Filter Designer. This product is available in the following three flavours:

	Licence type	Demo	Educational	Professional
Filter design	Max IIR filter order (*design method*)	20	20	100
	Max FIR filter order (*design method*)	64	200	499
	Max num poles/zeros (*design method + script*)	100	200	500
	Max all-pass filters	2 biquads	4 biquads	10 biquads
	FIR Multiband	4 bands	5 bands	8 bands
GUI functions	Save project	✗	✓	✓
	Save analyser data	✗	✓	✓
	Export to Excel	✗	✓	✓
	Export charts	✗	✓	✓
Automatic code generation	Documentation	Specification only	✓	✓
	Matlab, Python, Octave and Scilab	✗	✓	✓
	Arm CMSIS-DSP, C, C# and Xilinx	✗	✓	✓
	Fast Arm CMSIS-DSP	✗	✗	✓
ASN FilterScript	Max interface variables	3	6	20
	IIR design methods (max filter order)	10	10	20
	FIR design methods	64	200	499
	Laplace transforms (analog)	✓	✓	✓
	Ask the DSP Expert	✗	✗	✓
	Licence	non-commercial use	non-commercial use	commercial use

- Licences are perpetual licences or subscription based.

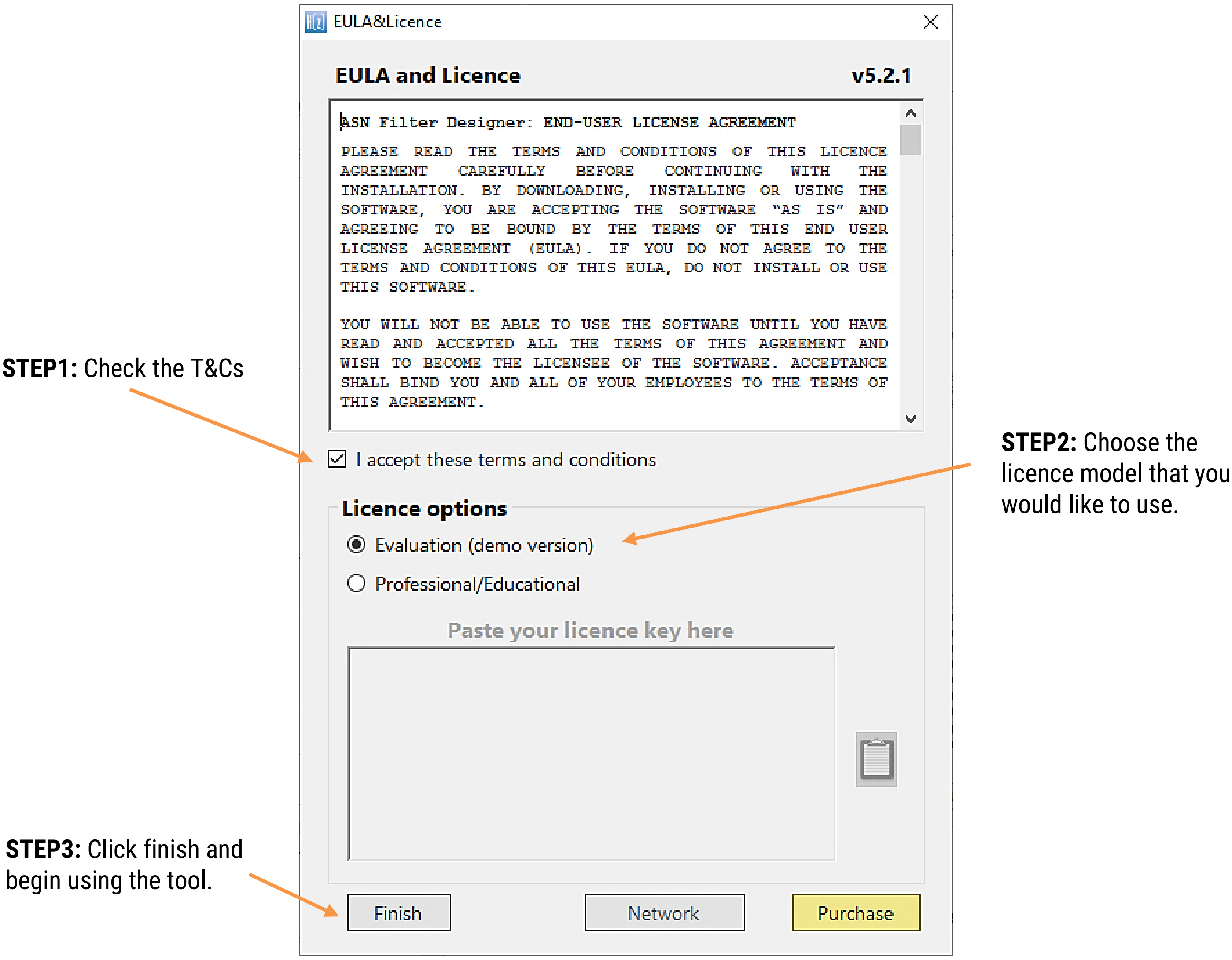
Please contact us at sales@advsolned.com if you would like to buy a licence for the educational or professional edition. We also offer academic and group licensing discounts, so please ask!

1.1. Getting started

The ASN filter designer has been designed around Microsoft's .NET technology, and as such requires .NET framework 4.0 to be installed before installation of the ASN Filter designer can continue. Where, the currently supported operating systems are: Windows 11, 10, Windows 8, Windows 7 and Windows Vista.

As most modern versions of Windows will already have .NET 4.0 pre-installed, no action will usually be required. However, in the unlikely event that it is not installed, you may download the .NET 4.0 client framework here.

1.1.1. EULA

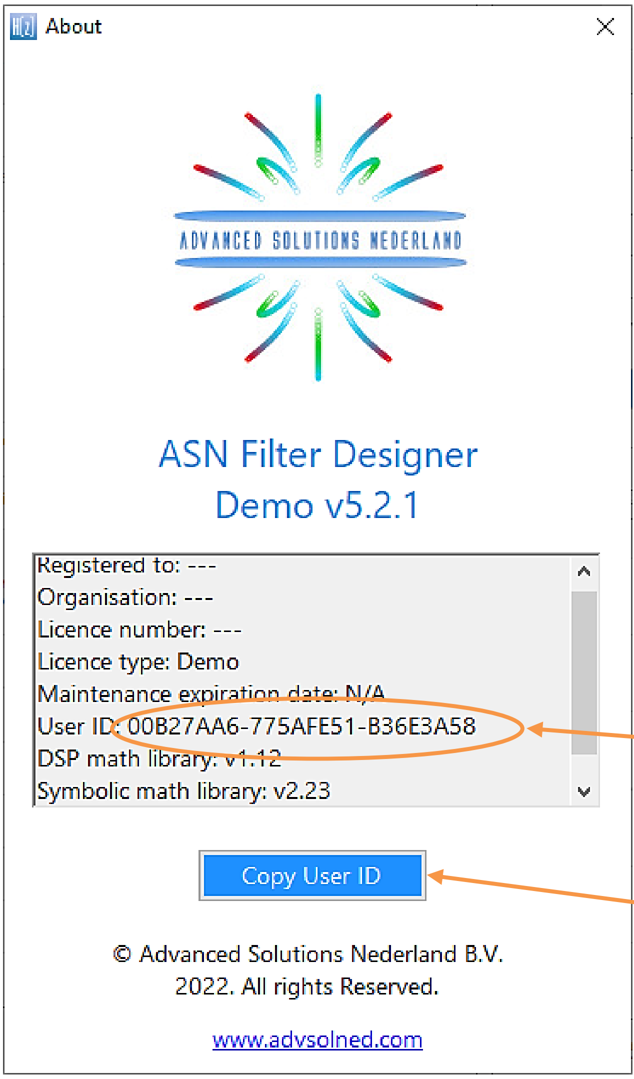


Your licence details may be viewed inside the GUI via the **Help > About** menu.

1.1.2. User ID

All licences are issued based on non-editable computer details, such as the CPU serial number and are intended for use on a single computer only. A unique **User ID** is generated and displayed in the **About** box. Before purchasing a Professional or Educational licence this code must be sent to ASN Support for generation of the licence.

Help > About



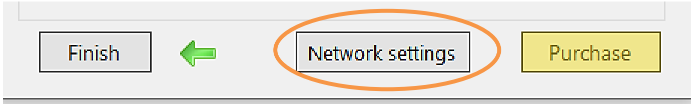
User ID

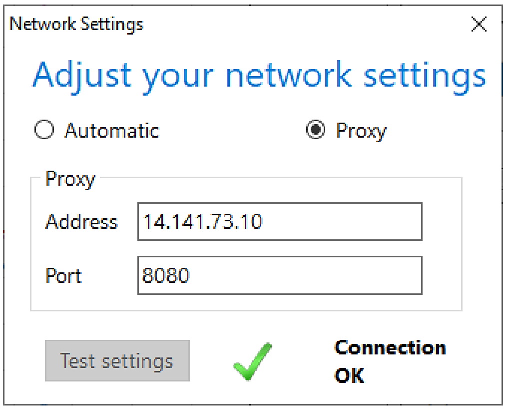
Click the **Copy User ID** button to copy the User ID to clipboard and send this to ASN for licence generation.

A processing fee may be levied for the regeneration of a lost licence or transfer to another computer.

1.1.3. Network settings

The tool supports Proxy servers. An internet connection is required in order to validate your licence file.



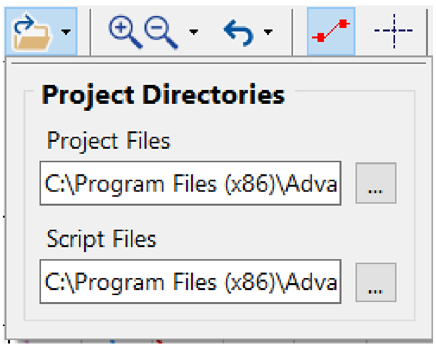


As seen, a valid **Address** and **Port** number are required. The licence manager will default to **Automatic** if no settings are found.

You may also modify your Proxy settings via the **Help > Network settings** menu.

1.1.4. User directories

After finishing the installation, it is advisable to set up your project directories:



Project Files: This is the default location of where all design project files are stored. The tool is shipped with several example project files, but you may modify this location to suit your needs.

Script Files: This is the default location of all ASN FilterScript files.

These settings will be automatically saved when the GUI closes.

If you are evaluating the software, this step may be skipped.

1.1.4.1. Other important directories

Directory name	Description
\Datafiles	Default location for external data files to be loaded into the signal analyser.
\Python	Python software development framework.
\Matlab	Matlab software development framework.
\Scilab	Scilab software development framework.
\Arm	Arm uVision project examples.
\ANSIC	ANSI C software development framework.
\CSharp	C# .NET software development framework.

The software development frameworks allow users to quickly and easily import and integrate filters designed within the ASN Filter Designer into 3rd party applications, such as an algorithm within Matlab. The software frameworks are discussed in section 5.2.3.

1.1.5. Computer requirements

Processor: The high performance DSP libraries are based around Intel's MKL technology, which requires an Intel processor in order to achieve optimal performance. Although the tool will run on other types of processors, the performance will not be optimal and may in some cases lead to sluggish performance. Therefore, an Intel processor with a system passmark benchmark of at least 1500 is recommended.
Please see http://www.cpubenchmark.net for more information.

Screen: **A screen size of at least 14 inches is recommended**, but the UI will be automatically scaled for smaller screen sizes.

Mouse: any windows compatible mouse with a mouse wheel (required for zooming).

1.1.6. Technical references

This user's guide is intended as a concise reference guide, and assumes that the reader has a firm grasp of signal processing techniques. For any readers looking for background material, please consult the following references:

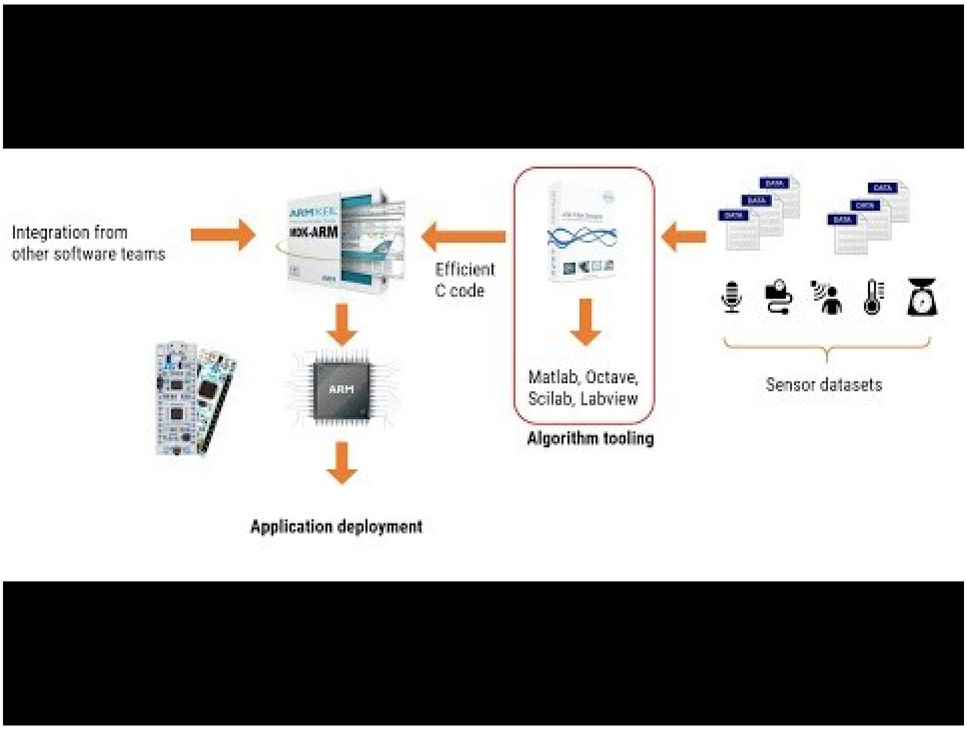
- Digital signal processing: principles, algorithms and applications, J.Proakis and D.Manoloakis
- Digital signal processing: a practical approach, E.Ifeachor and B.Jervis.
- Digital signal processing and signal processing, L.Jackson.
- Understanding digital signal processing, R. Lyons.

1.1.7. Product updates

Licensed users with valid maintenance will be automatically directed to the ASN website to download the latest version.

1.1.8. Product overview and getting started videos

The ASN DSP IIR/FIR digital filter designer.. It's so awesome!



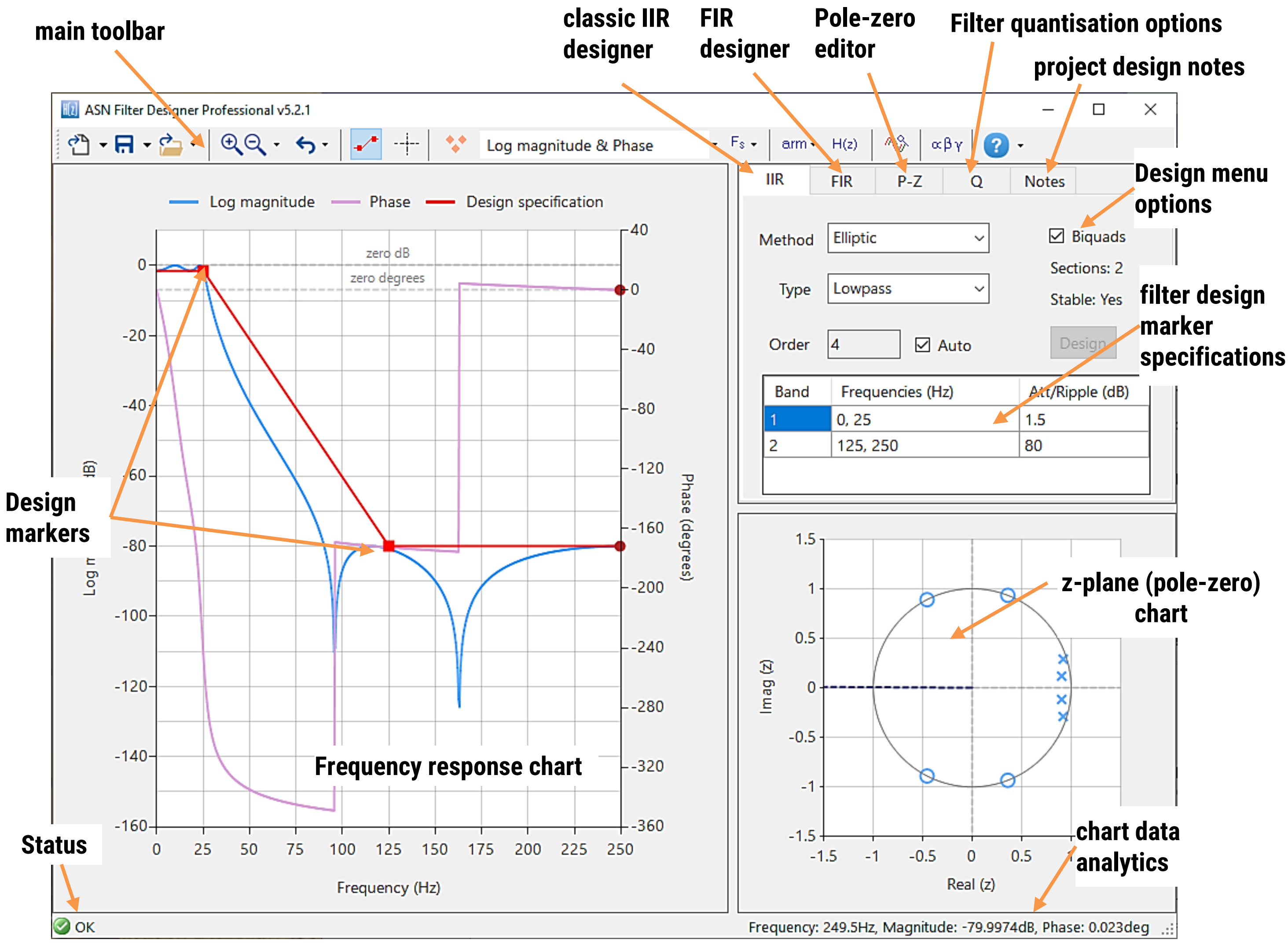
The ASN Filter Designer – Learn the basics

1.1.9. Coaching tips

The tool provides users with very detailed help, in the form of coaching tooltips. You may enable these tips via the help menu, i.e. **Help > Show/Hide coaching tips**.

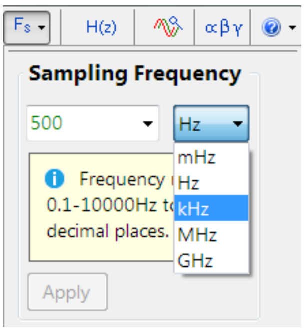
2. The filter designer UI

The main filter designer UI is show below.



2.1. Setting the sampling frequency (Fs)

Before embarking upon any design, it is recommended to set the **Sampling Frequency, Fs**. Note that the specified sampling frequency is used for all filters and the signal generator.



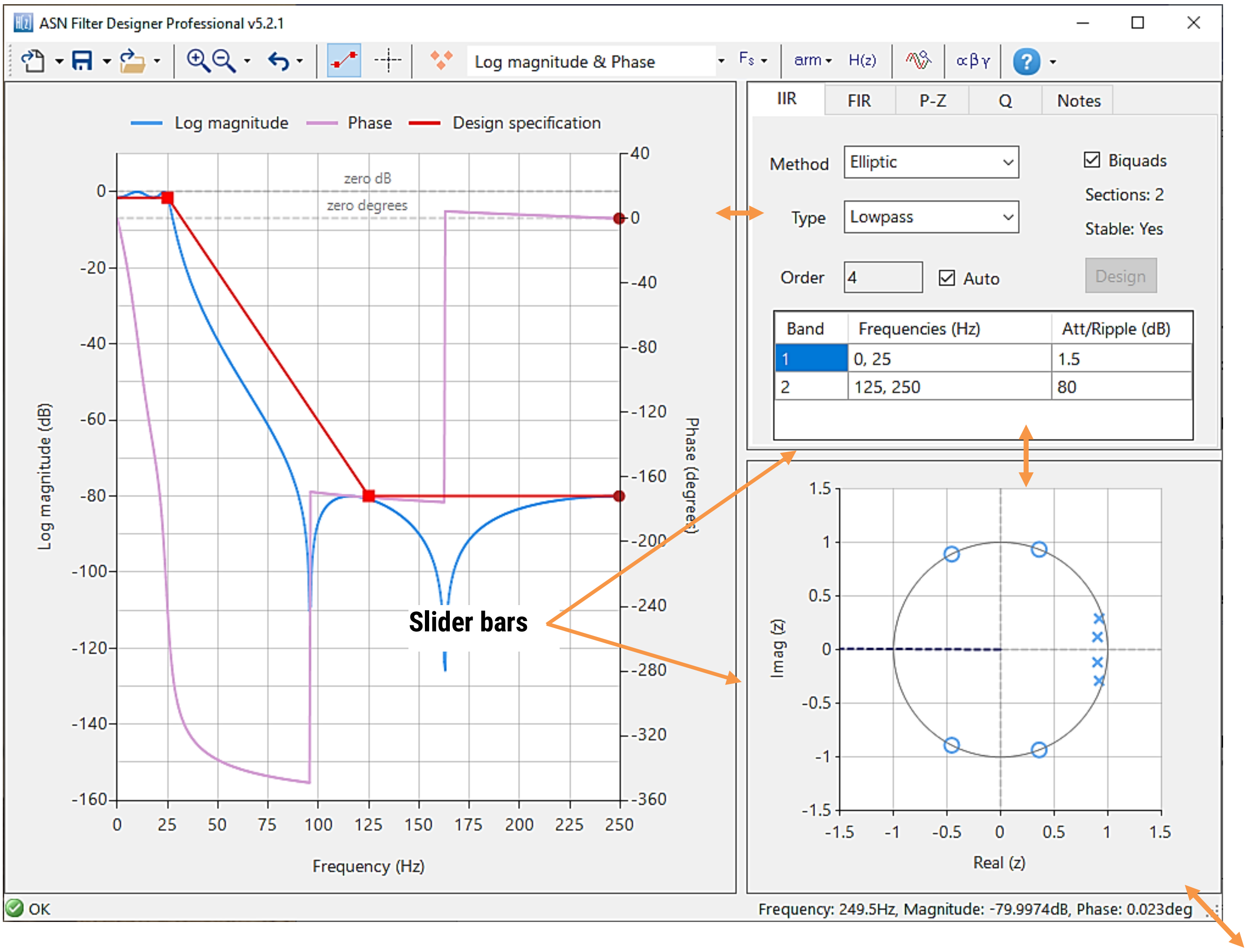
The information textbox will offer advice regarding valid frequency ranges.

The sampling frequency may be specified up to 4 decimal places, which is useful for designing filters based on fractional sampling frequencies, such as multiples of the 44.1kHz audio standard. Common examples include audio interpolation filters: 44.1kHz × 128 = 5.6448MHz and 44.1kHz × 256 = 11.2896MHz.

Changing the sampling frequency will delete all poles and zeros and reset the design to its default settings. Therefore, ensure that *you set the correct sampling frequency* before customising your design!

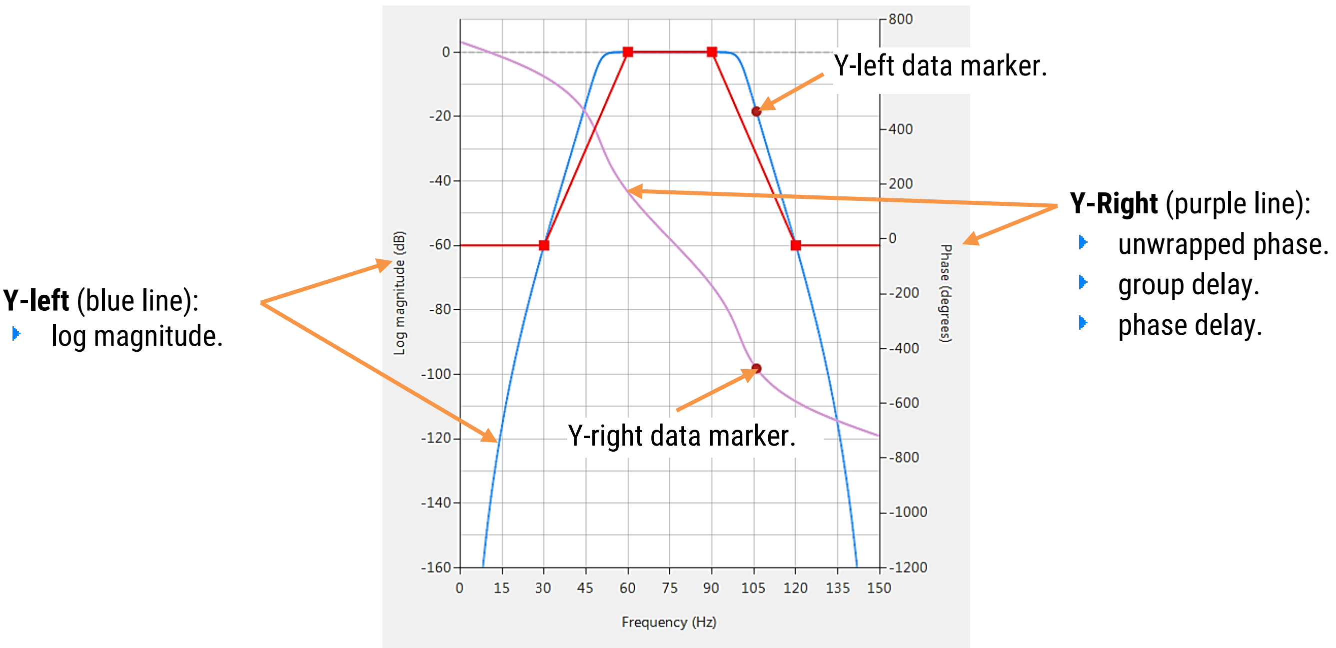
2.2. Resizing the charts

Use the slider bars to resize the design menu area, the P-Z (pole-zero) chart or the frequency response chart.



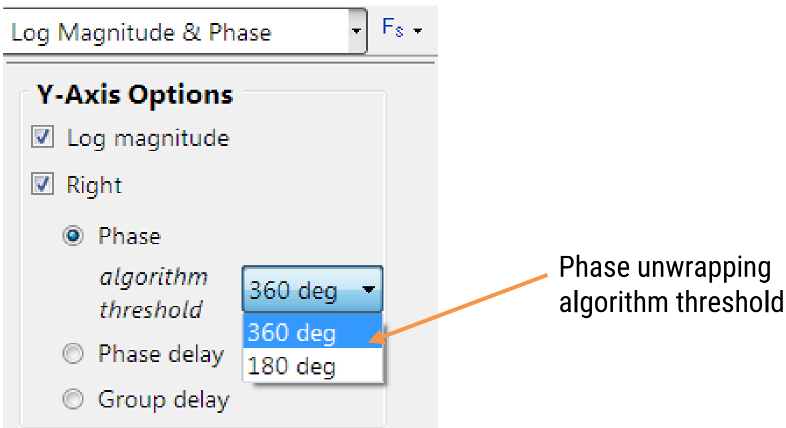
2.3. Frequency response chart

The frequency response chart shows the designed filter's frequency response, data markers (used for the chart data analytics) as well as the design markers used for the design specification.



2.3.1. Changing chart view

Select which data you wish to view via the **main toolbar** > **chart options** menu, as shown below:



As seen, the left Y-axis is always log magnitude, and the right Y-axis may be switched between phase, phase delay and group delay respectively.

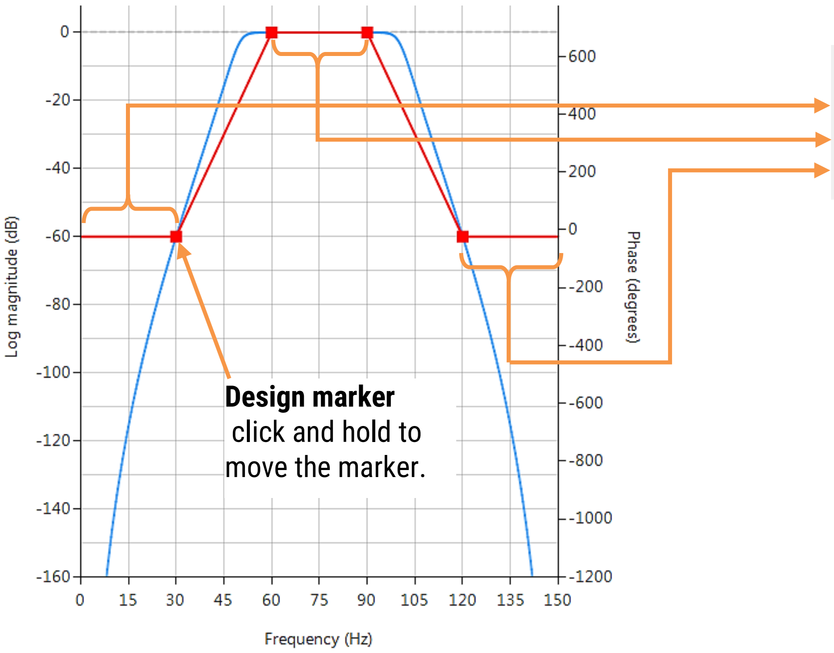
The phase unwrapping algorithm threshold allows you to switch between **360 degrees** (default) and **180 degrees**. Where the latter is particularly useful for viewing the continuous phase spectrum.

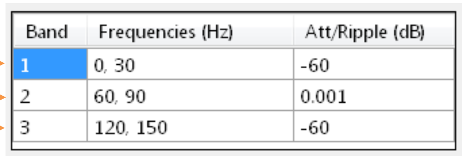
2.3.2. Design specification markers

The design specification markers concept forms the essence of the intuitiveness of the tool, allowing designers to graphically specify their design specifications and see the true filter frequency response in real-time:

Click and hold the red square with the left mouse button and then drag in any direction to modify the marker's position. The filter specification table will automatically be updated.

The filter specification is broken up into *bands* and summarised in the filter specification table.





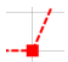
Band	Frequencies (Hz)	Att/Ripple (dB)
1	0, 30	-60
2	60, 90	0.001
3	120, 150	-60

You may fine tune the design specification by manually modifying the entries in the filter specification table.
NB. This will also change the appearance of the design specification line to dashed lines ---- *until the* **Design** *button is pressed (see overleaf for more details).*

2.3.2.1. Design specification line appearance

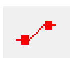
The design specification line appearance may take one of three settings depending on the action being performed:

Normal: This is the standard setting used for designing with the IIR and FIR design methods.

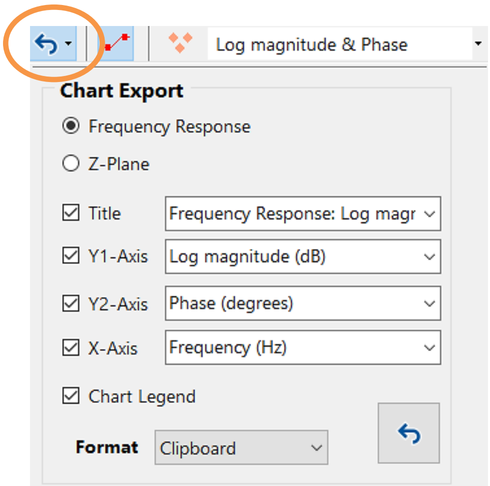


Dashed: If a filter specification table entry is being modified by the user or in the quantisation menu.

Illustration: If using the P-Z editor (**User Defined** mode), and an H1 filter designed by the design methods is present.

You may show or hide the design specification line via  the button in the main toolbar.

2.3.3. Exporting charts



Licensed users may export both the frequency response and z-plane charts to **clipboard** or as a high resolution picture file – where, **bmp, gif, jpeg, emf** and **png** formats are supported. The GUI allows you to edit the axis titles and include or exclude the chart legend.

You may also export the chart data to a text file (**Text file**), which allows for further customisation (such as adjusting line thickness) in third party programs.

2.3.4. Data analysis

Data analysis is performed with the mouse. Where, moving the mouse over the chart will automatically produce data markers and data analytics (shown at the bottom right side of the GUI).

The data analysis algorithm implements a specialised version of the Discrete Fourier transform, which allows designers to perform high resolution frequency analysis of any point of interest on the magnitude, phase, group delay and phase delay charts respectively.

Care should be exercised when analysing the frequency response chart, as the specialised implementation computes the Fourier component at specified frequency points rather than the standard $\frac{F_S}{N}$. The virtue of this implementation results in the evaluation of the exact magnitude and phase values at specific frequency points. Although this is desirable for the magnitude response (allowing you to see the exact magnitude at a given frequency point), the unwrapped phase estimate may vary slightly when panning and zooming over certain ranges - see section 2.3.9 for more information.

Version 5.1.1 and onwards includes data rulers functionality,  that allows users to analyse the frequency response chart using rulers.

2.3.4.1. Zooming in/out

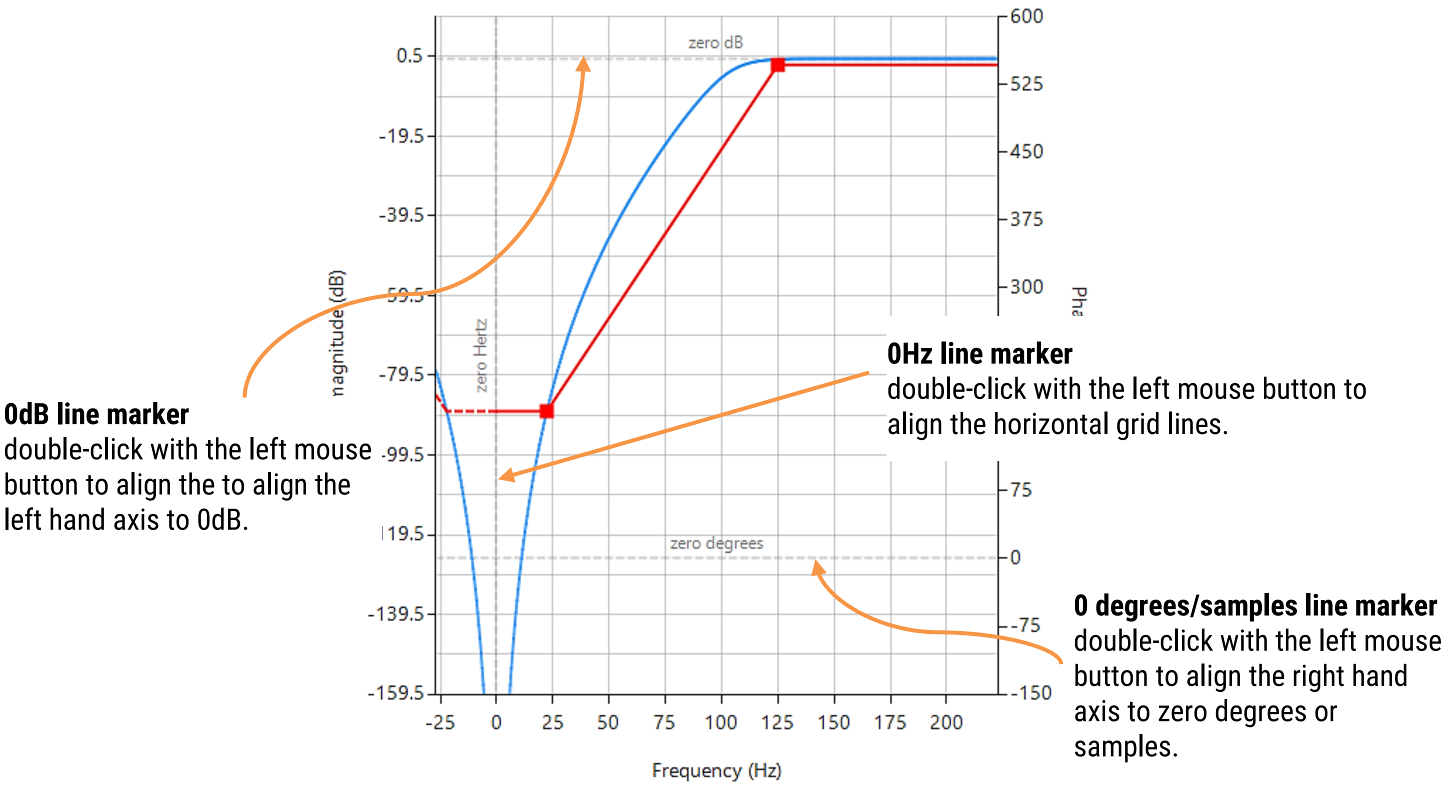
You may zoom in and out into any area by using the mouse wheel:

The zoom is centred on the position of the mouse pointer, in order to accommodate regional zoom functionality.

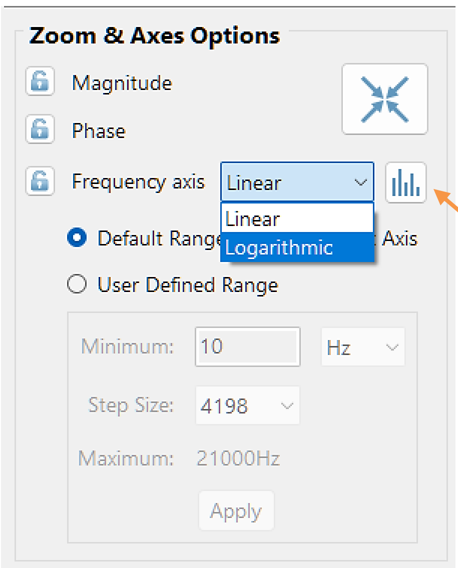
2.3.4.2. Panning

Panning may be achieved by depressing the left mouse button and dragging the mouse in any direction. Where, the frequency panning range is limited to ± Nyquist.

2.3.5. The 0dB, 0 degrees/samples and 0Hz line markers



2.3.6. Logarithmic and linear frequency Axis



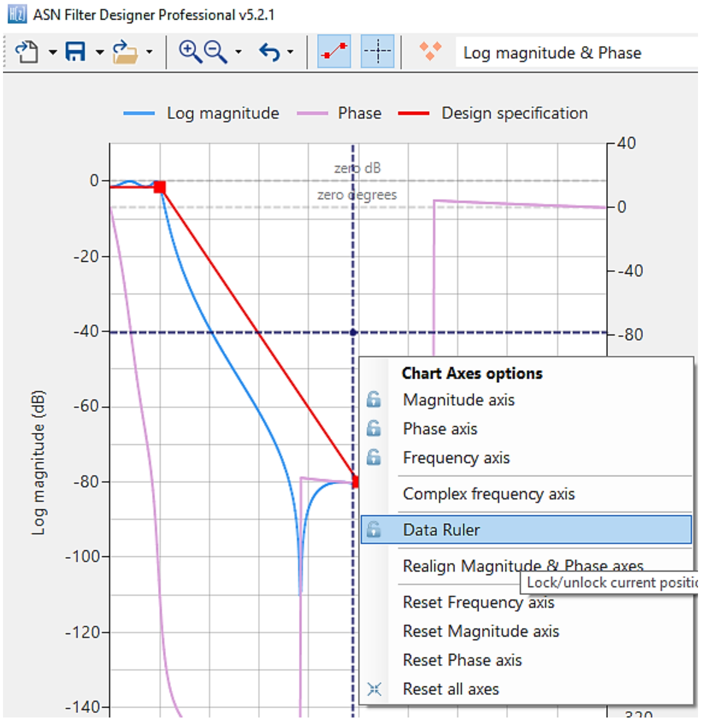
You may choose between a **Logarithmic** and **Linear** frequency axis using the Axis listbox. A Logarithmic frequency axis is useful for analysing IIR and converted analog transfer functions.

Use the Audio frequency axis shortcut to set the frequency axis to a logarithmic scale (e.g. 10Hz-21kHz). *This option will only be available when the sampling rate is either 8kHz, 11.025kHz, 16kHz, 22.05kHz or ≥42kHz*.

2.3.7. Data rulers

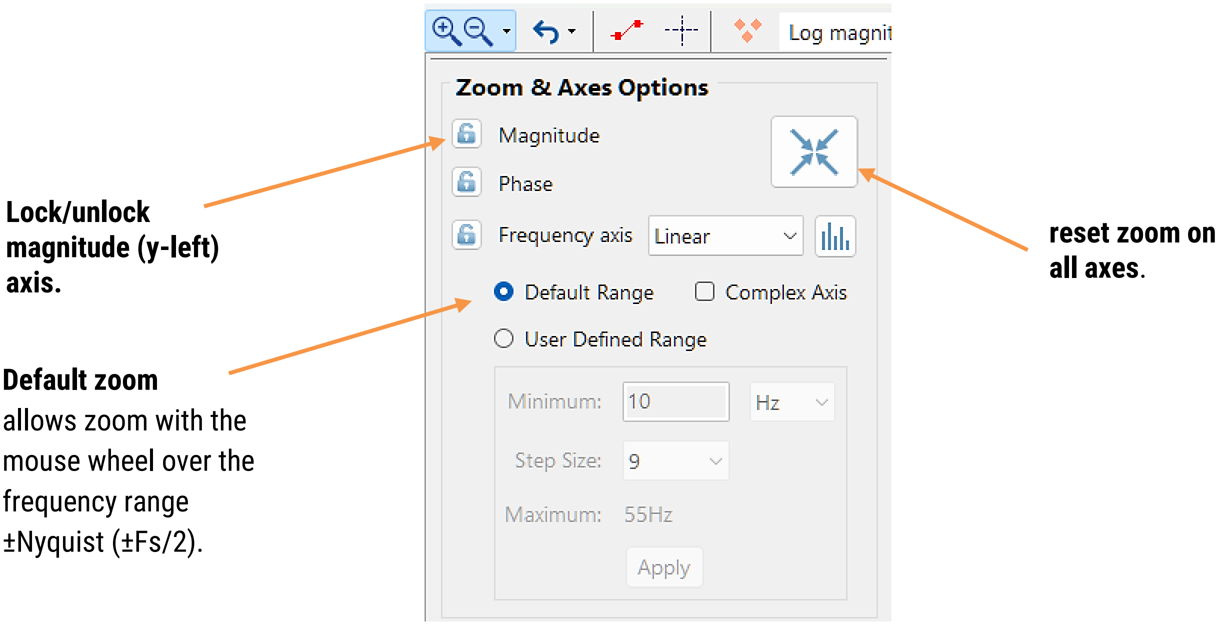
You may get more detailed analysis from the Data rulers 

Use the chart context options menu (right mouse button click on the chart) to lock the rulers on the chart. This is useful for marking a position of interest, such as frequency of a filter's -3dB point.



2.3.8. Chart Zoom & Axes options

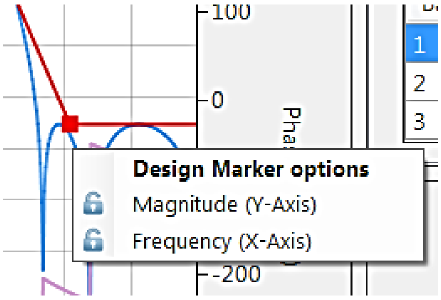
The ASN Filter Designer provides designers with a comprehensive zooming and axes options menu for undertaking analysis of demanding filter designs.



2.3.8.1. Locking axes

In order to simplify data analysis with three axes, you may lock a specified axis for zooming/panning purposes. This has the advantage of allowing you to customise each chart axis to your exact requirements.

2.3.8.2. Locking a design marker

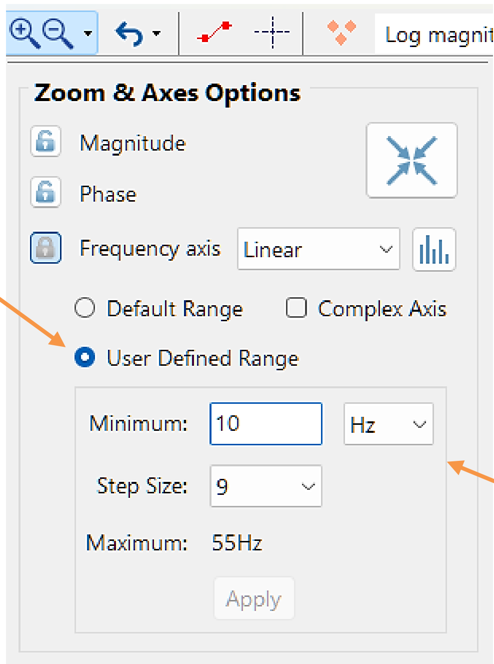


In order to help fine tune a design marker's specification, you may restrict a design marker's X (frequency) and Y (magnitude) movement. Hovering the mouse over a design marker and clicking on the right mouse button, presents an options menu as shown on the left.

When designing IIR Butterworth filters, setting **Rp** (passband ripple) to **3dB** will automatically lock the Rp **Magnitude** design marker(s) at 3dB. Using the **Design Markers options** menu (shown on the left), you may remove the lock if so required.

2.3.8.3. Zooming to a specific frequency range

user defined zoom
You may zoom to specific frequency range with the **User defined** zoom function.
The universality of this function allows you zoom to mHz resolution even when the sampling rate is in the MHz region!



Choose the frequency scale that you wish to zoom to. Notice here that we are setting the x-axis to the range: `10-55mHz`.

Panning is disabled on the x-axis (frequency) when this function is enabled!

2.3.9. Interpreting the phase spectrum and 0Hz

The frequency spectrum is constructed via the CZT (chirp z-transform) which is a generalisation of the more traditionally used DFT (discrete Fourier transform) - thus, allowing designers to analyse any frequency range desired. This feature also allows designers to perform detailed phase discontinuities analysis as well as detailed data analysis at other frequency scales (such as mHz, Hz, kHz etc) even when the sampling rate is in MHz range.

Care should be exercised when interpreting the results, as only 500 CZT data points are used over the desired frequency range, instead of the approach adopted by other tools that use thousands of DFT points. As a result, zooming or panning over certain ranges may give slightly different results to the original ±Nyquist (±Fs/2) phase plot. The difference is attributed to the higher resolution (finer step size) between the CZT computation points, which affect the phase unwrapping algorithm which is a relative function (the magnitude spectrum estimates will always remain the same). These zoomed values should be interpreted as the true phase values.

In order to overcome glitches, the CZT is not actually computed at 0Hz, but at 1e-4Hz. Where, this minimum value is automatically adjusted depending on the frequency scale.

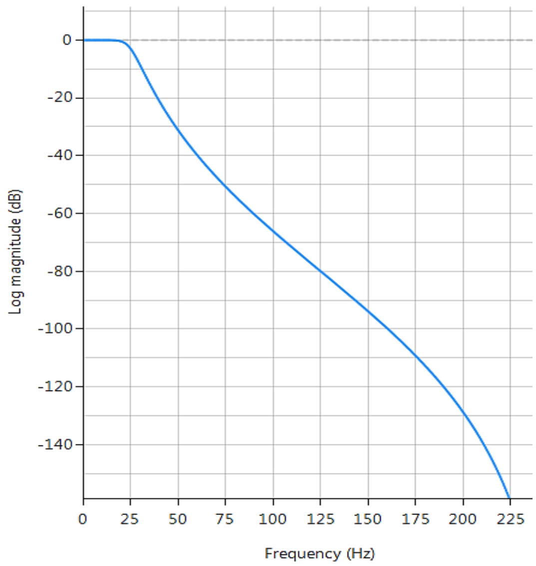
2.3.10. Errors in high order polynomials

The tool will for FIR filters and the filter script use the given **Num** and **Den** polynomials for computation. However, if these positions are modified via the P-Z editor, they will be handled via the roots-to-poly algorithm which will attempt to reconstruct the polynomial from the presented roots using double precision arithmetic. For lower orders this will generally result in an almost identical polynomial, but as a consequence of the errors inherent to the root finding algorithm, higher order polynomials (> 60 or so) may significantly deviate from the ideal result.

2.4. Classical IIR Filter design

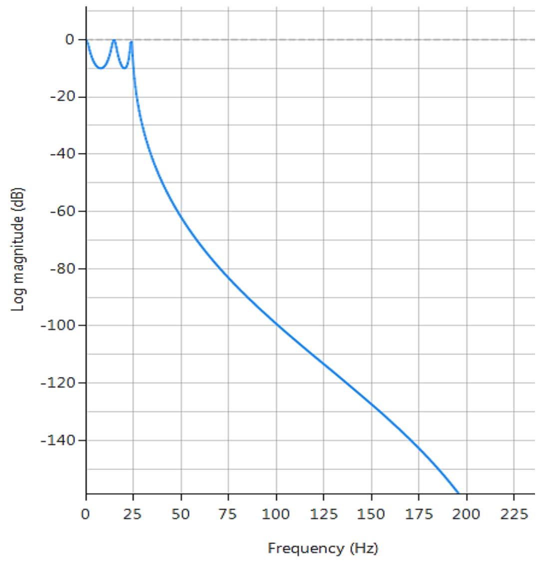
The IIR filter designer allows developers to implement the following classical design prototype methods for lowpass, highpass, bandpass and bandstop filters:

Butterworth (5th order)



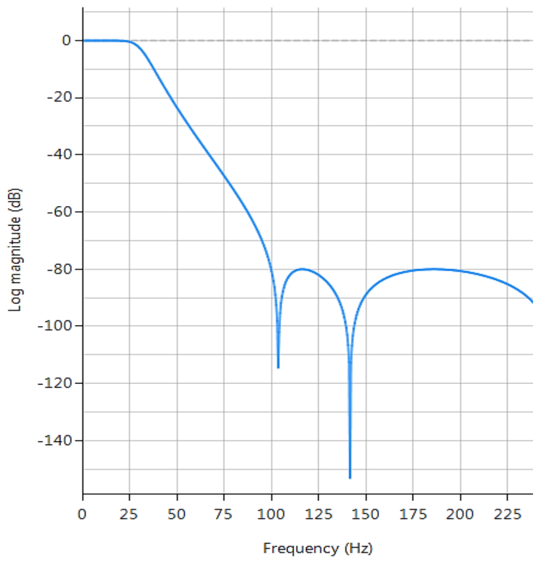
- Smooth monotonic response (no ripple).
- Slowest roll-off for equivalent order.
- Highest order of all supported prototypes.

Chebyshev type I (5th order)



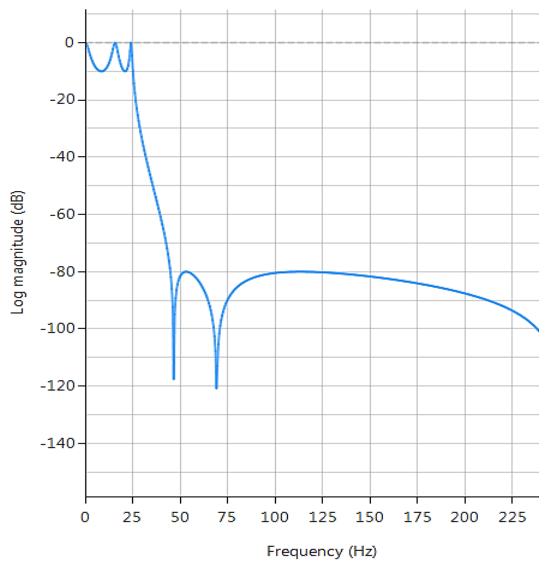
- Maximally flat stopband.
- Faster roll off (passband to stopband transition) than Butterworth.

Chebyshev type II (5th order)



- Maximally flat passband.
- Slower roll off (passband to stopband transition) than Chebyshev Type I.

Elliptic (5th order)



- Equiripple in both the passband and stopband.
- Fastest roll-off.
- Lowest order filter of all supported prototypes.

The frequency response charts shown above show the differences between the various design prototype methods for a 5th order lowpass filter with the same specifications. As seen, the Butterworth response is the slowest to roll-off and the Elliptic the fastest.

The Bessel prototype is not supported, as the Bilinear transform warps the linear phase characteristics. However, a Bessel filter design method is available in ASN FilterScript.

2.4.1. IIR Designer GUI

Double-click on the tab to re-design with default settings.

Filter order: As default, the tool computes this automatically based on the technical specification (**Auto** checked).

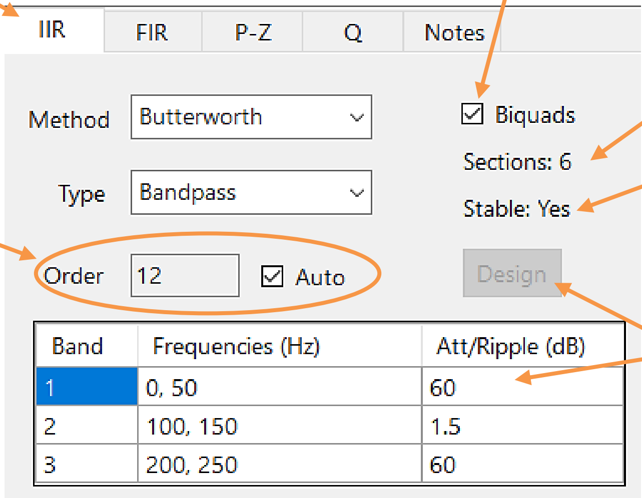
You may override the automatic computation and specify your desired filter order by unchecking **Auto**.

[Biquad or single section](#) implementation.

number of biquad sections in the filter cascade.

filter stability (poles inside the unit circle).

1. Fine tune a table entry by double-clicking on it.
2. Click on the **Design** button to update.



Filter orders of up to `100` (professional version only) may be constructed. However, in the case of bandpass and bandstop design, only even filter orders are available.

IIR designs may be extended upon by utilising the P-Z editor, by allowing designers to modify or create a new filter by editing, adding or deleting any poles or zeros. In this mode (**User defined**), the **Method** dropdown list changes to **User defined**, as the design is no longer categorised by an analog prototype. The comprehensive editor options allow for the design and customisation of any combination of poles and zeros, including the re-optimisation of the filter structure for implementation - see section 8 for more details.

2.4.2. Method specifications

The IIR Designer UI automatically determines the lowest required filter order for the given specifications to be met (**Auto** checked). Depending on the **Method** selected, the passband and stopband ripple/attenuation characteristics met exactly or are designed to be 'at least' or 'no more than' the specified value. The table shown below summarises the properties of each Method.

Method	**Description**
Butterworth, Chebyshev Type I, Chebyshev Type II and Elliptic	Designs a filter with the lowest order required having no more than Rp dB of passband ripple and at least Rs dB of attenuation in the stopband.
Butterworth	Setting **Ripple** to **3** (i.e. 3dB) will automatically change the design rules to match the passband specification exactly, i.e. the 3dB cut-off frequencies. ⚠ The tool will automatically lock the Rp design marker(s) at 3dB, please refer to section for details on the [design marker locking mechanism](#).

2.4.3. Biquads

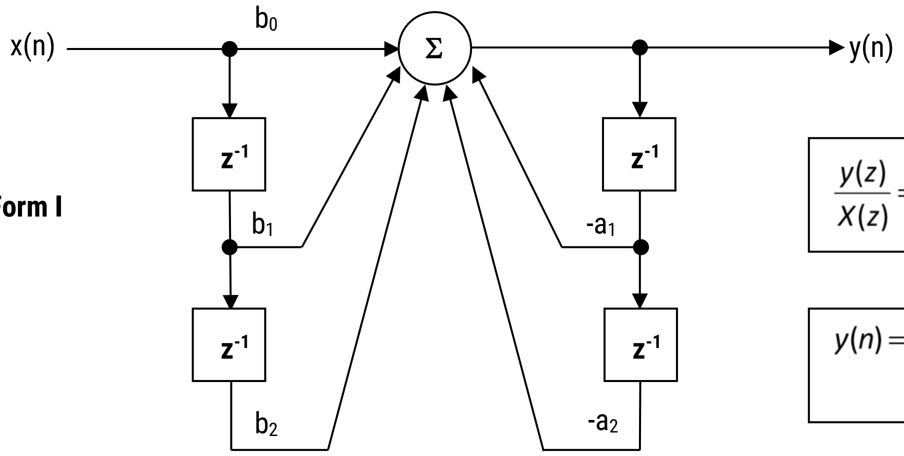
All classical IIR filters are implemented as biquad filters (i.e. two poles and two zeros) as default. For any users requiring a single section implementation, simply uncheck the **Biquads** checkbox. However, as mentioned in section 2.3.10, higher filter orders generally lead to stability problems when poles are near to the unit circle.

The biquad implementation is particularly useful for fixed point implementations, as the effects of quantization and numerical stability are minimized. However, the overall success of any biquad implementation is dependent upon the available number precision, which must be sufficient enough in order to ensure that the quantized poles are always inside the unit circle.

Analysing the biquad structures (shown overleaf), it can be seen that although the transfer functions are identical, the difference equations (i.e. time domain implementation) are quite different. The Direct Form II Transposed structure is considered the most numerically accurate for floating point implementation, and is therefore the default filter structure. However, the Direct Form I is advocated for fixed point implementation by virtue of the single accumulator.

The ASN filter designer supports the following three IIR filter structures:

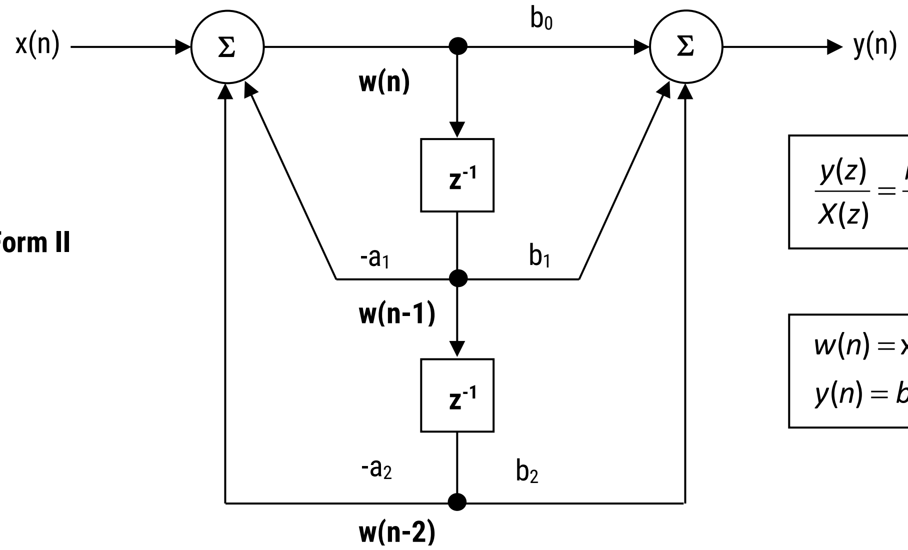
Direct Form I



$$\frac{y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

$$y(n) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) - a_1 y(n-1) - a_2 y(n-2)$$

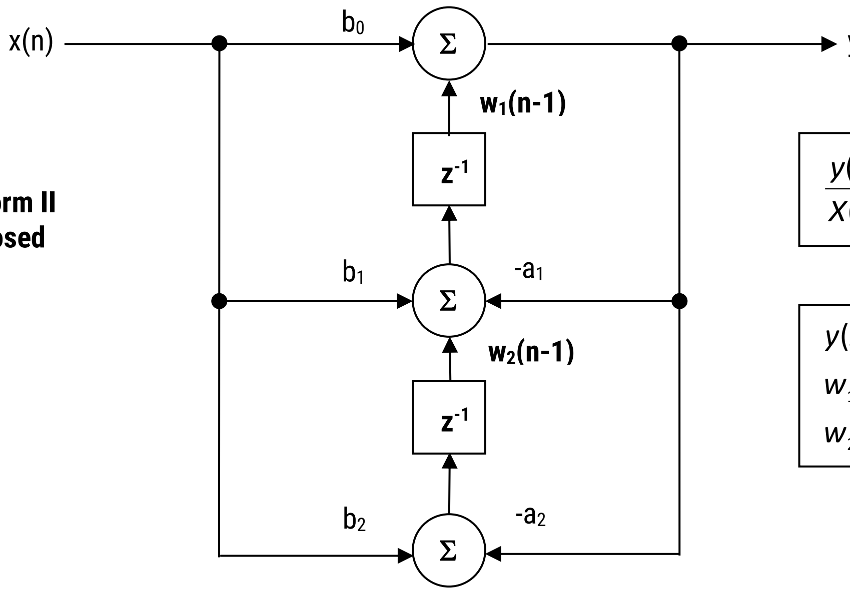
Direct Form II



$$\frac{y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

$$w(n) = \mathrm{x}(n) - a_1 w(n-1) - a_2 w(n-2)$$
$$y(n) = b_0 w(n) + b_1 w(n-1) + b_2 w(n-2)$$

Direct Form II Transposed

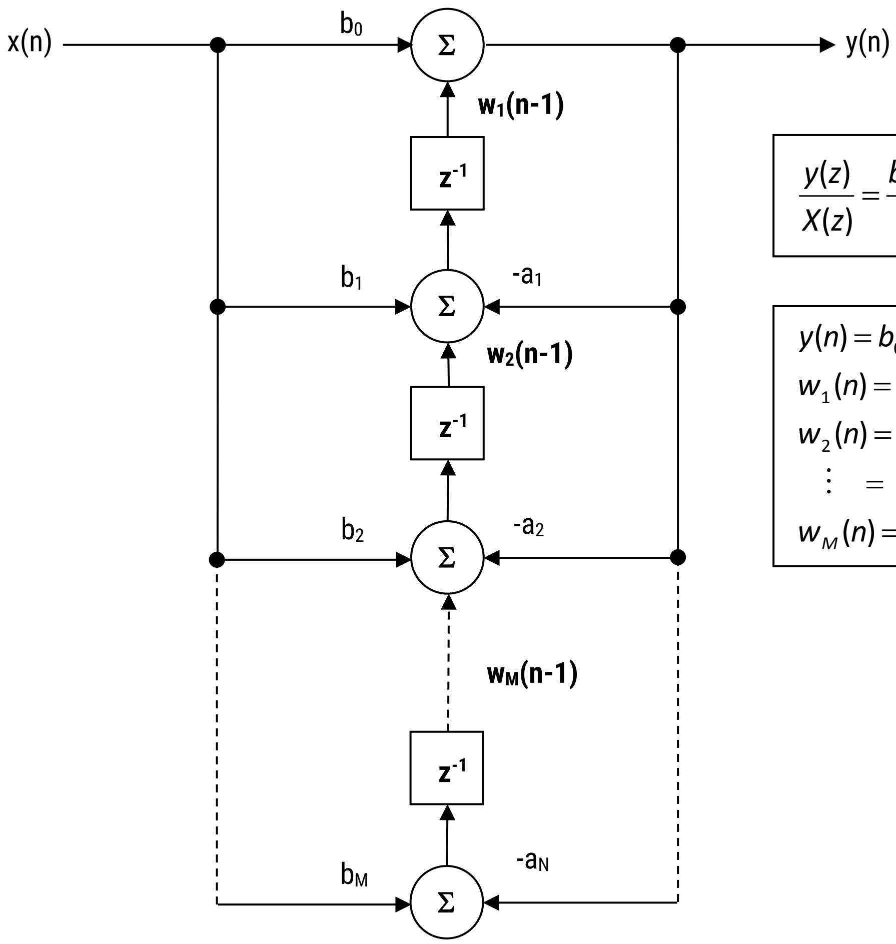


$$\frac{y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

$$y(n) = b_0 x(n) + w_1(n-1)$$
$$w_1(n) = b_1 \mathrm{x}(n) - a_1 y(n) + w_2(n-1)$$
$$w_2(n) = b_2 \mathrm{x}(n) - a_2 y(n)$$

2.4.4. Single section IIRs

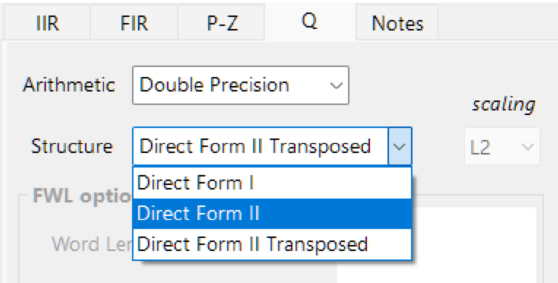
The ASN Filter Designer supports the design and implementation of both biquad and single section IIR filters. The concept of a **Direct Form II Transposed** single section filter is shown below for the case when *M=N*:



$$\frac{y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + ... + b_M z^{-M}}{1 + a_1 z^{-1} + a_2 z^{-2} + ... + a_N z^{-N}}$$

$$\begin{aligned} y(n) &= b_0 x(n) + w_1(n-1) \\ w_1(n) &= b_1 \mathrm{x}(n) - a_1 y(n) + w_2(n-1) \\ w_2(n) &= b_2 \mathrm{x}(n) - a_2 y(n) + w_3(n-1) \\ \vdots &= \quad \vdots \\ w_M(n) &= b_M \mathrm{x}(n) - a_N y(n) \end{aligned}$$

2.4.5. IIR structure and Arithmetic options

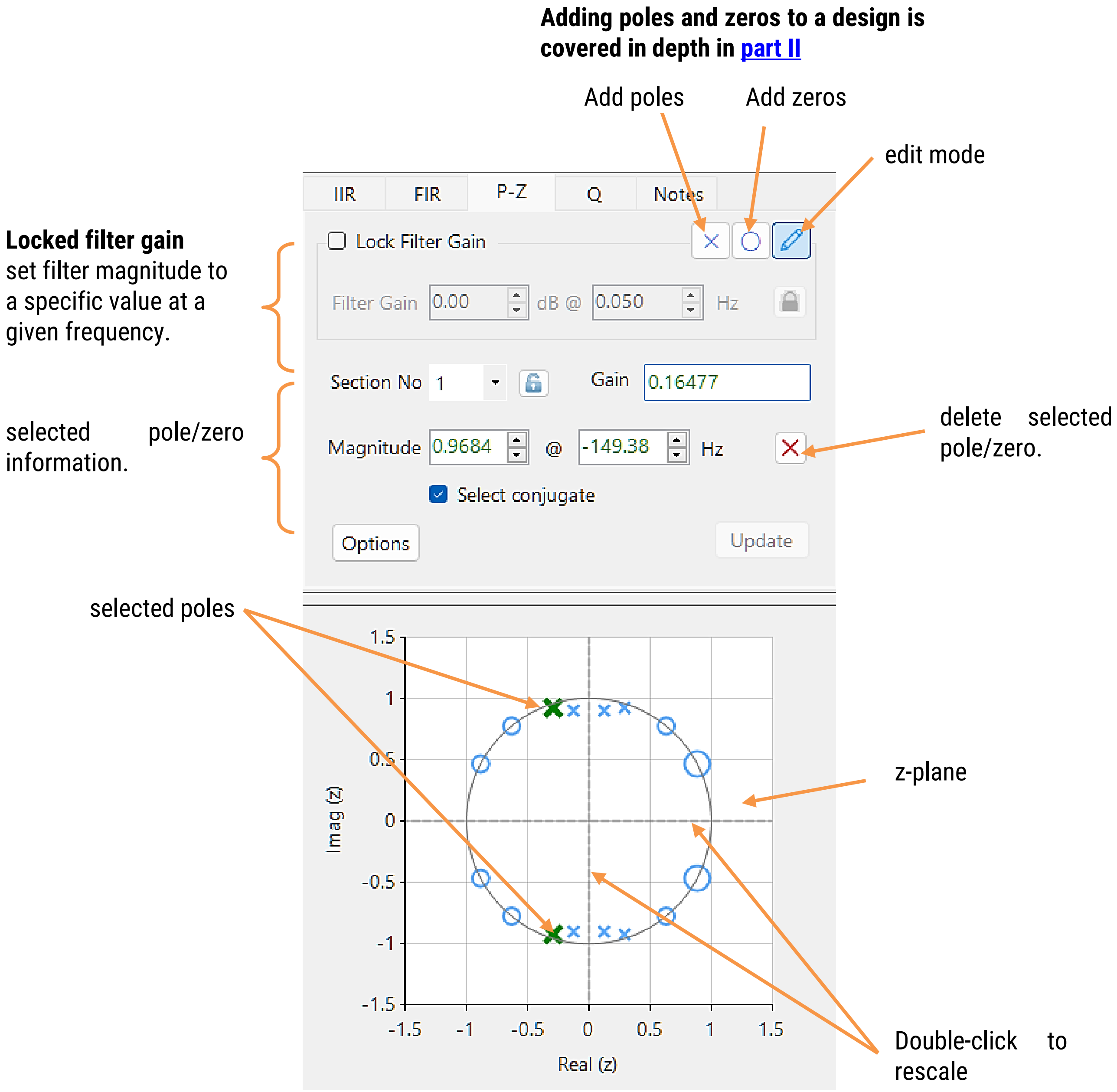


Filter structures and arithmetic options for both IIR and FIR filters may be found under the **Q** tab.

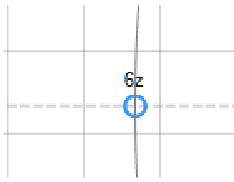
NB. When using **Fixed point** arithmetic and the **Direct Form II** structure, the **scaling** option must be set - see section 5 for more details.

3. P-Z editor

The P-Z (pole-zero) editor provides designers with a comprehensive but easy to use pole-zero editor, together with a few other useful options not commonly found in other filter design software.



If any poles (#p) or zeros (#z) overlap, then a number appears next to them.



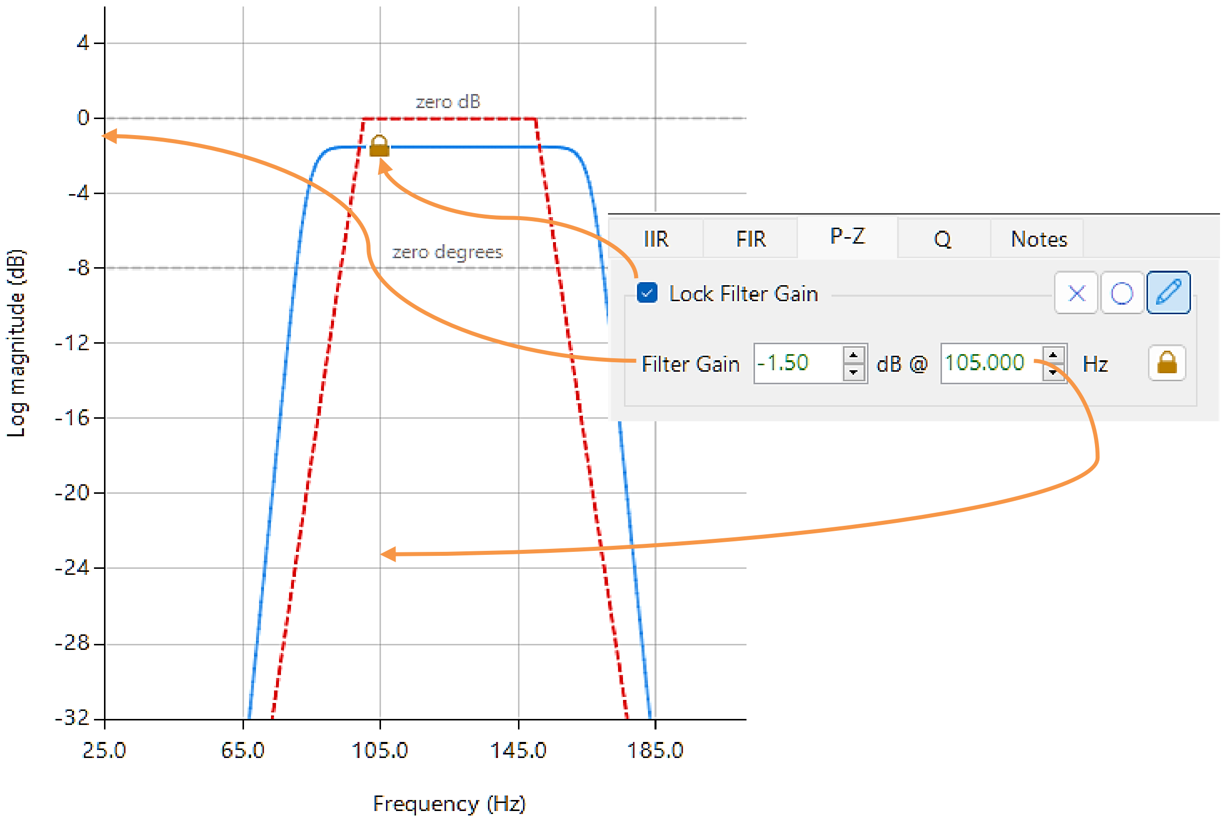
3.1. Zooming in/out and panning

As with the frequency response chart, you may zoom in and out into any area by using the mouse wheel.

Scrolling may be achieved by depressing the *left mouse button and dragging* the chart in any direction.

3.2. Locked filter gain

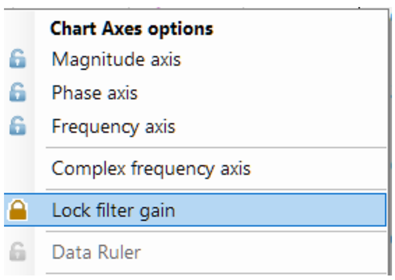
The **Locked Filter Gain** automatically calculates the filter gain required in order to set the magnitude spectrum to the specified gain in dB at the specified frequency. After clicking on the button, a brown padlock will appear on the frequency response chart at the specified location.



The specified **Filter Gain** is set to -1.5dB at 105Hz.

The exact **Locked Filter Gain** value will appear in the filter summary, and is automatically included into the filter implementation via the signal analyser.

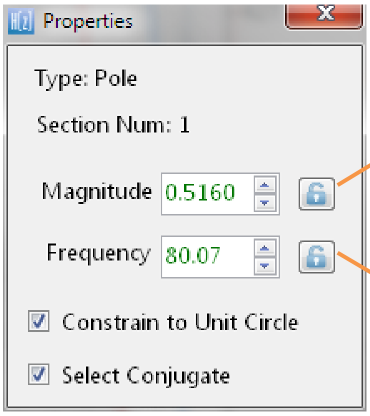
From v5.3.3 onwards, the padlock is fully interactive and can be dragged about with the mouse. This functionality is extremely useful for fine-tuning the gain of the filter visually.

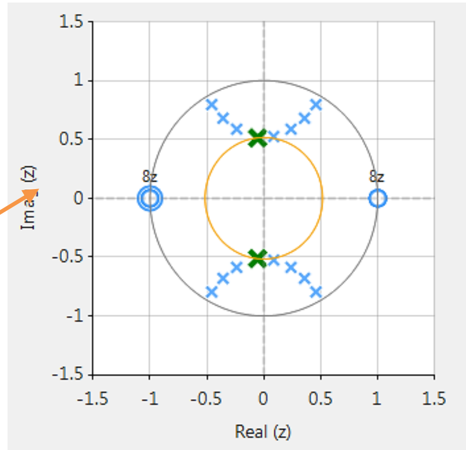


You may use the chart zoom context to menu to enable functionality too

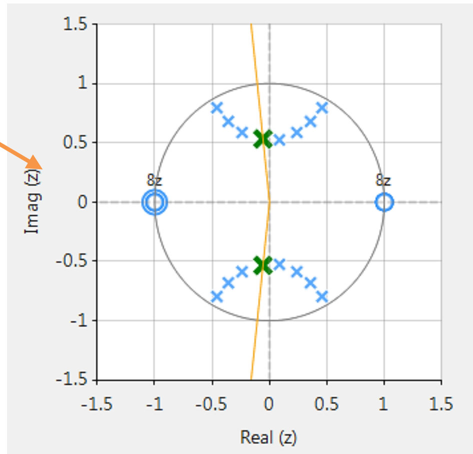
3.3. Pole-zero properties window

You may get the specific properties of a **pole** or **zero** by double-clicking on it in the P-Z chart.





Locked magnitude



Locked frequency

Locking the magnitude or frequency results in orange guidelines (see right) appearing on the chart. This is functionality is extremely useful for locking a dimension when modifying pole/zero position with the mouse.

3.4. Section number and section lock

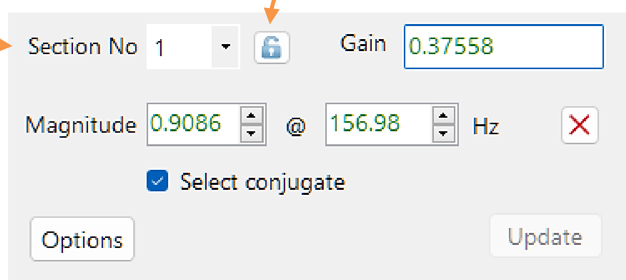
Section number

This allows you to highlight the pole and zeros of a specific section in the H1 filter:

- For FIR filters or single section IIR filters: the section number will always be equal to `1`.
- For biquad IIR filters: this will be a list of all the biquad sections in the filter cascade.

Section lock

Clicking on the section lock, allows you to focus on a specific section by highlighting all of the poles-zeros of the selected section number (**Section No**) and minimising the rest.



3.5. FIR Filter design

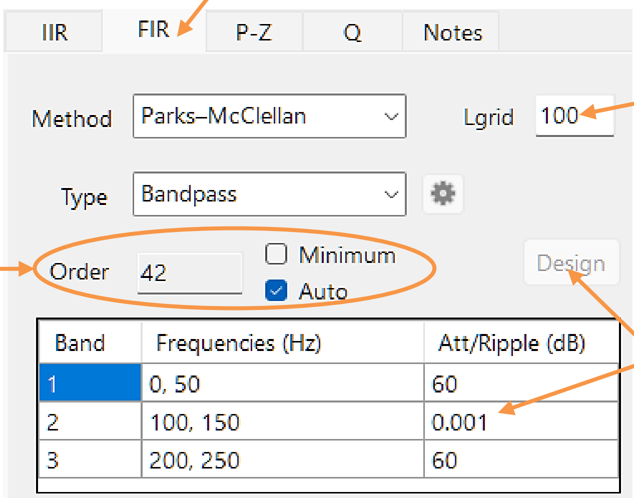
The FIR (finite impulse response) filter designer is implemented via the Parks-McClellan algorithm, and allows for the design of the following filter types:

Filter Types	Description
Lowpass	Designs a lowpass filter
Highpass	Designs a highpass filter
Bandpass	Designs a bandpass filter
Bandstop	Designs a bandstop filter
Multiband	Designs a multiband filter with an arbitrary frequency response
Hilbert transformer	Designs an all-pass filter with a -90 degree phase shift
Differentiator	Designs a 1^{st} order differentiator (numerical differentiation)
Double Differentiator	Designs a 2^{nd} order differentiator (numerical differentiation)
Integrator	Designs a 1^{st} order integrator: bandlimited cumsum (numerical integration)
Double Integrator	Designs a 2^{nd} order integrator: bandlimited cumsum (numerical integration)

The Parks-McClellan algorithm offers a degree of flexibility over other FIR design methods, in that each band may be individually customised in order to suit the designer's requirements.

Double-click on the tab to re-design with default settings.

Filter order: By default, the tool computes this automatically based on the technical specification (**Auto** checked).
You may override the automatic computation and specify your desired filter order by unchecking **Auto**.



Parks-McClellan algorithm grid step size parameter (`16–900`).

Fine tune a table entry by double-clicking on it.
Click on the **design** button to update.

Filter orders of up to `499` (professional version only) may be constructed, where this is limited to `200` for streaming audio applications. As with the IIR filters, an FIR's zeros may be modified by the P-Z editor (**Method** dropdown list changes to **User defined**), including the ability of adding poles and converting it into an IIR filter - see section 8 for more details.

Higher order FIR designs (>100): In order to speed up plotting performance, updates to the P-Z chart are postponed until the *left mouse button* is released.

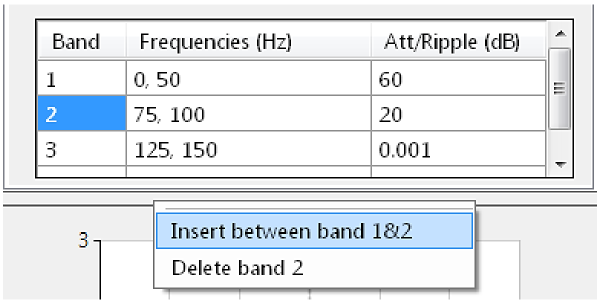
The order estimation of the Parks-McClellan algorithm may sometimes underestimate the filter order required for the given specifications. Therefore, in order to automatically increase the order estimate by 2 (overestimation) you may uncheck the **Minimum** checkbox.

3.5.1. Convergence and errors

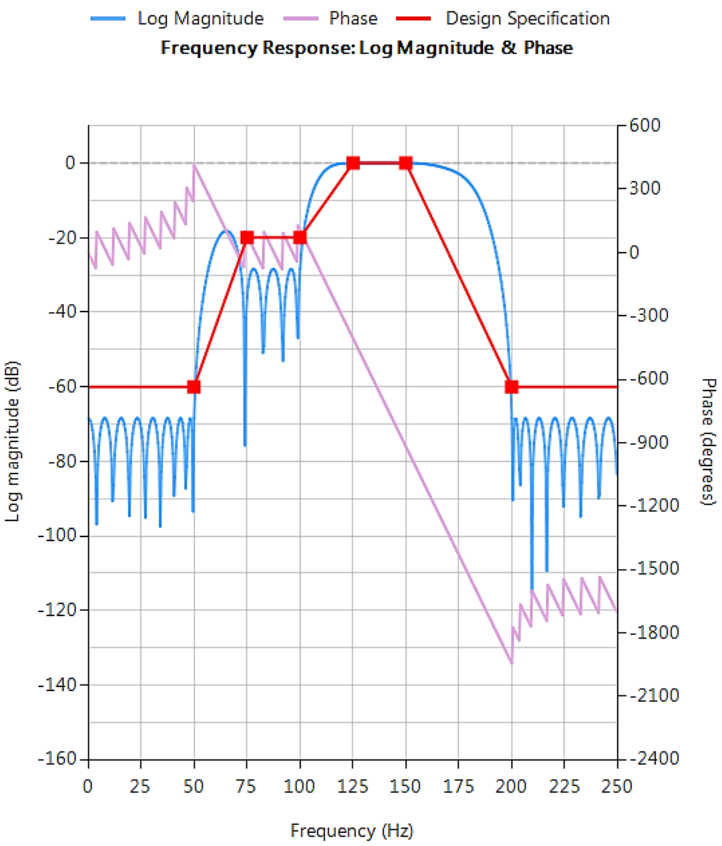
The Parks-McClellan algorithm is an optimal Chebyshev FIR design method, however the algorithm may not converge for some specifications. In such cases, increasing the distance between the design marker bands generally helps.

Errors in the root finding algorithm usually lead to undesirable results for high order filter implementations. As a consequence, the zeros presented in the P-Z chart for higher orders (> 60 or so) should only be interpreted as an illustration of the true positions. Also, if you are designing a high order FIR filter with a few hundred taps, it is not recommended to use the P-Z editor for editing the positions of the zeros.

3.5.2. Multiband FIR



In order to implement an arbitrary frequency response, you may use the **Multiband** design method. Extra bands may be added or removed from the design specification table by right-clicking on a **Band** and selecting the required option.

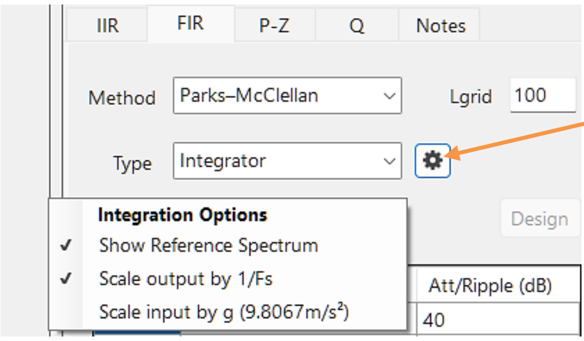


The design method requires that at least one band is a passband.

All bands with an attenuation of 10dB or less are classed as *passbands*. Depending on the level of band attenuation specified, the tool will automatically convert a stopband into a passband and vice versa.

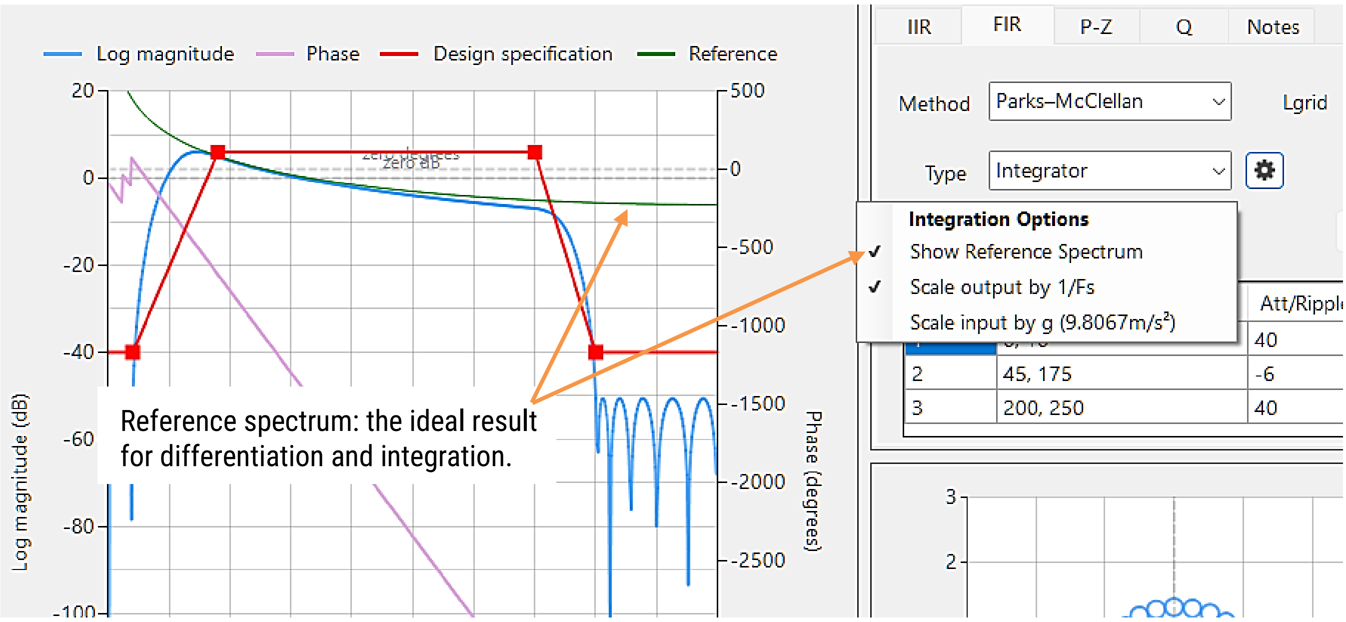
3.5.3. Integrators and differentiators

The tool simplifies the design of FIR Integrators and Differentiators need for MEMS sensor applications and other IoT applications.



The settings menu can be used to set scaling options (needed for MEMS data and numerical integration/differentiator) and displaying the Reference spectrum.

Use the Design markers to fine tune the filter's specification.



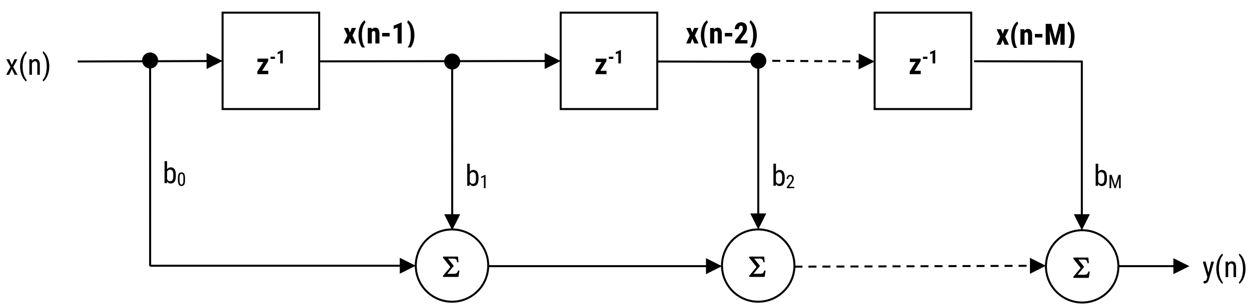
Moving the mouse over the Reference spectrum provides extra analytics, such relative error and expected output amplitude for test sinusoid.

Reference spectrum	Transfer function	Design method description
Differentiator	$H(z) = 1 - z^{-1}$	$H(z)$ is augmented with a lowpass filter.
Double Differentiator	$H(z) = 1 - 2z^{-1} + z^{-2}$	$H(z)$ is augmented with a lowpass filter.
Integrator	$H(z) = \dfrac{1}{1 - z^{-1}}$	As $H(z)$ is BIBO unstable, it is augmented with a bandpass filter.
Double Integrator	$H(z) = \dfrac{1}{1 - 2z^{-1} + z^{-2}}$	As $H(z)$ is BIBO unstable, it is augmented with a bandpass filter.

3.5.4. FIR structures

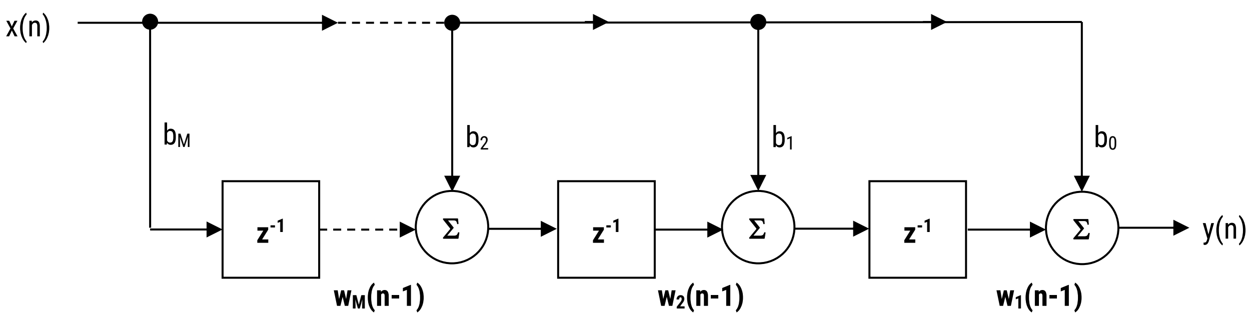
FIR (finite impulse response) filters are useful for a variety of signal processing applications, including audio signal processing and noise cancellation. Although several practical implementations for FIRs exist, the direct form structure and its transposed cousin are perhaps the most commonly used, and as such all designed filter coefficients are intended for implementation in a Direct form structure.

$$\frac{y(z)}{X(z)} = b_0 + b_1 z^{-1} + b_2 z^{-2} + ... + b_M z^{-M}$$



Direct Form structure

$$y(n) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) + \ldots + b_M x(n-M)$$



Direct Form Transposed structure

$$\begin{array}{lllll} y(n) & = & b_0 x(n) & + & w_1(n-1) \\ w_1(n) & = & b_1 \mathrm{x(n)} & + & w_2(n-1) \\ w_2(n) & = & b_2 \mathrm{x(n)} & + & w_3(n-1) \\ \vdots & = & \vdots & + & \vdots \\ w_M(n) & = & b_M \mathrm{x(n)} & & \end{array}$$

The ASN filter designer supports the design and implementation of both **Direct Form** and **Direct Form Transposed** FIRs. As with IIR filters, the default structure is the **Direct Form Transposed** structure by virtue of its superior numerical accuracy when using floating point.

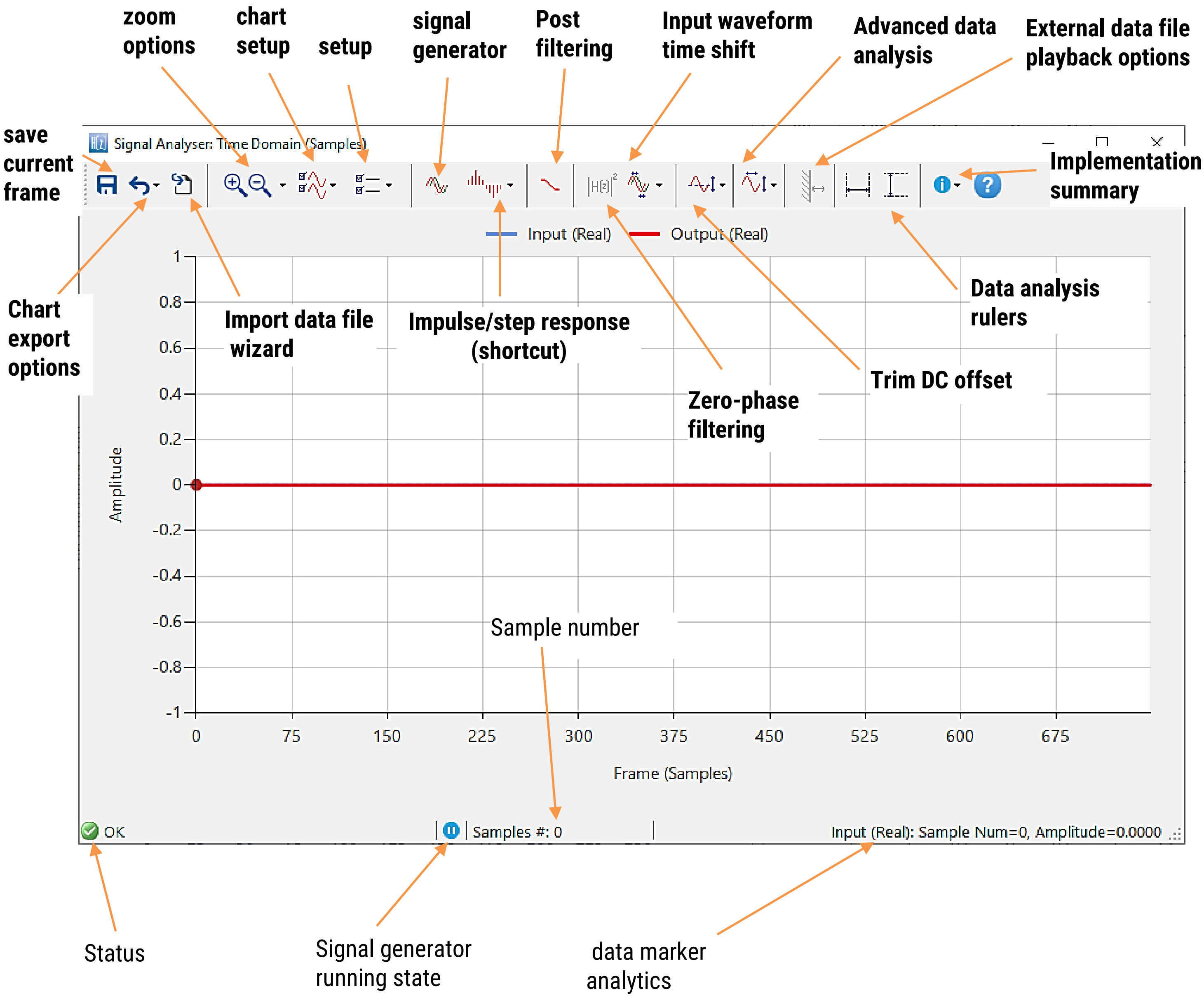
4. The signal analyser

You may start the signal analyser by clicking on the button in the main toolbar.

The signal analyser allows designers to test their design on audio, real (user) data or synthetic data via the built-in signal generator. Default data playback is implemented as streaming data, providing a simple way of assessing the filter's dynamic performance, which is especially useful for fixed point implementations.

Both frequency domain and time domain charts are fully supported, allowing for design verification via transfer function estimation using the cross and power spectral density functions. As with all other charts, the signal analyser chart fully supports advanced zooming and panning, as well as comprehensive chart data file export options.

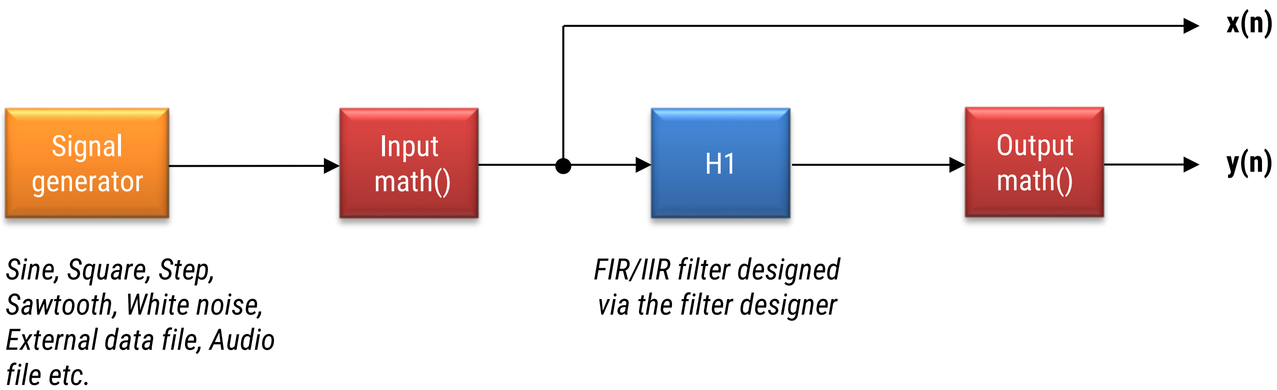
The signal analyser GUI is shown below.



4.1. Architecture

The signal analyser GUI is comprised of a time/frequency domain analyser and a signal generator. The GUI allows designers to explore the time and frequency characteristics of their designed H1 filter for various types of quantisation and inputs, but is flexible enough to also support analysis of 3rd party datasets. The signal analyser supports implementation for both real and complex coefficient filters, allowing for experimentation of the most demanding filter designs!

A block diagram of the signal analyser's architecture is shown below:



As seen, the H1 filter is preceded and proceeded by optional **math()** function blocks which are useful for variety of signal processing operations, and may be independently enabled or disabled. If no mathematical function is required (Function() = `None`) the block is disabled and the data fed directly through. All operations are performed on complex data ($x = a + bi$), where the signal generator automatically converts real data into complex data by $x = a + 0i$. The following options are supported:

Function ()	Math operation	Description
`None`	-	Disable the function block.
`Abs`	$\lvert x\rvert = \sqrt{a^2 + b^2}$	Absolute.
`Ln`	$log_e\, x$	Natural logarithm.
`Angle`	$tan^{-1}\left(\frac{b}{a}\right)$	Compute the arctangent (phase in radians).
`RMS`	$\frac{\sqrt{a^2 + b^2}}{\sqrt{2}}$	Root mean square.
`Sqr`	x^2	Square.
`Sqrt`	$\sqrt{x}$	Square root.
`TKEO`	$y(n) = x^2(n-1) - x(n)(x-2)$	TKEO (Teager-Kaiser energy operator) algorithm.

In order to assess real-time performance of the filters, data from the signal generator is streamed (per sample) by default. However, in order to allow for impulse, step response and external data set evaluation, a blocked based mode is also provided - see section 4.2.1 for more information.

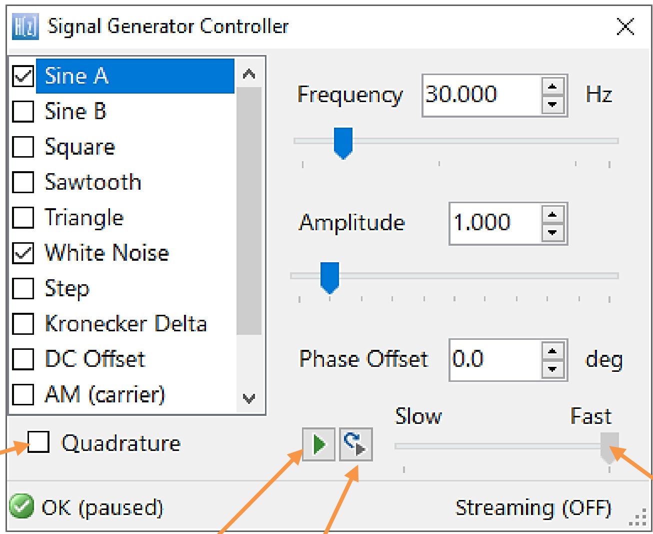
4.2. The signal generator

The signal generator allows you to test your filter with a variety of input signals, such as sine waves, square waves, white noise or even your own external test data.

The signal generator may be started by clicking on the  button in the tool bar.

Signal methods
Enable the methods you wish to use by setting the checkmark. Click on the method in order to set its properties.

As seen on the right, the output signal is comprised of a 30Hz sinewave and white noise.



signal method properties

convert the signal method into a quadrature signal (real component in-phase and imaginary component shifted by 90 degrees).

play/pause signal generator.

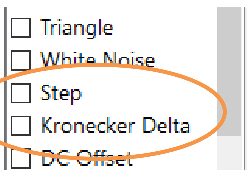
Reset signal generator and re-run.

playback speed (streaming only). Adjust the chart update speed.

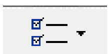
NB. This is disabled when streaming audio.

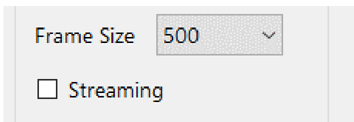
Two independent sinewaves (**Sine A** and **Sine B**) are available, allowing you to experiment with simple signal configurations for a variety of practical applications.

4.2.1. Impulse, step response and short external data set evaluation



You may evaluate your filter's impulse and step response characteristics by enabling either the **Kronecker Delta** method (impulse response evaluation) or the **Step** method (for step response evaluation). By default, the amplitudes are set to `1.000`, but may be changed as required. *See section 4.2.3 for details on the shortcut.*

Finally, for instant results, block based mode should be selected by unchecking the **Streaming** checkbox in  the Setup menu.



This feature is covered in depth in the following video tutorial.

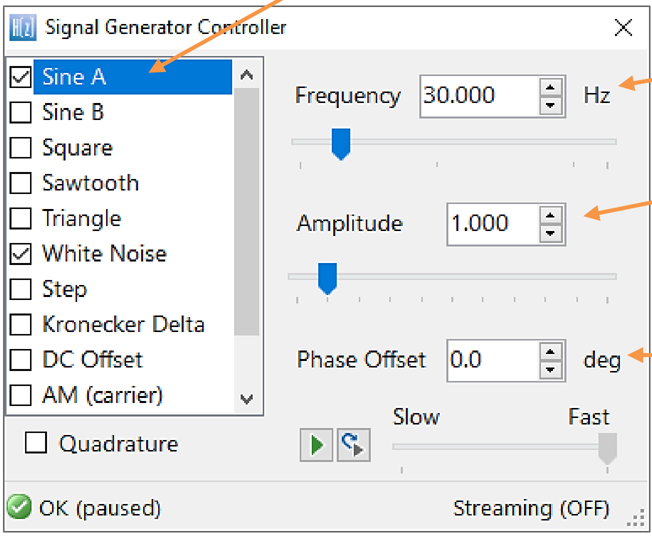
Block base mode is an extremely useful method of evaluation of short external data sets (refer to the Data Import Wizard on how to load them), as the filtering performance on datasets less than or equal to the selected **Frame Size** can be instantly visualised and optimised accordingly.

4.2.2. Streaming your first application with the tool

As an example application, let us assume that we want to test a designed filter with a 30Hz sinewave of amplitude `1.000` with some additive `White Noise`. This can be simply achieved by setting the generator up as follows:

STEP 1

Select the **Sine A** method

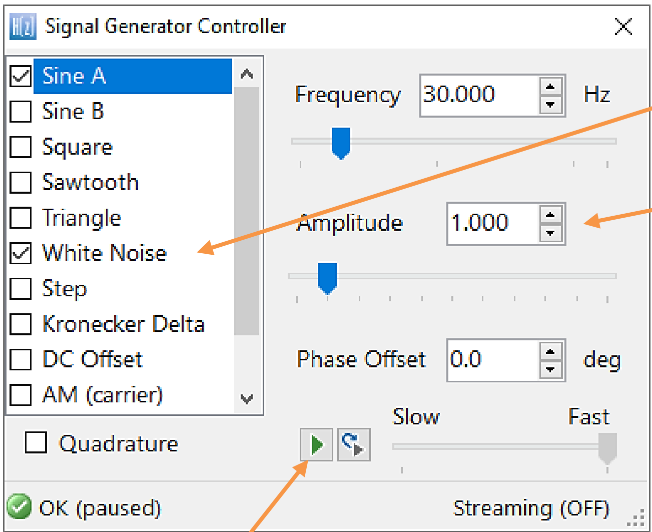


Set **Frequency** to `30.000`

Set **Amplitude** to `1.000`

(optional): if you want to generate a cosine instead of a sine, set **Phase Offset** to `90.0`

STEP 2



Select the **White Noise** method

Set **Amplitude** to an arbitrary value, e.g. `1.000`

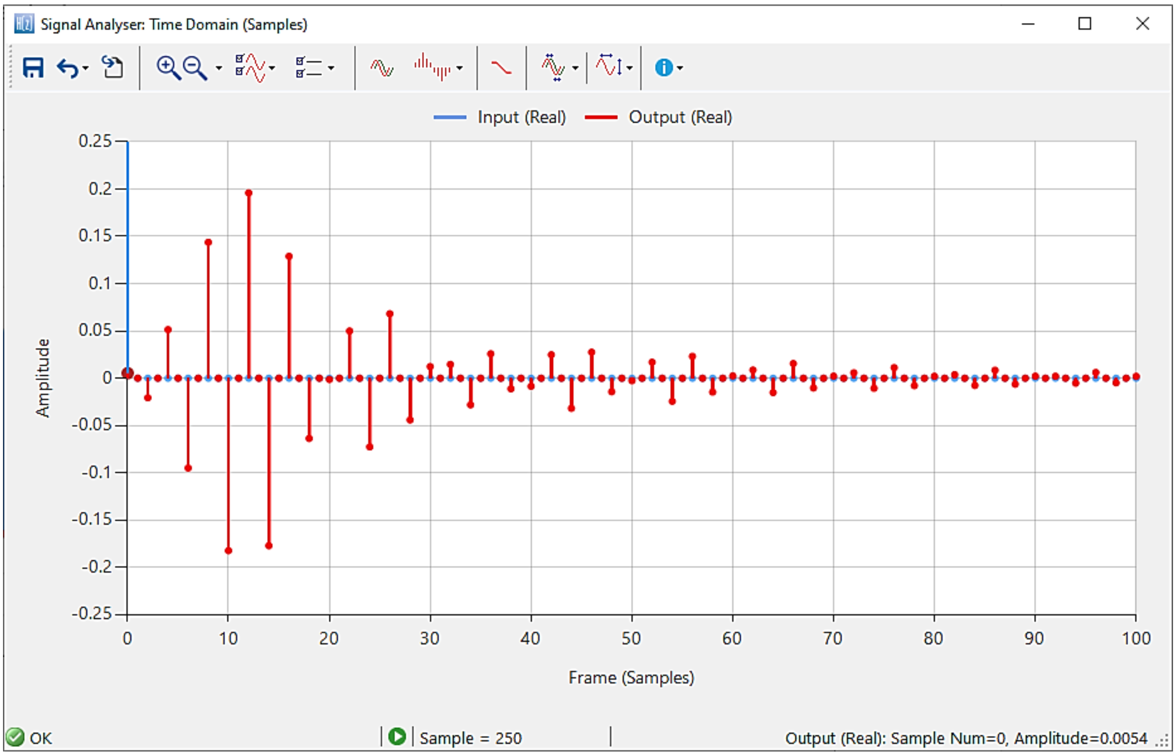
STEP 3

Play the signal generator.

4.2.3. Impulse and Step response shortcut

In order to expedite evaluation of the filter cascade's impulse or step response, a useful shortcut is available. The shortcut may be accessed by clicking on the  option in the toolbar.

By default, the impulse response is set to 250 points and the chart type is set to **Stem**. You may override these options in the chart options and setup menu.



4.3. Basic data analysis

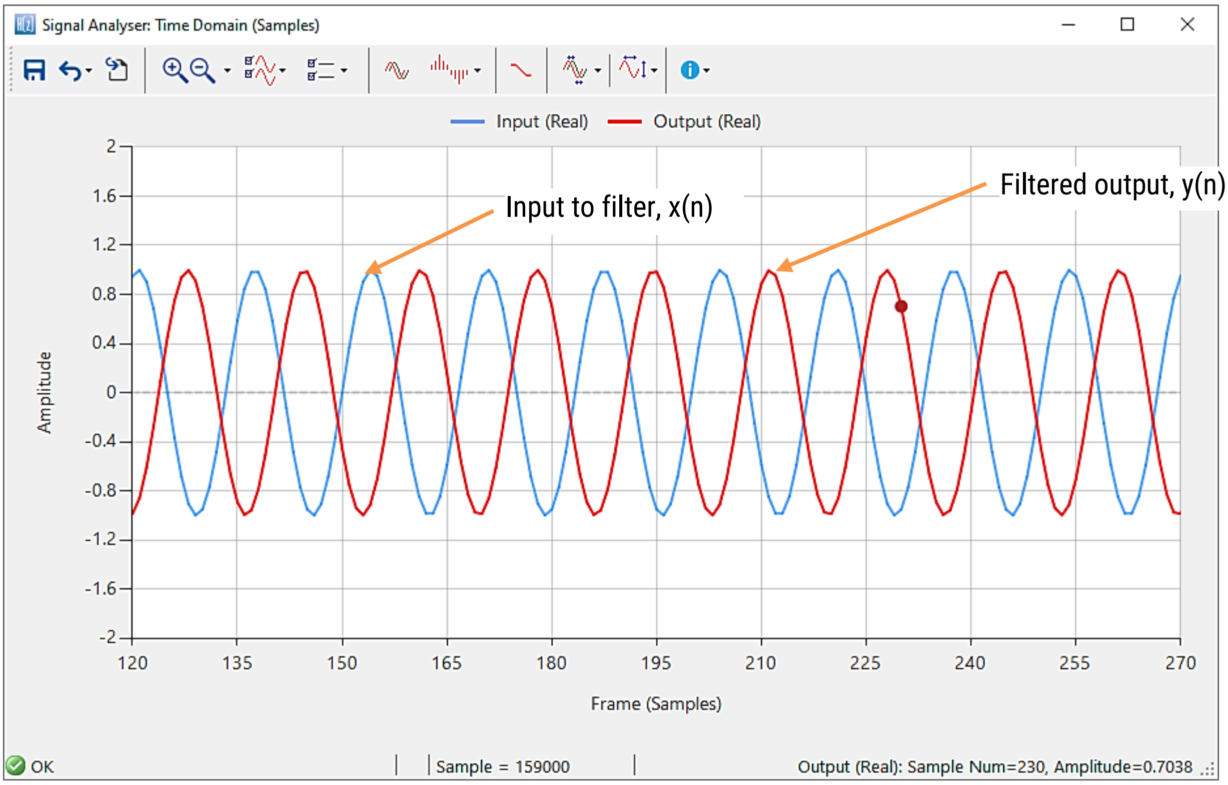
Basic data analysis is performed with the mouse. Where, moving the mouse over the chart will automatically produce data markers and data analytics (shown at the bottom right side of the GUI). The signal analyser is directly coupled to the filter designer GUI. This means that you may modify the filter characteristics and see the effects in real-time in the signal analyser. This functionality is very useful when designing audio filters, as the new filter settings can be heard immediately on the streaming audio feed as discussed later on in section 4.5.

When conducting frequency analysis, the data analysis algorithm implements a specialised version of the Discrete Fourier transform, which allows designers to perform high-resolution frequency analysis of any point of interest on the magnitude spectrum respectively.

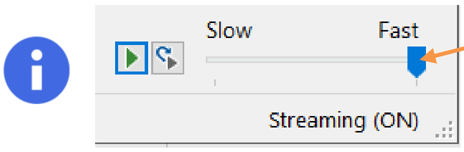
Use the **Left** and **Right** Arrow keys to move the data markers for a more fine-tuned analysis.

4.3.1. Time domain analysis

Upon clicking the signal generator's play button the signal analyser window will be updated.



As seen, the signal analyser resembles an oscilloscope, where live data from the signal generator is fed (streamed) into the H1 filter on a sample-by-sample basis. You may perform data analysis on the chart data by panning, zooming with the mouse.



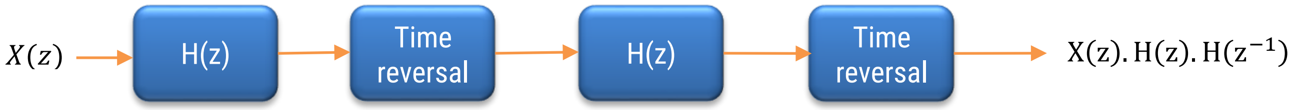
Signal generator playback speed (streaming only): You may adjust the chart update speed by setting the slider accordingly.

NB. The slider is disabled when streaming audio.

4.3.1.1. Zero-phase filtering

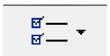
For biomedical developers, the tool offers designers the ability to implement zero-phase filtering . This is very useful for eliminating the effects of an IIR filter's non-linear phase, as it is set to zero.

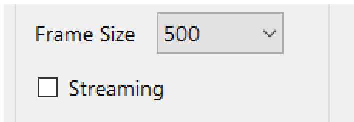
The zero-phase filtering operation is anti-causal and therefore cannot be run in real-time, as seen in the following block diagram:



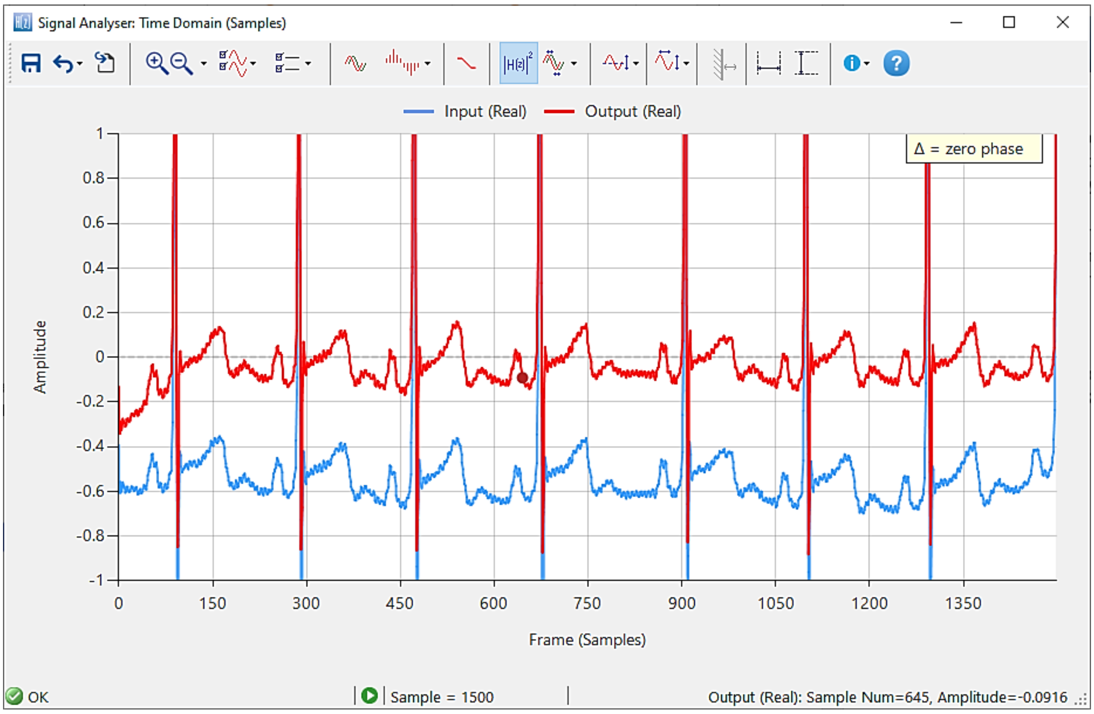
After enabling Block based mode (see below), and clicking on enable, the tool will automatically perform the zero-phase filtering operation on all filters in the cascade, i.e. H1, H2, Heq and H3. Although it should be noted that enabling this option will not modify the original filter transfer function displayed in the main design canvas.

The filtering effects on an input waveform are:
1. Zero phase distortion.
2. The net filter transfer function is equal to the squared magnitude of the original filter transfer function.
3. The net filter order is double the original filter order.

Block based mode should be selected by unchecking the **Streaming** checkbox in the Setup menu. 

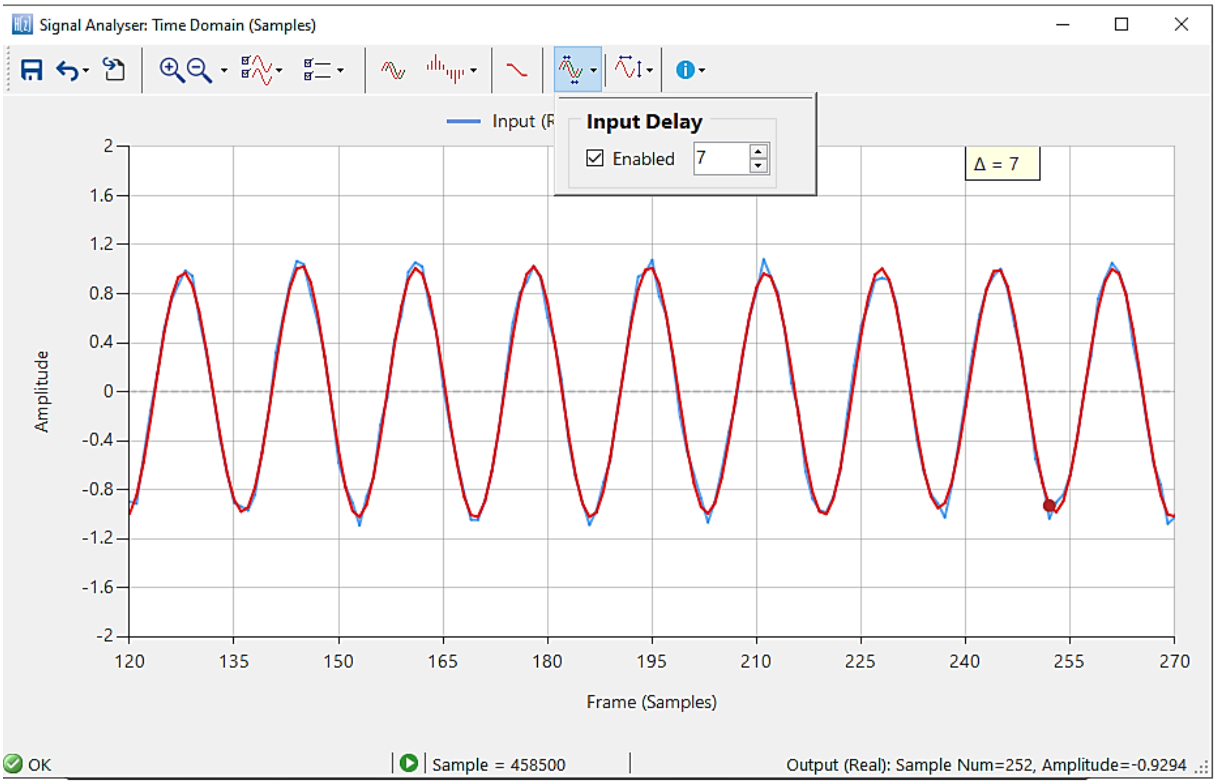


Example of zero-phase filtering with a 2nd order Butterworth IIR filter



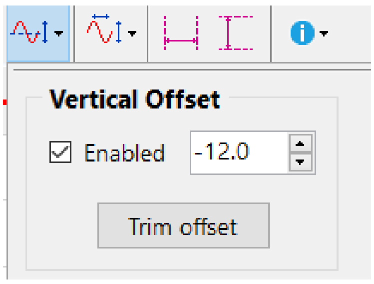
4.3.1.2. Delaying the input waveform

The tool offers designers the ability to delay the input waveform in order to visually compensate for the effects of phase/group delay.



Unlike zero-phase filtering, this feature does not affect the original waveform, and is only intended for display/analysis purposes.

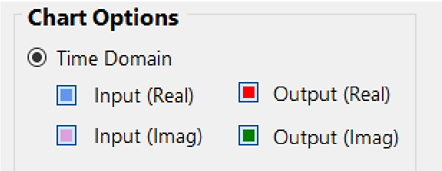
4.3.2. Trimming the DC offset



As some IoT datasets have large DC offsets (usually data taken from raw ADC values). The tool offers designers the ability to trim the DC for better analysis.

Clicking on the **Trim offset** button automatically removes the mean of the waveform.

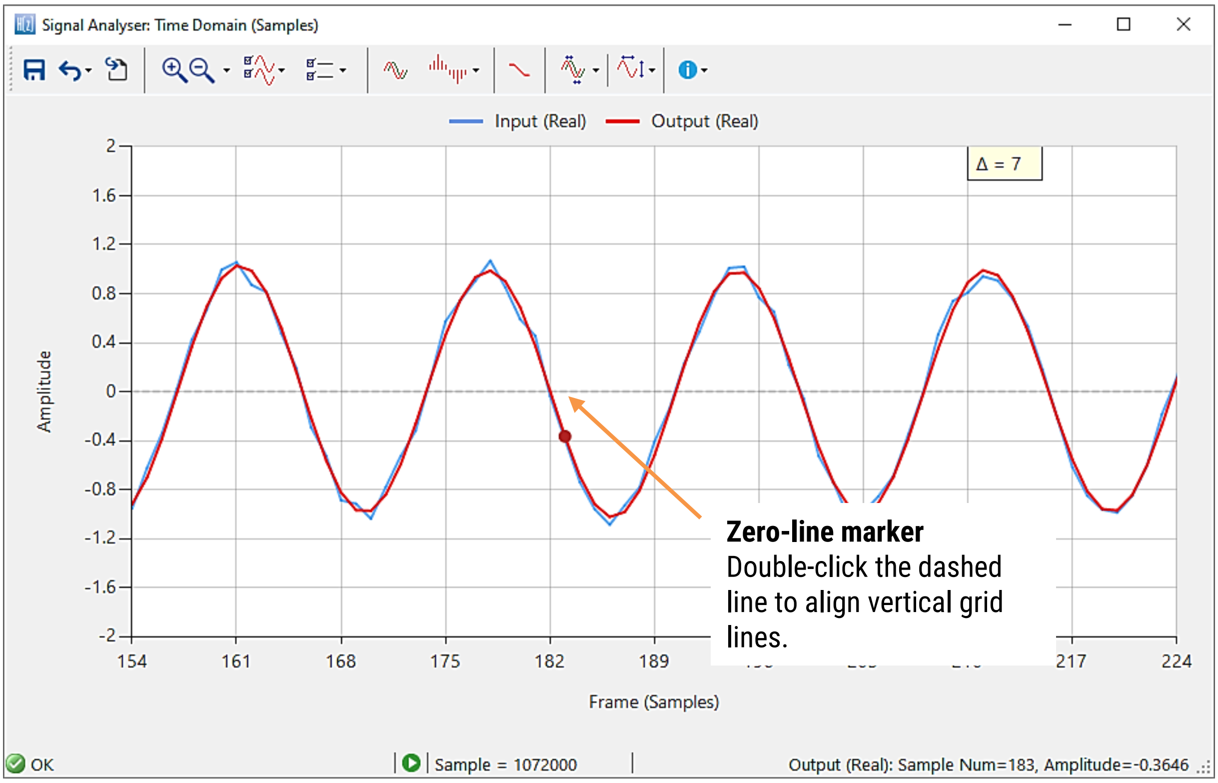
4.3.2.1. Input and Output waveform traces



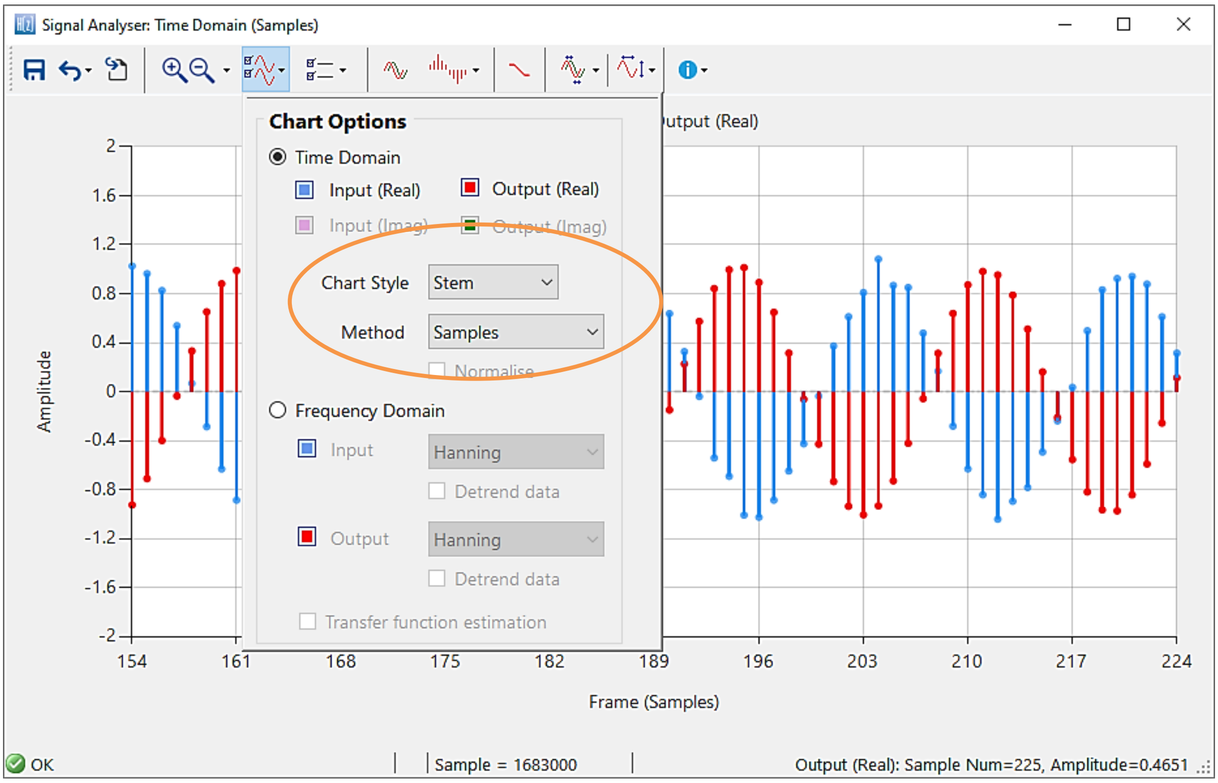
The signal generator automatically converts its output signal into a complex signal. The options **Real** and **Imag** refer to the real and imaginary components of the input and output signal respectively.

As complex filters are somewhat of a speciality, the default display setting is **Real** for both input and output signals.

4.3.3. The zero-line marker

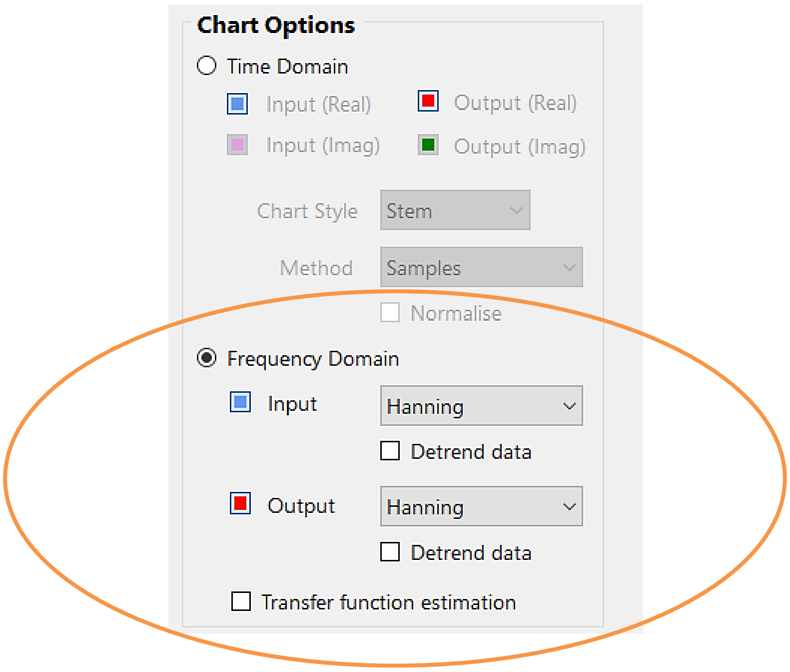


4.3.4. Stem chart

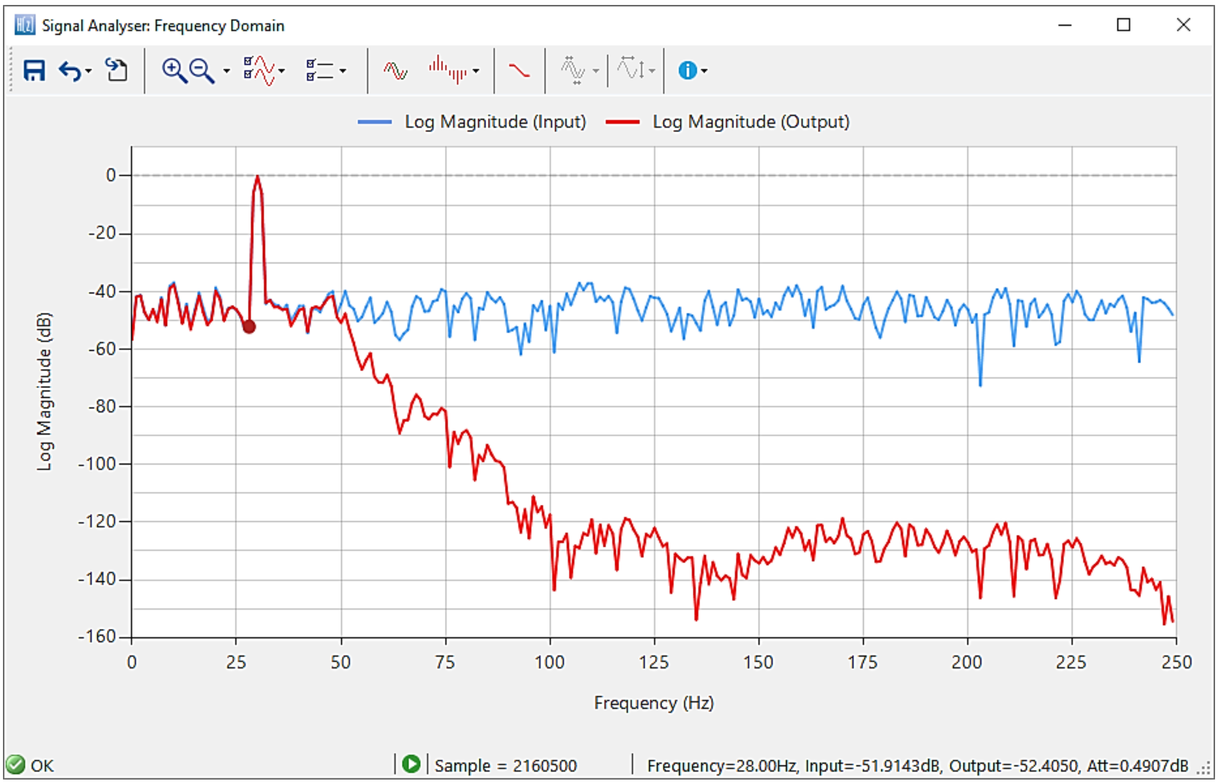


4.3.5. Frequency domain analysis

The signal generator's default view is time domain analysis, but you may also perform frequency domain analysis by altering the chart options, as shown below:

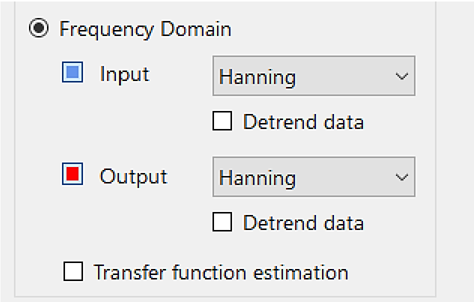


Detrending data: When performing frequency domain analysis, any low frequency information will be smeared by a large DC offset, e.g. biomedical data. In this case, the DC offset or data trend may be removed before windowing using the **Detrend data** option.



As with the time domain chart, you may perform data analysis on the frequency domain chart data by panning, zooming with the mouse.

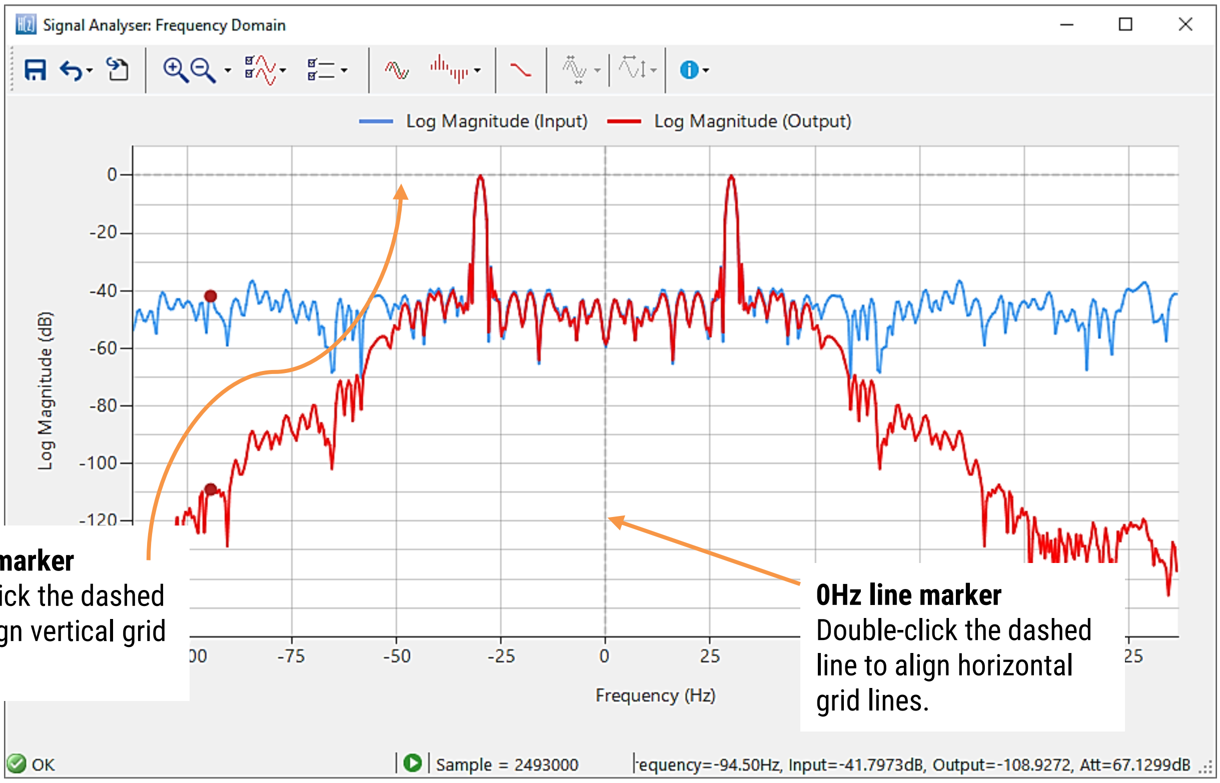
4.3.5.1. Input and Output waveform traces



Two traces are used for the frequency domain analysis (as seen on the left). Although when **Transfer function estimation** is enabled, only the **Output** (red trace) is shown.

The colours of the traces cannot be altered!

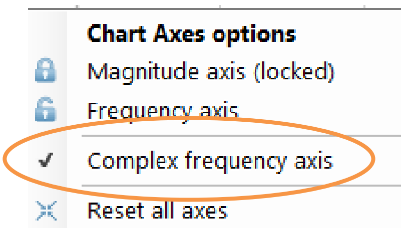
4.3.6. The 0dB and 0Hz line markers



0dB line marker
Double-click the dashed line to align vertical grid lines.

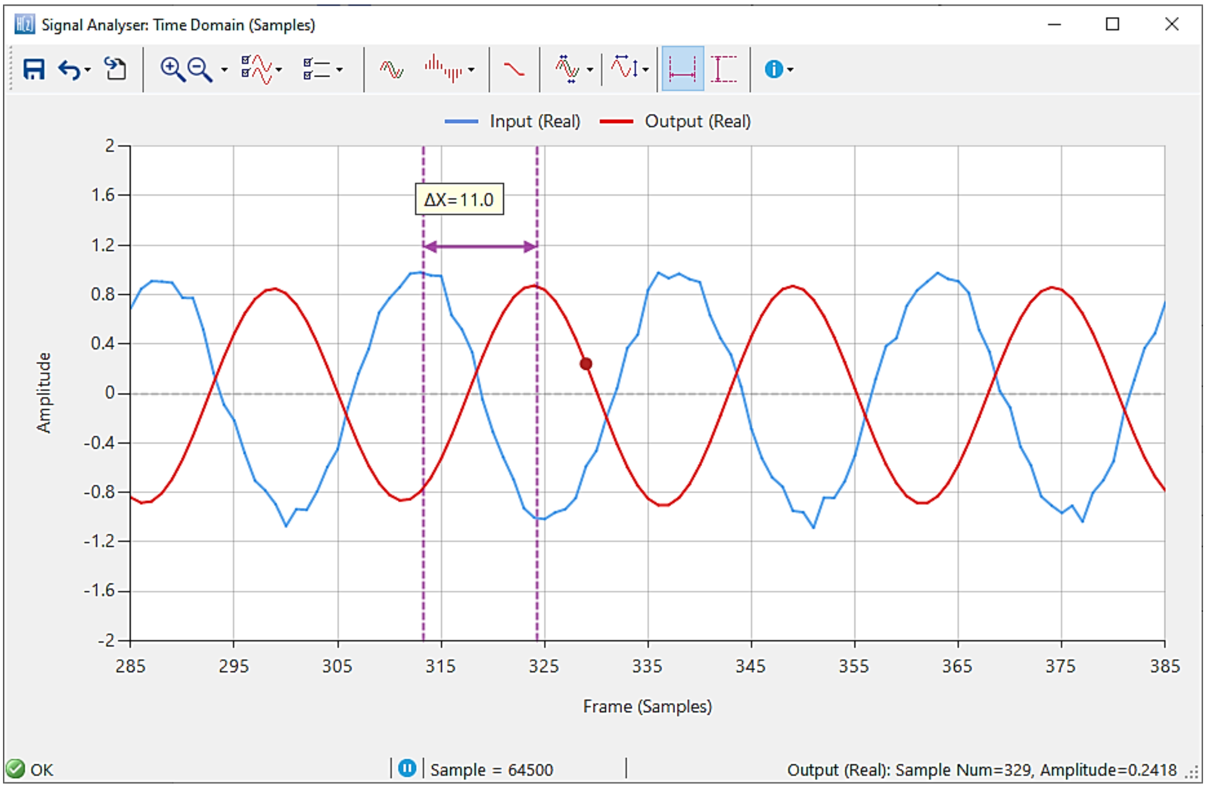
0Hz line marker
Double-click the dashed line to align horizontal grid lines.

The 0Hz marker is available for complex filters. You may enable or disable the complex analysis view via the **Chart Axes options** menu (Right mouse button → Complex frequency axis)



4.3.7. Data analysis rulers

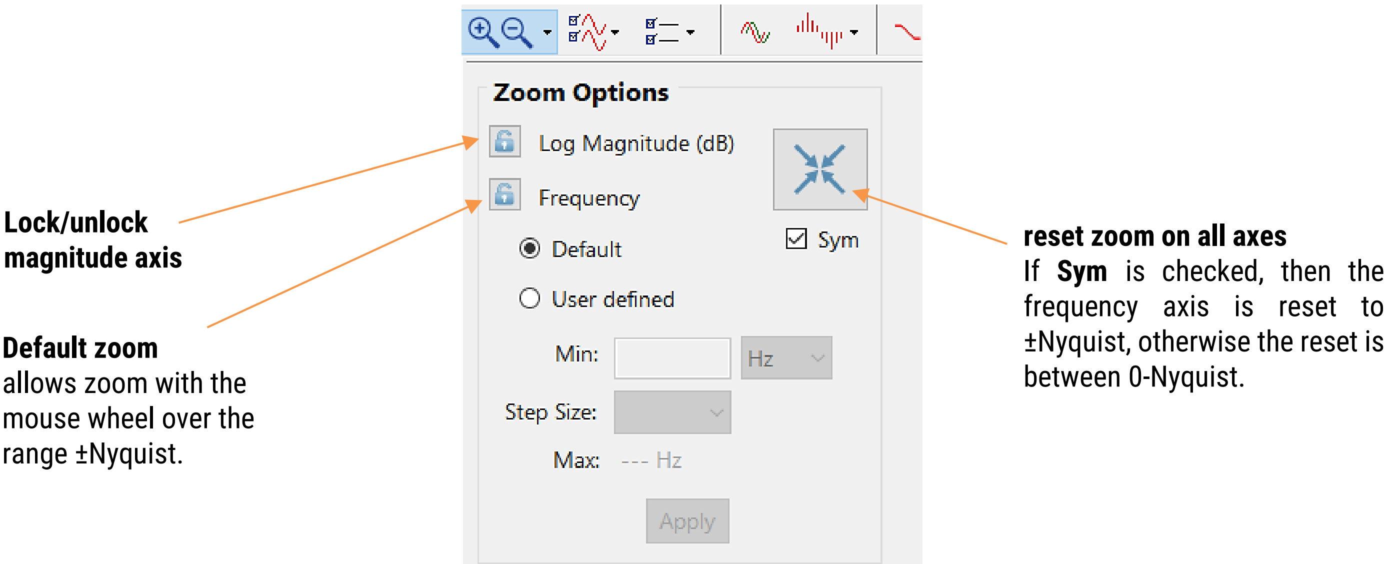
Two data rulers (in the horizontal and vertical plane) are available for more detailed data analysis.



Double-clicking on the ruler will move it to zero.

4.3.7.1. Chart zoom options

The signal analyser provides designers with a comprehensive zooming menu for undertaking analysis of demanding signals.



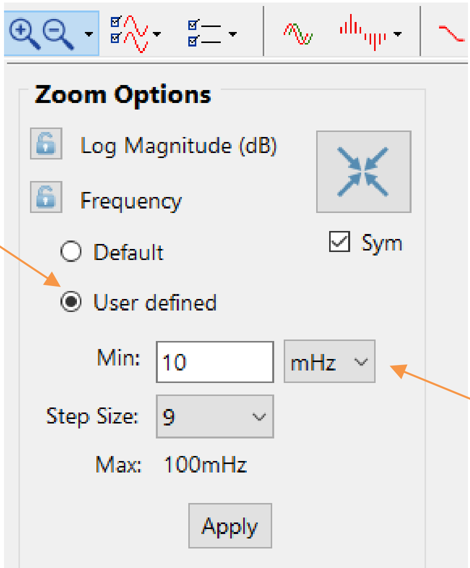
4.3.7.1.1. Locking axes

In order to simplify data analysis, you may lock a specified axis for zooming/panning purposes. This has the advantage of allowing you to customise each chart axis to your exact requirements.

4.3.7.1.2. Zooming to a specific frequency range

user defined zoom
You may zoom to a specific frequency range with the **User defined** zoom function.
The universality of this function allows you zoom to **mHz** resolution even when the sampling rate is in the **MHz** region!

This functionality is covered in the following [video tutorial]



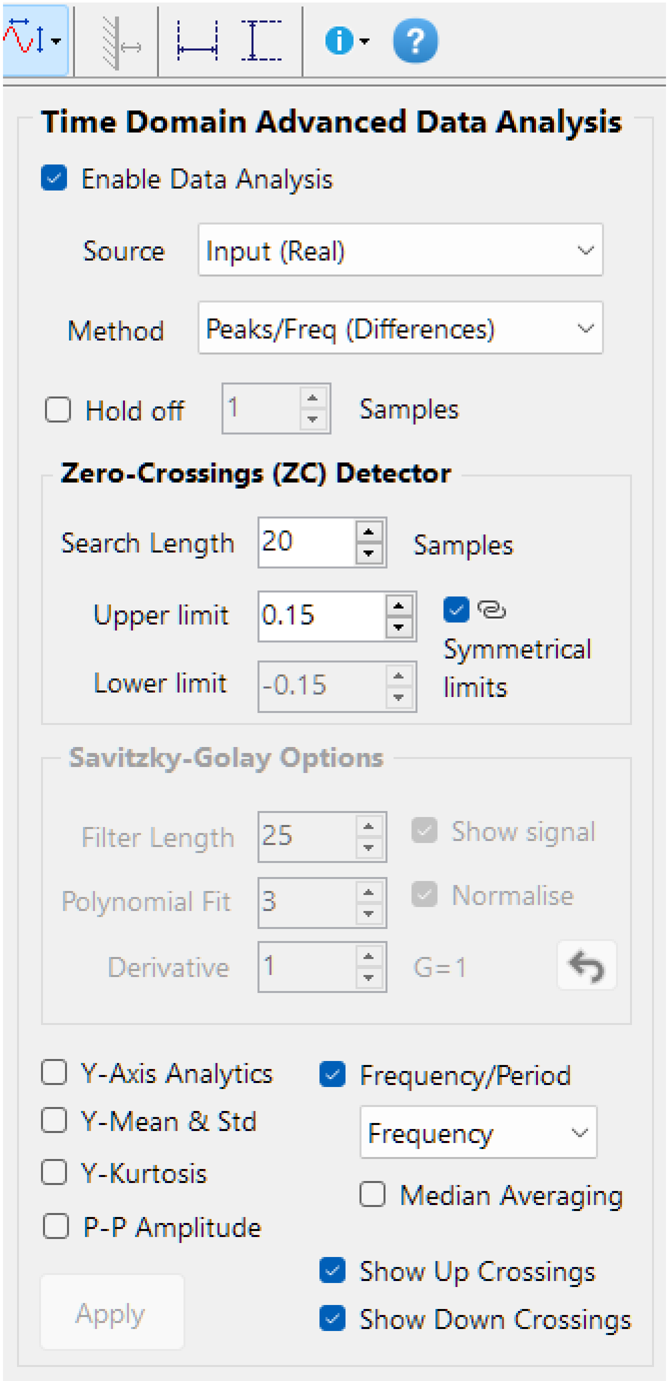
Choose the frequency scale that you wish to zoom to. Notice here that we are setting the x-axis to the range: `10-100mHz`.

Panning is disabled on the x-axis (frequency) when this function is enabled!

4.4. Advanced data analysis

The data analysis methods discussed in the previous section are suitable for a variety of simple tasks. However, for designers looking for more scientific analysis of their datasets, such as frequency measurement, the **Advanced Data Analysis** methods menu offer a collection of useful methods.

The Advanced data analysis UI is shown below, where it can be seen that the analysis methods are only available for time domain analysis. The in-built help should enable you to quickly set up and perform your analysis.

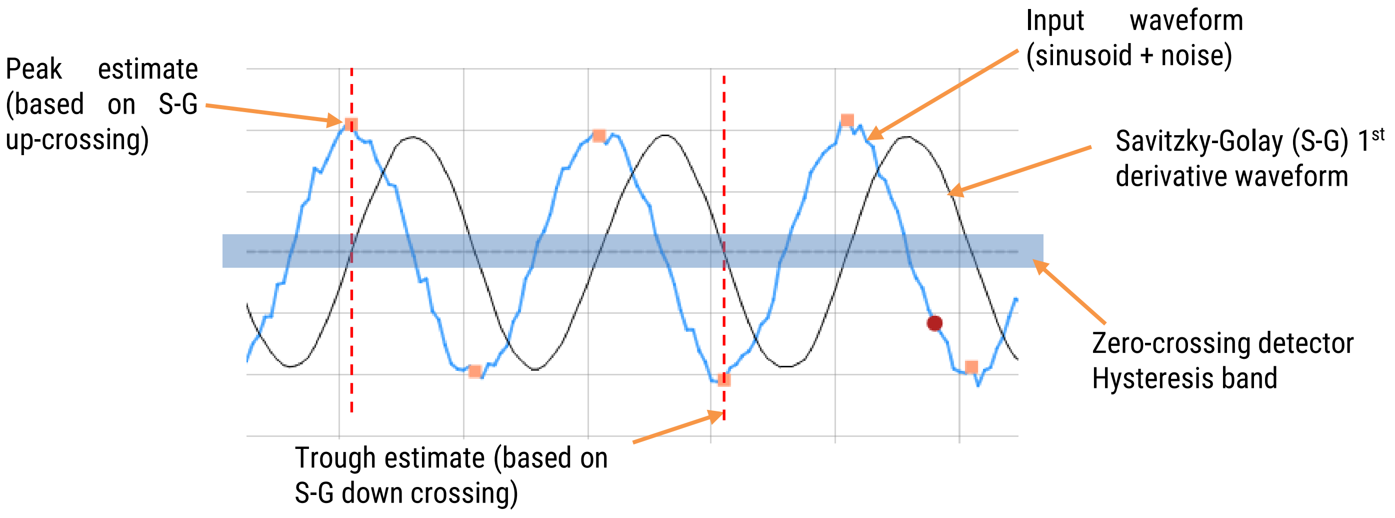


As seen, the GUI has a comprehensive list of options for a variety of algorithms as discussed in the proceeding section.

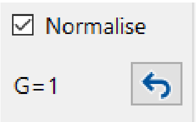
4.4.1. Analysis methods

The following analysis methods are implemented:

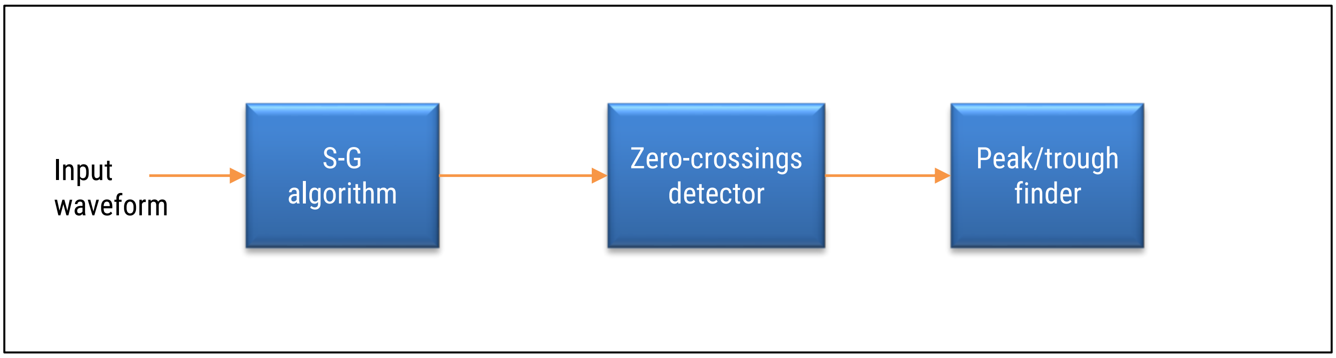
- **Y-analytics**: Display Y-Axis data analytics, such as Min, Max and ΔY.
- **Y-mean & Std**: display Y-axis mean and standard deviation.
- **Y-kurtosis**: Statistical measure of outliers in the probability distribution. Higher kurtosis represents more outliers, whereas low kurtosis represents less outliers. Good for ball-bearing wear and tear glitch-tracking in motor preventative maintenance applications using accelerometer data.
- **Peaks/Freq (differences)**: Determine peaks and troughs of a waveform using differences. This method is equivalent to Matlab's FindPeaks() method.
- **Peaks/Freq (Savitzky-Golay)**: Determine peaks and troughs of a waveform using a robust Savitzky-Golay differentiation filter. The waveform to be analysed is passed through the Savitzky-Golay algorithm for a specified **Derivative** (i.e. 1st, 2nd 3rd, 4th etc), **Polynomial Fit** and **Filter Length**. The algorithm is actually an FIR filter, fitting polynomials (order specified by **Polynomial Fit**) to data in order to provide a robust derivative estimate.
  Peaks and trough estimates are found by using the normalised amplitude of the Savitzky-Golay algorithm's output signal (shown below in black) and finding the zero-crossing positions – where an **up-crossing** pertains to a peak and a **down-crossing** pertains to a trough. The complete concept is illustrated below:



Scaling the Savitzky-Golay waveform: the **Normalise** checkbox enables automatic scaling (default), but you may override the scaling algorithm and set your own scaling factor. This is useful for biomedical datasets, where the automatic scaling may amplify undesirable artefacts.

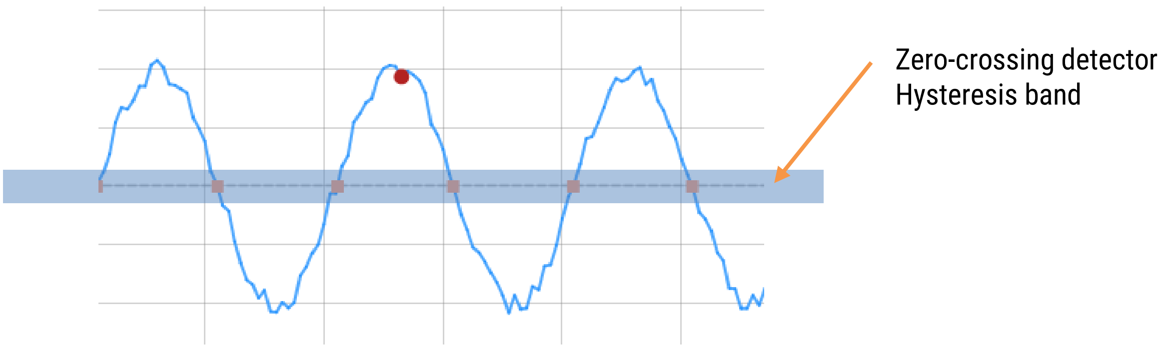


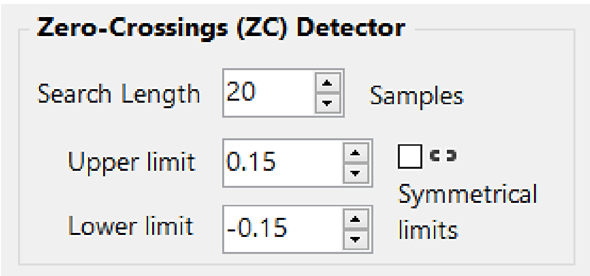
- The **zero-crossings detector** Hysteresis band is used to minimise the effects of noise on the peaks and troughs detection algorithm – *a signal must transition through the complete band in order to be accepted as valid*.



The peaks and troughs estimates are used for determining frequency, period, and min/max.

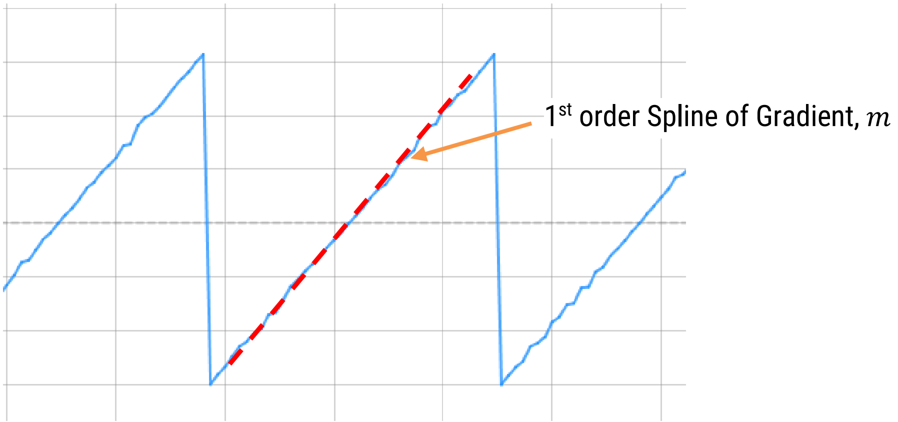
- **Frequency (zero-crossings)**: determine the frequency/period of a waveform using the zero-crossings information.





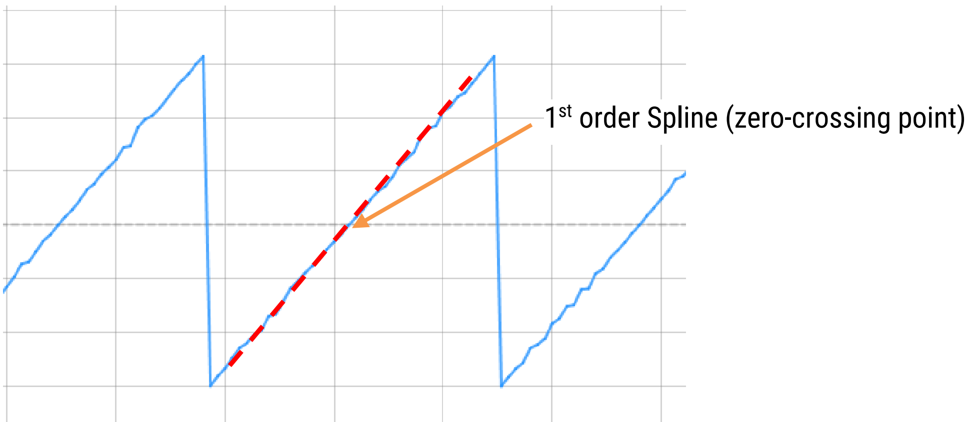
You may choose symmetrical (default) or set the upper and lower limits independently. The search length field is used for the ZC search algorithm, where a transition must occur within the specified search length time. For very slow biomedical data, such as PPG signals, this value should be increased to as much as 50 samples.

- **Frequency (phase)**: determine frequency/period of a waveform using its instantaneous phase information. This method is only applicable to complex sinusoids (i.e. analytic signal) using a complex filter. A first-order linear spline is fitted to the waveform per period between $\pm\frac{\pi}{2}$ in order to determine the average gradient as shown below:



An estimate of the sinusoid's frequency is given by: $\widehat{freq} = m \times \frac{f_s}{2\pi}$

- **Frequency (ZC)**: determine frequency/period of a waveform using its instantaneous phase information. This method is only applicable to complex sinusoids (i.e. analytic signal) using a complex filter. A first-order linear spline is fitted to the waveform per period between $\pm\frac{\pi}{2}$ in order to determine the zero-crossing point, as shown below:



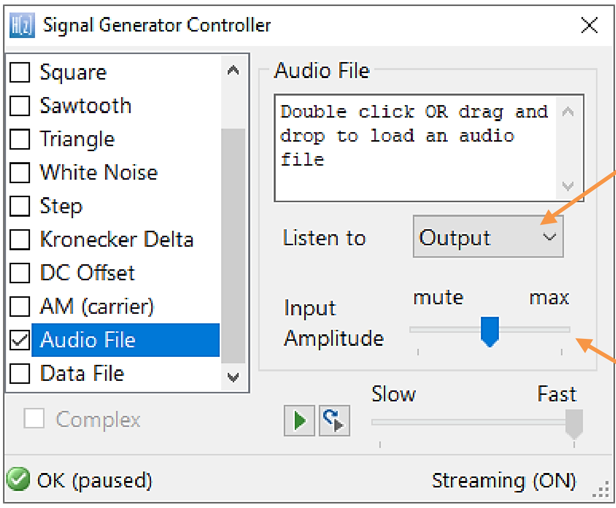
An estimate of the sinusoid's frequency is given by taking the median value over several frames of data.

- **Hold off:** this option allows users to ignore the specified number of samples before beginning the analysis. The Hold off method is especially useful in Block based mode, as the filter's initial output values can be ignored.

- **Median Filtering:** median filters are good for ignoring outliers. This option is good for biomedical applications, where cross-zero crossing estimates used for Heart beat estimation may be affected by glitches or motion artefacts.

4.5. Audio file

The signal generator allows you to load **.wav** audio files for playback via the **Audio File** method. Both mono and stereo formats are fully supported for the following sampling rates: `8.000`, `11.025`, `16.000`, `22.050`, `44.100` and `48.000kHz`. There is no restriction as to the length of the **.wav** file.



choose what you want to listen to (i.e. the input or output signal)

Adjust the amplitude of the input signal.

You may add extra signals to the input audio stream by enabling the methods as discussed above. If using the professional version, a maximum limit is placed on the filter order that can be evaluated. For an IIR filter, this is set at `30`, and for an FIR this is set at `300`.

You may import a wav audio file directly by dragging and dropping it into the analyser chart area.

All audio input signals are normalised and converted into floating point format for use with the signal generator. Adjusting the input amplitude to `>±1` will result in signal distortion.

If the sampling rate of the loaded audio file does not match the filter's sampling rate, you will receive a warning message. You may still continue with your experimentation, but with the understanding that the audio stream and filter are mismatched.

For higher sampling rates, such as 44.1kHz the UI may become sluggish on some computers. Internal analytics monitor the responsiveness of the UI and if deemed too sluggish, audio playback will be paused with a warning message in the signal analyser UI.

For converting other audio formats, such as mp3 and ogg into .wav the reader is referred to the freely available open source audio editing program Audacity (http://web.audacityteam.org/)

Audacity export option: Ensure that you choose **WAV(Microsoft)**, with **Signed 16-bit PCM** as encoding format.

4.6. External data file

The signal generator allows you to load external data files for playback via the **Data File** method. Two types of file format are supported: CSV (comma separated value), and single column data as shown below.

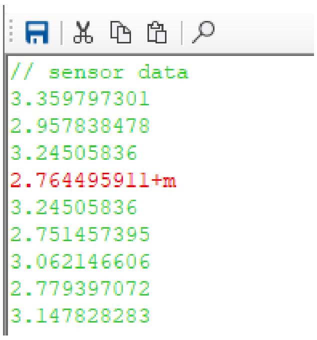
Data must be a single column text file and may contain real or complex values (**i** or **j**) and user comments (**//**).

Example 1	Example 2	Example 3

`// sensor data`
`59.8740`
`62.2261`
`59.8364`
`63.1592`
`59.9487`
`62.5620`
	`// sensor data`
	`59.8740+10j`
	`62.2261`
	`59.8364`
	`63.1592`
	`59.9487+i`
	`// marker data`
	`62.5620`
		`// sensor data`
		`59.8740+10i //begin`
		`62.2261`
		`59.8364`
		`63.1592`
		`59.9487-2.4i`
		`62.5620`

A maximum data length of `200,000` values may be loaded within a min-max data range of `±100,000`.

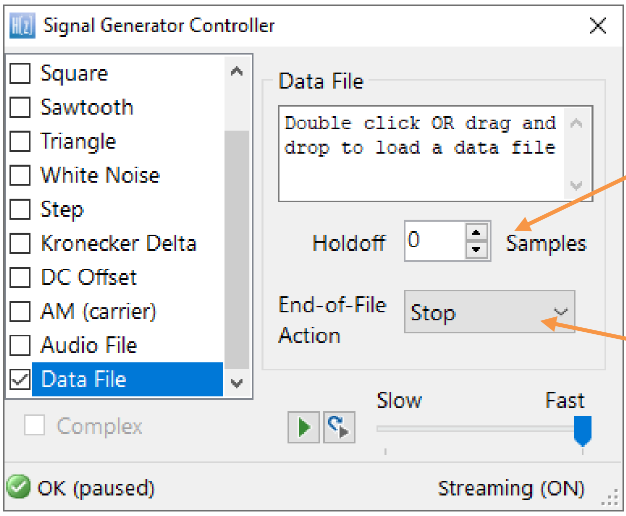
4.6.1.1. Correcting errors



If the import engine detects any errors (single column data file only), a file viewer window is opened and the error(s) highlighted in red (see left).

Use the toolbar options to edit the file, and then click on to re-save the file. After re-saving you need to import the file again.

4.6.1.2. Menu options



offset the data set by the specified number for samples: Use this option in conjunction with the **White Noise** method in order to pre-fill the filter's data lines with non-zero data.

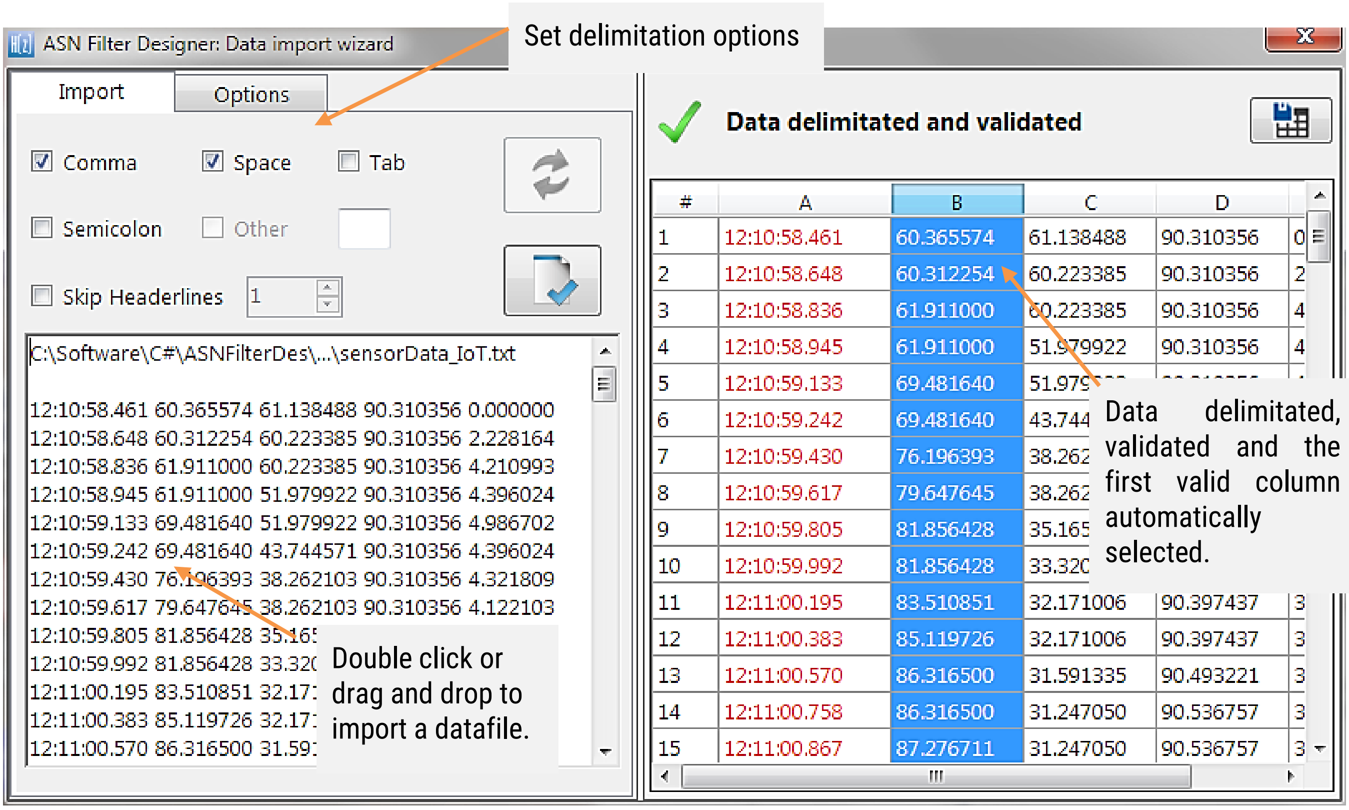
choose what action should be performed at the end-of-file:

- **Stop** playback
- **Repeat** from the beginning
- **Continue** streaming (if enabled)

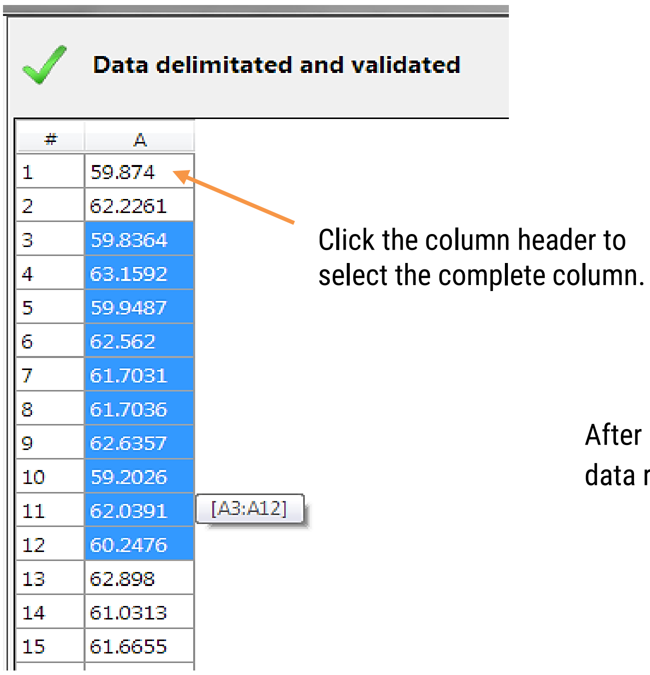
4.6.2. Importing data via the data import wizard

The signal analyser GUI incorporates an advanced data file import wizard . Text based data files of almost any format many be loaded and delimitated via the GUI.

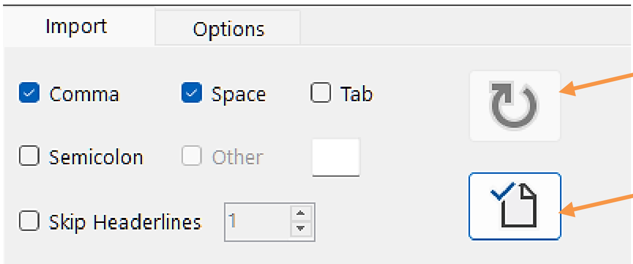
You may import a sensor file directly by dragging and dropping it into the analyser chart area.



NB. **Comma** and **Space** are automatically checked when a **.csv** file wildcard is detected.



After importing your dataset, you may use the mouse to select a data range that you wish to export.

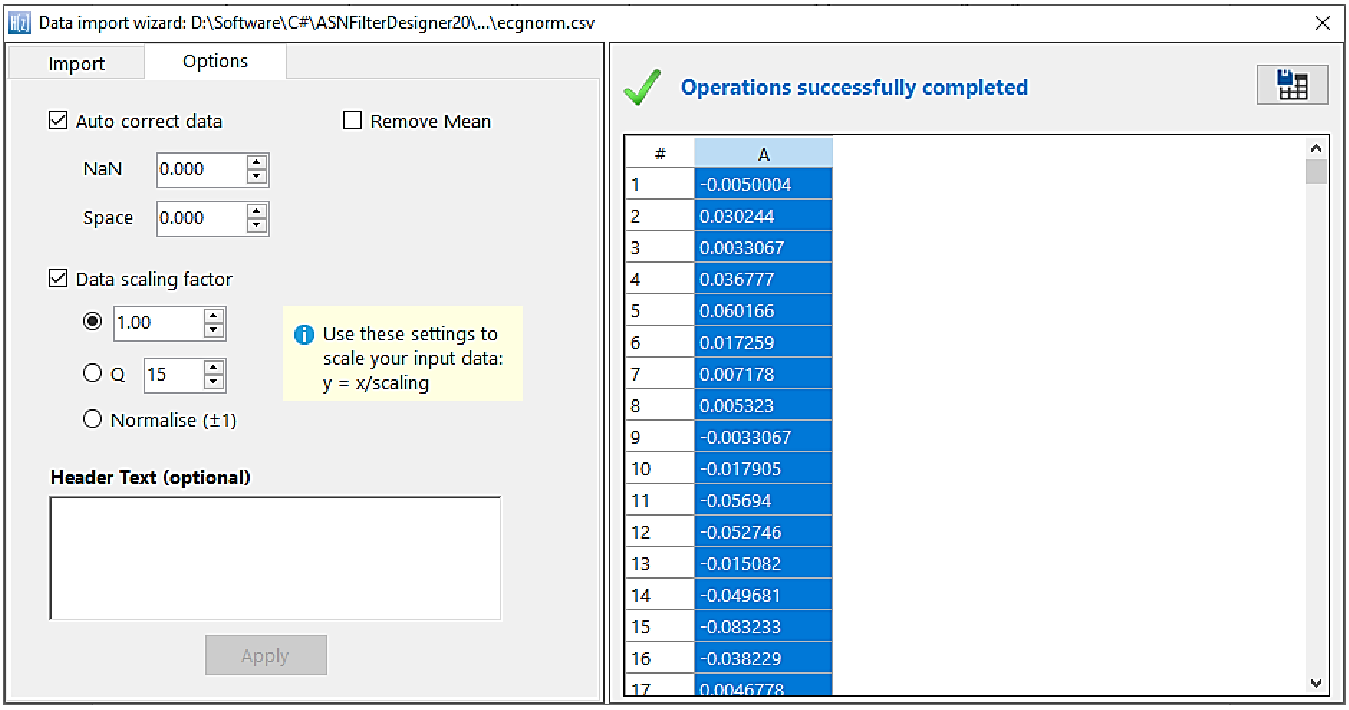


Click the 'refresh' button in order to re-load the file with the current (modified) delimitation settings.

After editing entries in the data table, you may re-validate your selected data set by pressing the 're-validate' button.

4.6.2.1. Data options

After selecting a data range, you may perform extra processing on the data, such as removing the mean, removing any invalid entries (NaN or a space) and scaling the data – see below.



Use the data scaling options to scale the data accordingly. For example, when dealing with ADC data in Q15, Q24, Q31 format, the 'Q' scaling option can be used to convert the ADC data back into floating point values.

An optional field is provided for entering **Header Text**, which is automatically placed at the beginning of the generated CSV file.

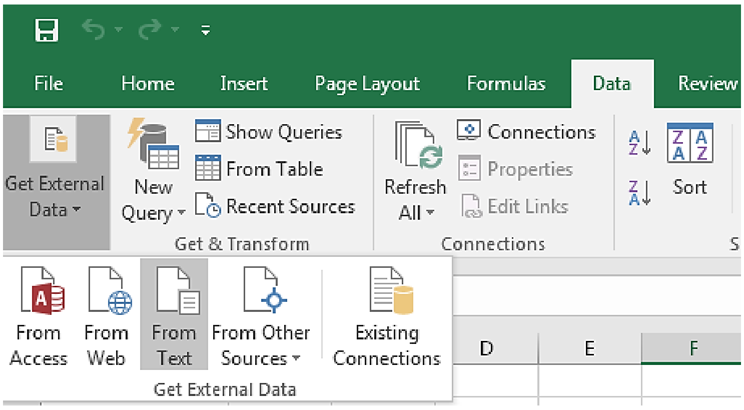
Finally, you may export (i.e. generate) a CSV data file by pressing on the  button.

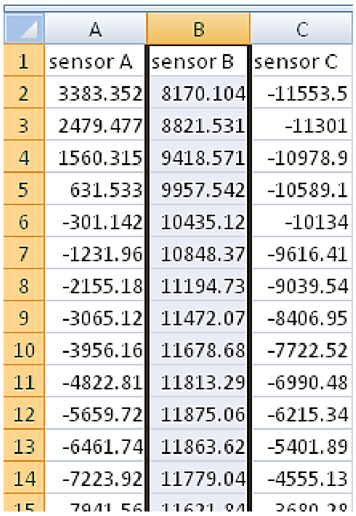
A video demonstration is available here.

4.6.3. Importing data from a Microsoft Excel spreadsheet

Many sensor datasets are available in Microsoft Excel spreadsheets. In order to export a spreadsheet's dataset to the ASN filter designer for analysis, the following steps should be followed:

1. Import the datafile and select the column of data that you wish to import into the filter designer:

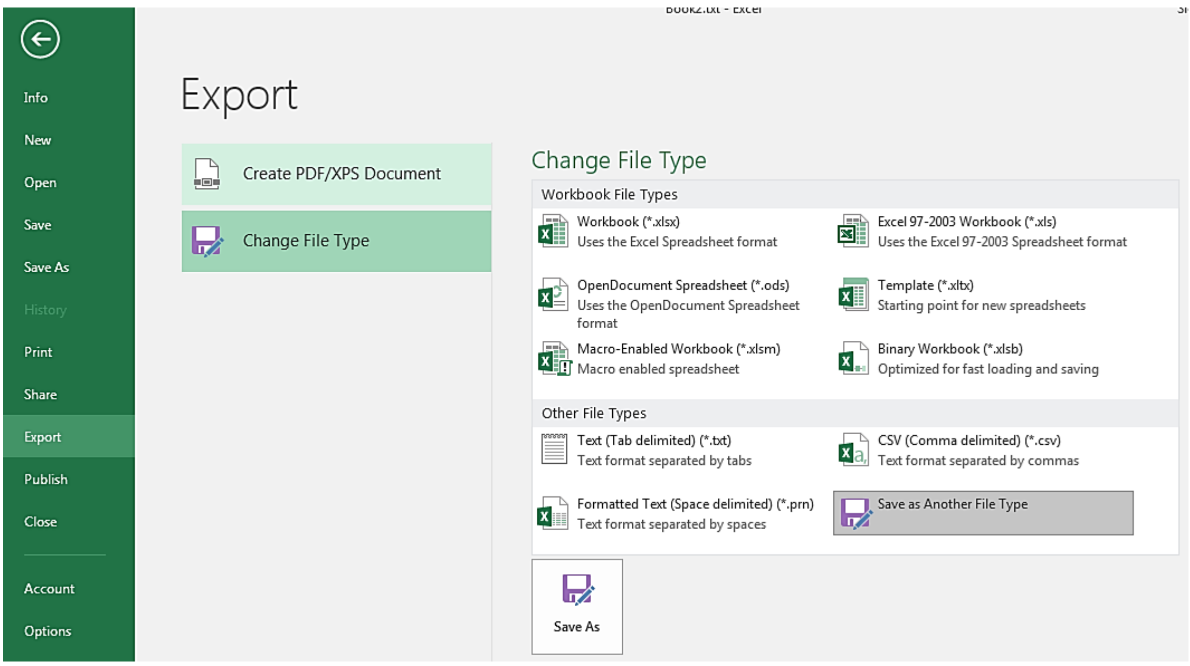




2. Paste the dataset into a new workbook.

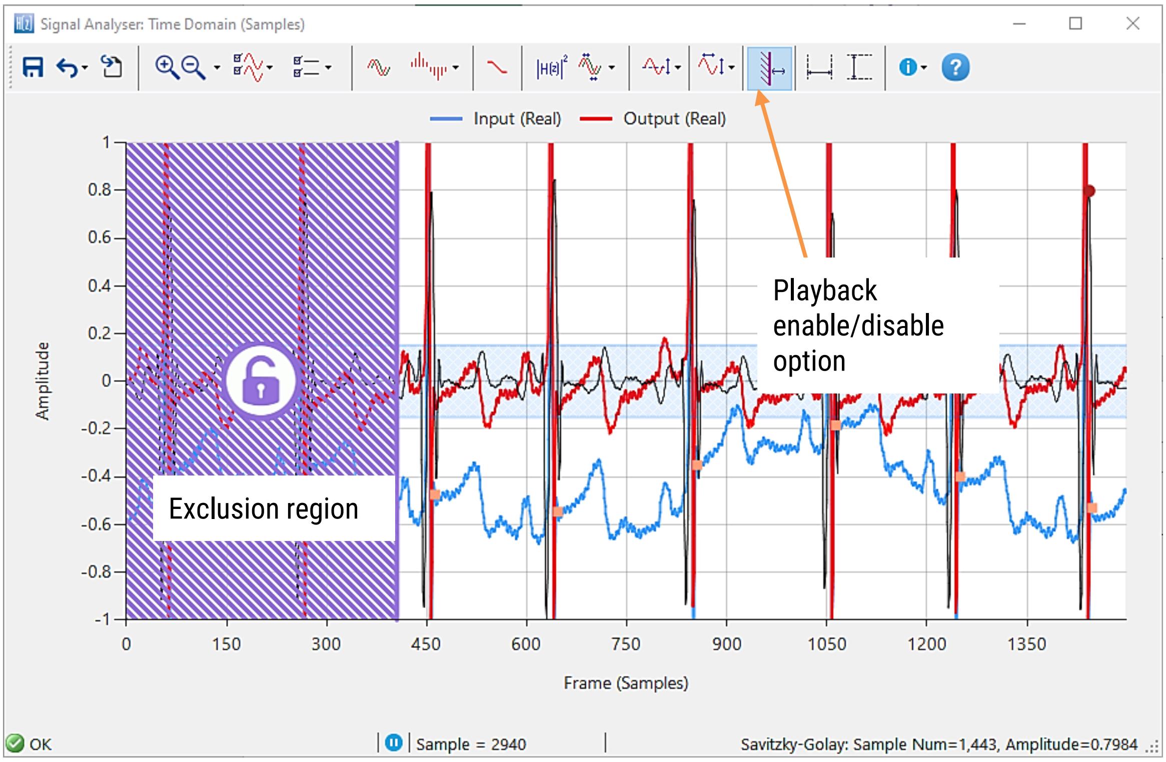
3. Save the new workbook as a Text file:

Export > **Save as Another File Type** > **Text (MS-DOS) *.txt**



4.6.4. Playing back an external datafile from a specific point

When analysing large datasets, it is handy to be able to analyse a specific part of the dataset. Therefore, the tool provides designers with an interactive method of selecting a start point within the dataset via the toolbar menu.



Use the mouse to select an exclusion region. After clicking on the play button, the exclusion region will be locked and the playback started.

The filter cascade will be automatically pre-filled with data from at least 1500 samples before the start point. This overcomes any filter group delay issues and ensures that the displayed results are always correct.

4.7. Amplitude modulation

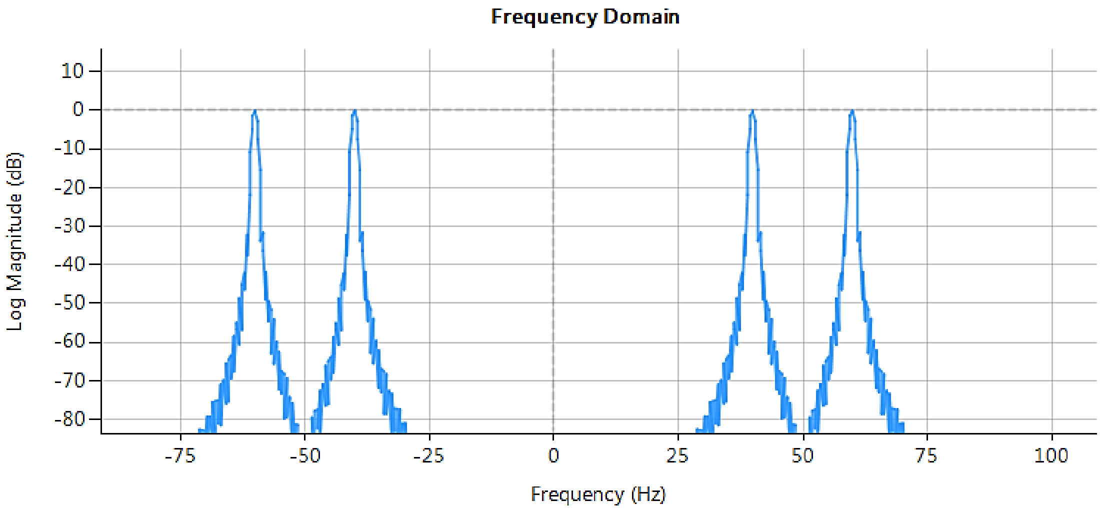
The signal analyser fully supports AM [amplitude modulation, **AM (carrier)**]. The type of AM implemented is the so called double sideband-suppressed carrier modulation given by:

$$y(n) = A_c \cos(2\pi f_c n / f_s)\, m(n)$$

where, $m(n)$ is the summed output of the other enabled signal generator output signals (e.g. **Sine A, Sine B, White Noise** etc.). The resulting output spectrum is given by:

$$Y(f) = \frac{1}{2} A_c [M(f - f_c) + M(f + f_c)]$$

The example shown below illustrates the resulting spectrum for a 10Hz sinusoid amplitude modulated with a 50Hz carrier signal ($A_c = 2$).



As expected, the resulting spectrum has two peaks (40Hz and 60Hz) centred around ±50Hz.

4.7.1. Practical application

AM has found particular use in the sensor world when performing accurate strain measurements using a loadcell sensor excited by an AC source. In such an application, the carrier frequency, f_c and an excitation sinusoid are the same frequency, and the phase offset (due to instrumentation electronics) between the two sinusoids is considered to be < 0.1 degree, which has minor impact on the estimate. Using the theory developed above, it can be seen that any unwanted DC offsets from an instrumentation amplifier and ADC also present in $m(n)$ will be moved to f_c and the desired sinusoid moved down to DC (0Hz). The amplitude of the sinusoid can now be easily extracted with a simple lowpass filter, which will smooth the output by filtering out the unwanted components higher up in the spectrum.

4.8. Setup

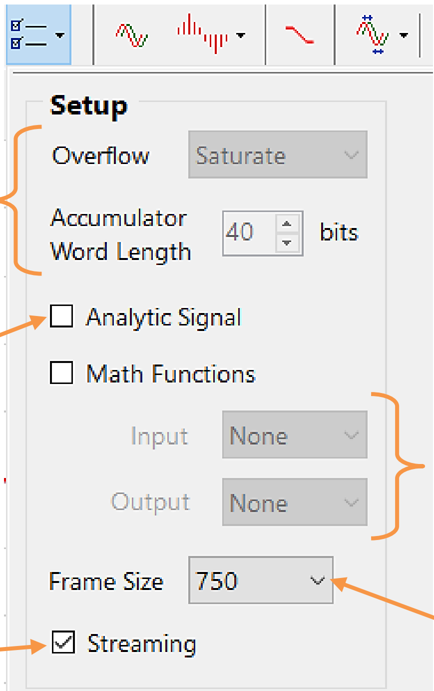
The Setup menu allows you to customise the Fixed point quantisation settings, input/output mathematical functions, frame size and select between streaming and blocked based mode.

Fixed point quantisation settings

- set the overflow rules: **Saturate** or **Wrap**.
- set the accumulator length: `40 bits` is the default.

Hilbert transform only
Automatically creates an analytic signal. *See [below](#) for more information.*

selects streaming or block based mode (see section 4.2.1 too).



I/O math functions
In order to condition the input and output data, six extra mathematical functions are available:
Abs, **Angle**, **Ln**, **RMS**, **Sqr** and **Sqrt**

Frame size
select analyser frame size, between `100-1500`.
NB. this setting will be automatically altered when streaming audio and computing the transfer function.

4.8.1. The Hilbert Transform and the analytic signal

Checking the **Analytic Signal** checkbox will automatically delay the input data stream by N/2 (where N is the filter order) and re-order the filter coefficients in order to produce an analytic signal. The delayed input signal (real component) is also pre-filtered with a first order Butterworth highpass filter in order to remove any DC components. Where, the cut-off frequency point (-3dB) of the filter is equal to one-fifth of Band 1's upper frequency value.

4.8.2. Block based and streaming mode

Block based mode will process a complete frame of input and output data and then reset the signal generator to its initial conditions. This functionality is extremely useful for instant evaluation of a sinewave's initial phase shift as well as instantly evaluating the filter's impulse and step response respectively. However, in many cases the near instant update will seem like the GUI has frozen or is inactive, therefore evaluation with the **White Noise** generator is recommended for users looking for a visual cue.

Streaming is the default setting and is used to assess real-time performance of the filters, where data from the signal generator is streamed (per sample) indefinitely. You may set the playback (chart update) speed by adjusting the playback slider on the signal generator, as discussed in section 4.2.

You may reset the signal generator by clicking on the **re-run** button.

When streaming audio the GUI enters a special type of streaming mode, whereby the playback slider and frame size controls are disabled.

4.9. Chart options

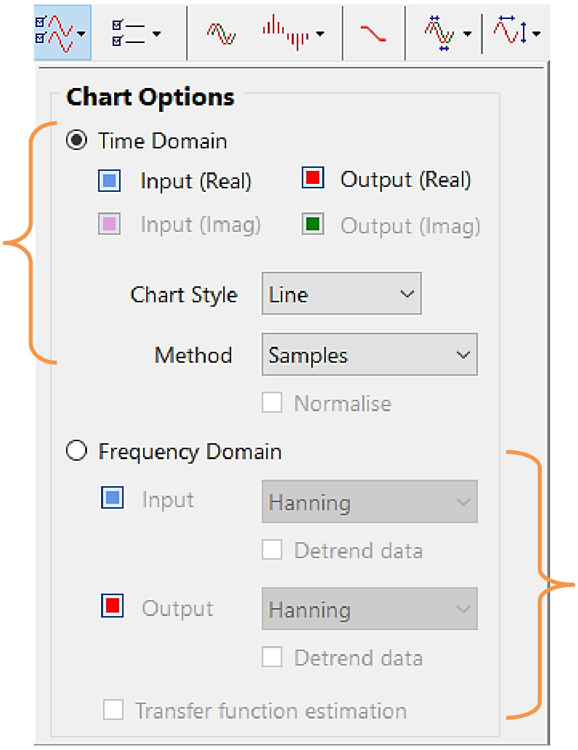
Chart options configure the chart for time or frequency domain analysis.

Time domain options
There are four chart traces (lines), representing the real and imaginary components of the Input and output signal respectively. You may enable or disable the traces that you wish to view.

Two lines styles are supported: **Stem** and **Line**.

Four analysis methods are available:

- Samples
- Phase
- Biased autocorrelation
- Real Cepstrum



Frequency domain options
There are two chart traces (lines), representing the log magnitude input and output spectra respectively. You may enable or disable the traces that you wish to view, and specify which smoothing window function is used.

Detrending data: When performing frequency domain analysis, any low frequency information will be smeared by a large DC offset, e.g. biomedical data. In this case, the DC offset or data trend may be removed before windowing using the **Detrend data** option.

Normalising data: When performing time domain analysis, the **Normalise** checkbox may be used for normalising **Autocorrelation** or **Real Cepstrum** data.

4.9.1. Biased autocorrelation

The biased auto-correlation is given by:

$$R_{xx}(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)x(n-k); k \geq 0$$

Autocorrelation is useful for finding periodic patterns, such as the period of sinewave buried in noise.

4.9.2. Real Cepstrum

The real Cepstrum is deconvolution technique heavily used in speech and audio applications. The essence of the Cepstral operation centres around use of the DFT (Discrete Fourier transform) and a log operator in order to deconvolve the transfer function (i.e. the slowly varying component) from the excitation (the faster moving component). The Cepstral deconvolution process may be described by the following block diagram:

Mathematically, considering $s(n)$ to be a convolution of an unknown transfer function, $h(n)$ and an excitation $e(n)$, we may write (where, FFT and IFFT are computationally efficient methods for computing the DFT and inverse DFT):

$$s(n) = h(n) * e(n)$$

$$FFT\ [s(n)] = H(w).E(w)$$

$$\log|S(w)| = \log|H(w)| + \log\ |E(w)|$$

$$IFFT\ \{\log|S(w)|\} = c(n) = IFFT\ \{\log|H(w)| + \log\ |E(w)|\}$$

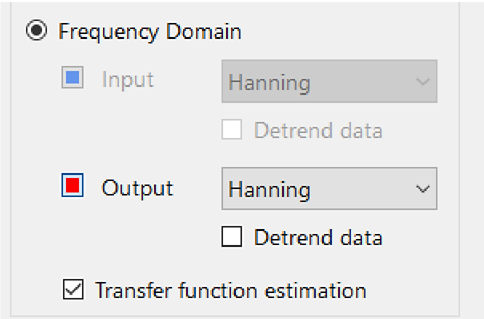
Entering the Cepstral domain, $c(n)$ notice how the transfer function and excitation are now *a linear combination* (i.e. additive) and such can be analysed separately. Notice also that unlike other pole-zero modelling methods, the Cepstrum may be used to model the effects of a system comprised of an unknown number of poles and zeros without any explicit knowledge of the system, as the analysis is non-parametric. However, care should be exercised when using this method for transfer function analysis, as no performance function is used, and as such the resulting coefficients are not strictly speaking optimal.

A new set of terminology was invented by the original author, and as such, 'frequency' was named 'quefrency' and 'spectrum' named 'Cepstrum'. The index of the Real Cepstrum (which is actually discrete time) is referred to as the quefrency axis.

4.9.3. Transfer function estimate

In order to validate the magnitude frequency response of the designed filter with sample data, the **Transfer function estimation** option is available. This method estimates the system transfer function by computing the quotient of the cross-power spectral density of **x** and **y** (i.e. the input and output) and the power spectral density of the input, **x** based on the Welch averaged periodogram method:

$$H_{xy}(f) = \frac{S_{xy}(f)}{S_{xx}(f)}$$



Enabling this functionality automatically sets the frame size to `1500`, enables the white noise signal generator and disables the input spectrum trace (although the data is still used in the computation). The averaging is performed over 10 frames using a **Hanning** window (default).

Care should be exercised when interpreting the results, as closely grouped poles/zeros may not appear to match the design specifications. Also, the results should only be interpreted as an estimation based on a window length of `1500` using a **Hanning** Window. Other types of Window functions will give different results, where the following functions are available:

Rectangular	**FlatTop**	**Hamming**
Blackman	**Lanczos**	**Hanning**
Blackman-Harris	**Gaussian**	**Chebyshev**

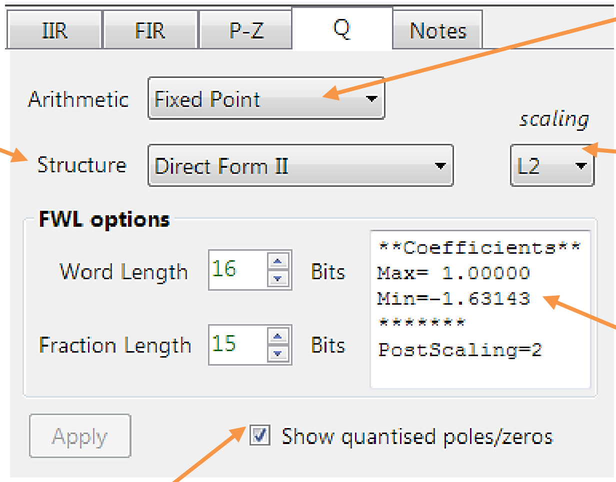
The Chebyshev window attenuation is fixed at -100dB.

5. H1 quantisation options and filter structures

The ASN filter designer provides designers with a rich assortment of quantisation analysis options for H1 filters.

Filter structure (used for implementation).

FWL (finite word length) composition.



Filter arithmetic used: **Double Precision**, **Single Precision** or **Fixed Point**

IIR Direct Form II filter scaling: **L1**, **L2** or **LInf** section scaling.

PostScaling: IIR biquad post scaling factor required for successfully implementing the current specifications.

Min/Max specifies the data range of the unquantised coefficients.

All quantised poles/zeros are shown in pale orange in the P-Z chart.

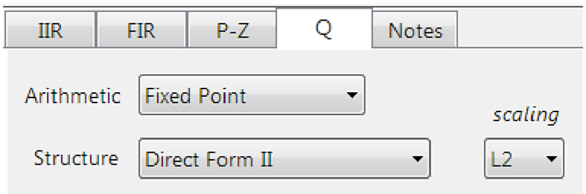
FIR Filters only: RFWL (Recommended finite word length) is a help analytic pertaining to the tool's Fractional Word Length analysis required for successfully implementing the FIR with the current specifications.

5.1. H1 Filter structures

You may experiment with various H1 filter structures and quickly assess your design's performance with different structures and quantisation settings.

H1 IIR structures:

- Direct Form I
- Direct Form II
- Direct Form II Transposed (default)



H1 FIR structures:

- Direct Form
- Direct Form Transposed (default)

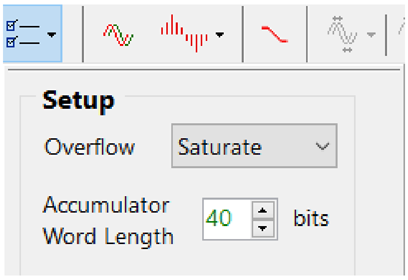
5.1.1. FWL (finite word length)

The system word length is split up into its *number of integer bits* and its *number of fractional bits (fractional length)*. Where, the general format is given by:

Q *number of Integer bits.number of fractional bits*

For example, if we assume that all of data values lie within a maximum/minimum range of ±1, we can use **Q0.15** format to represent all of the numbers respectively. Notice that **Q0.15** format is a 16-bit word, comprised of 1 sign bit, with a maximum of $2^{15}-1=32767$ and a minimum of $-2^{15}=-32768$.

Word Lengths of between 8-32bits may be implemented.



Accumulator word length options and overflow rules can be found in the signal analyser setup menu as discussed in section 4.8.

5.1.1.1. Direct Form II scaling

When implementing Direct Form II IIR filters, it is necessary to ensure that the feedback path, *w(n)* will not overflow (see section 2.4.3). The following scaling methods are available:

L1 norm	$G=\sum_{n=0}^{\infty}\lvert w(n)\rvert$	L1 norm assumes that the input is bounded and ensures that regardless of the type of input there will be no overflow. Needless to say, L1 scaling is extreme and should only be used when L2 or L∞ scaling is insufficient.
L2 norm	$G=\left[\sum_{n=0}^{\infty}w^2(n)\right]^{\frac{1}{2}}$	L2 norm places an energy constraint on the input and output transfer function. This scaling scheme offers the smallest scaling factor.
L∞ norm	$G=\max\lvert W(w)\rvert$	L∞ norm ensures that the filter will not overflow when a sine wave is applied.

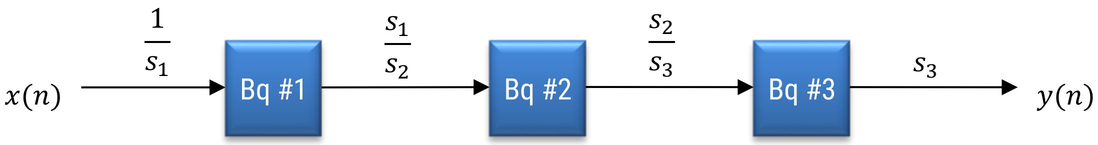
If $\lvert a_k\rvert\le 2$ and $\lvert b_k\rvert\le 2$ then the following difference equations may be used:

$$d(n)=\frac{x(n)}{G}-a_1w(n-1)-a_2w(n-2)$$

$$y(n)=G\times\left[b_0w(n)+b_1w(n-1)+b_2w(n-2)\right]$$

5.1.1.1.1. Biquad cascade scaling

A biquad cascade comprised of three biquads is shown below.



The scaling coefficients are given as s_1, s_2 and s_3 respectively. The filter designer tool automatically scales Bq#1's numerator coefficients by $G_1 \times \frac{s_1}{s_2}$, Bq#2's numerator coefficients by $G_2 \times \frac{s_2}{s_3}$ and Bq#3's numerator coefficients by G_3. The input scaling factor, $\frac{1}{s_1}$ and output scaling factor, s_3 are summarised in the filter summary under **Cascade Scaling Factors**. Where, the Input is actually given as s_1 instead of $\frac{1}{s_1}$ and **Output** is s_3. As a final point, rather than using the exact scaling factors, the values are actually rounded to the power of 2 (i.e. 2, 4, 8, 16 etc.) in order to simply the implementation.

5.1.1.1.2. Example

In order to fully understand the information presented in the ASN Filter Designer, the following example illustrates the filter coefficients obtained with double precision and with **Q1.14** quantisation.

```
Biquad #1
Gain = 0.022065
B = [ 1.00000000000,  1.42515458311,  1.00000000000]
A = [ 1.00000000000, -1.49439567907,  0.56622636801]

Biquad #2
Gain = 0.059997
B = [ 1.00000000000, -0.21088913424,  1.00000000000]
A = [ 1.00000000000, -1.56831045118,  0.67755833899]

Biquad #3
Gain = 0.122786
B = [ 1.00000000000, -0.77860154757,  1.00000000000]
A = [ 1.00000000000, -1.71966704418,  0.87471907332]
```

double precision

Applying L2 scaling with Q1.14 (note the effects of quantisation on the coefficient values), we obtain ($s_2 = 4$):

```
Cascade Scaling Factors
Input = 8
Output = 8
```

$$G_1 \times \frac{s_1}{s_2} = 0.022065 \times \frac{8}{4} = 0.0441$$

```
Biquad #1
Bq = [ 0.04412841797,  0.06286621094,  0.04412841797];  [ 723,  1030,  723]
Aq = [ 1.00000000000, -1.49438476563,  0.56622314453];  [ 16384, -24484,  9277]
```

$$G_2 \times \frac{s_2}{s_3} = 0.05999 \times \frac{4}{8} = 0.0299$$

```
Biquad #2
Bq = [ 0.02996826172, -0.00634765625,  0.02996826172];  [ 491, -104,  491]
Aq = [ 1.00000000000, -1.56829833984,  0.67755126953];  [ 16384, -25695,  11101]
```

$$G_3 = 0.1228$$

```
Biquad #3
Bq = [ 0.12280273438, -0.09558105469,  0.12280273438];  [ 2012, -1566,  2012]
Aq = [ 1.00000000000, -1.71966552734,  0.87469482422];  [ 16384, -28175,  14331]
```

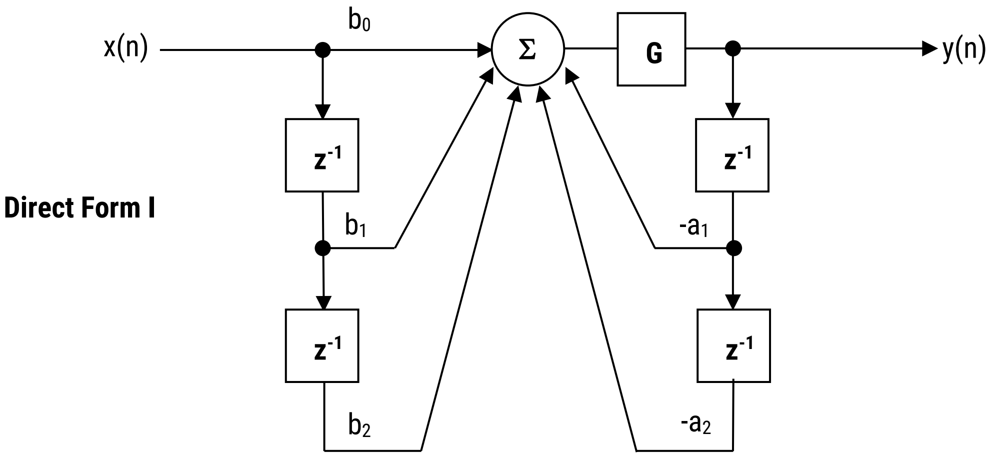
decimal coefficient value:
$\text{ceil}\,(2^{14} \times 0.02996) = 491$

The decimal coefficients may be directly inserted into a fixed-point algorithm for implementation.

5.1.2. Post Scaling Factor

In order to ensure that coefficients fit within the **Word length** and **Fractional length** specifications, all IIR filters include a **Post Scaling Factor**, which scales the numerator and denominator coefficients accordingly. As a consequence of this scaling, the Post Scaling Factor must be included within the filter structure in order to ensure correct operation.

The Post scaling concept is illustrated below for a Direct Form I biquad implementation.



Where, each coefficient is now scaled by G, i.e. $b_0 = \frac{b_0}{G}$, $b_1 = \frac{b_1}{G}$ and $a_1 = \frac{a_1}{G}$ etc. This now results in the following difference equation:

$$y(n) = G \times [b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) - a_1 y(n-1) - a_2 y(n-2)]$$

All IIR structures implemented within the tool include the Post Scaling Factor concept. This scaling is mandatory for implementation via the Arm CMSIS-DSP framework.

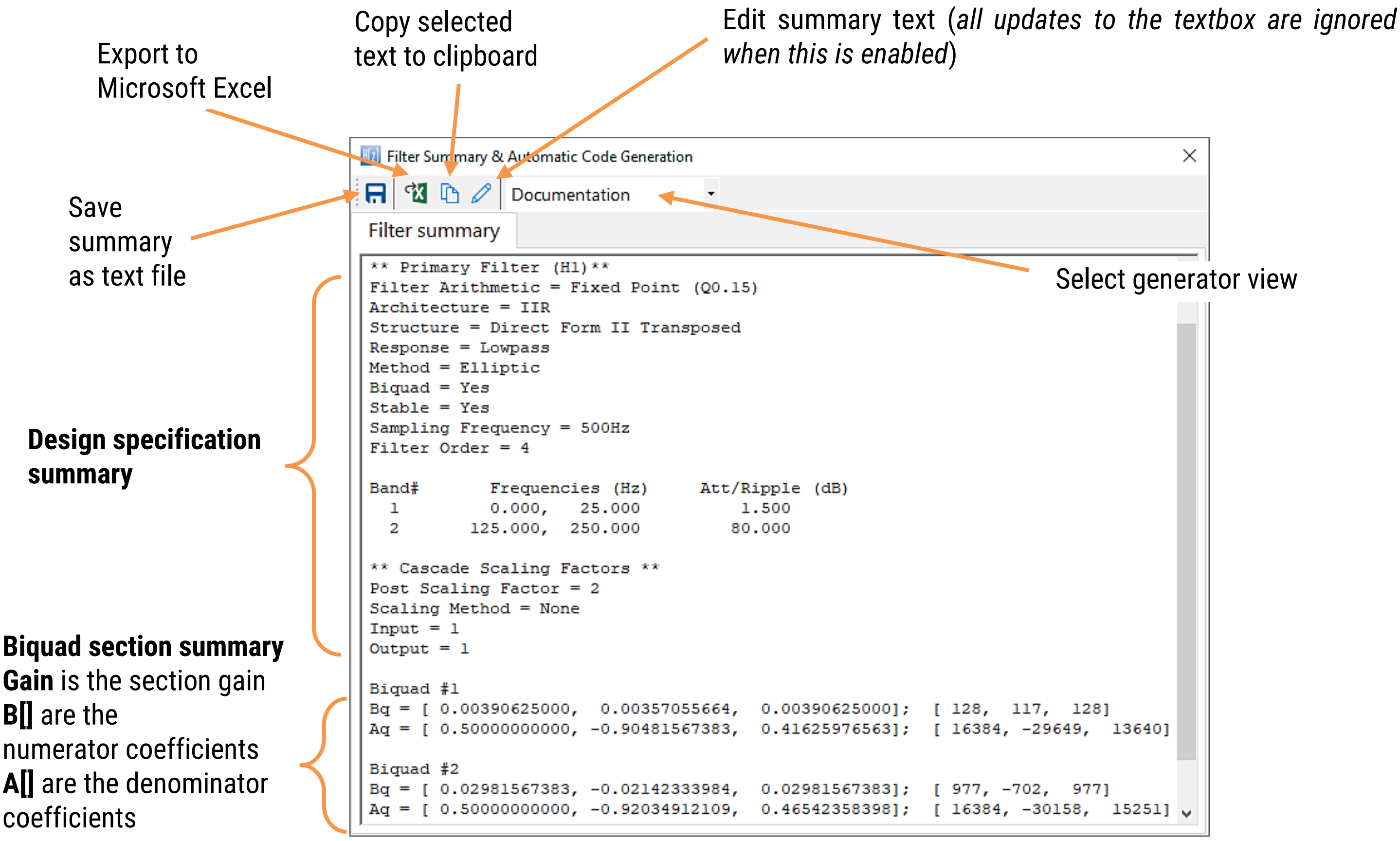
5.2. Filter summary and Automatic code generation

The filter summary presents the designer with a detailed summary of the filter coefficients and technical specifications used for the design. These details may be used for official documentation purposes in Microsoft Word or PowerPoint and provide a simple way of producing professional documentation within minutes.

The GUI also implements automatic code generation to various third-party applications, such as ANSI C and Matlab for further analysis or integration. The following export formats are supported:

- Python
- Matlab/Octave
- Scilab
- ANSI C
- C# .NET
- Arm CMSIS-DSP
- Xilinx Vivado

A detailed overview of each framework can be found in Support for 3rd party software development frameworks.

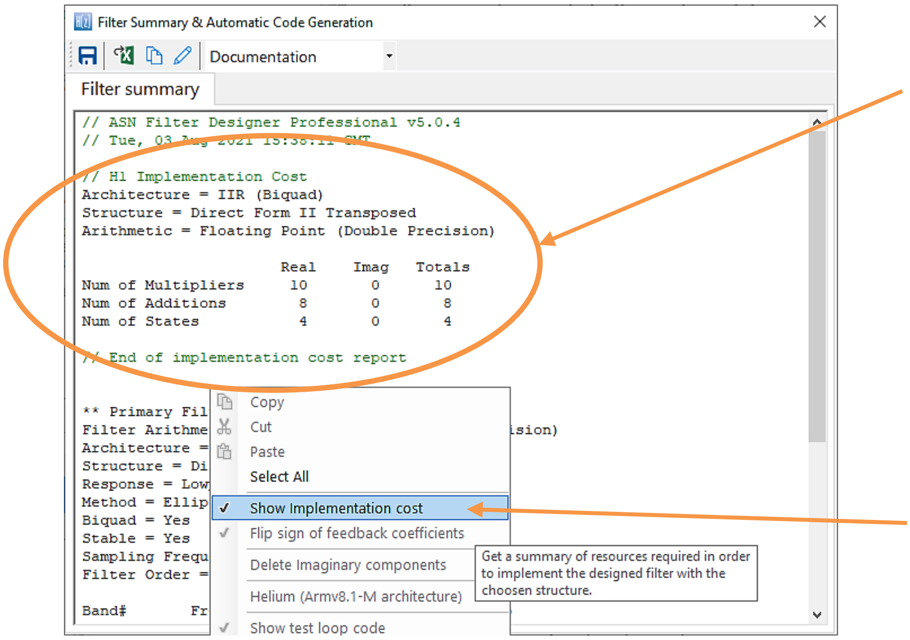


If the design is modified via the P-Z editor, the **Response** string changes to **User Defined** and the **Method** string is removed as the design does not adhere to a prototype design method anymore.

In order expedite the design and integration phase with 3rd party design tools, such as Analog Devices' SigmaStudio, registered users may export the filter summary to Microsoft Excel.

The export feature is only available in *Documentation* mode.

5.2.1. Implementation cost summary

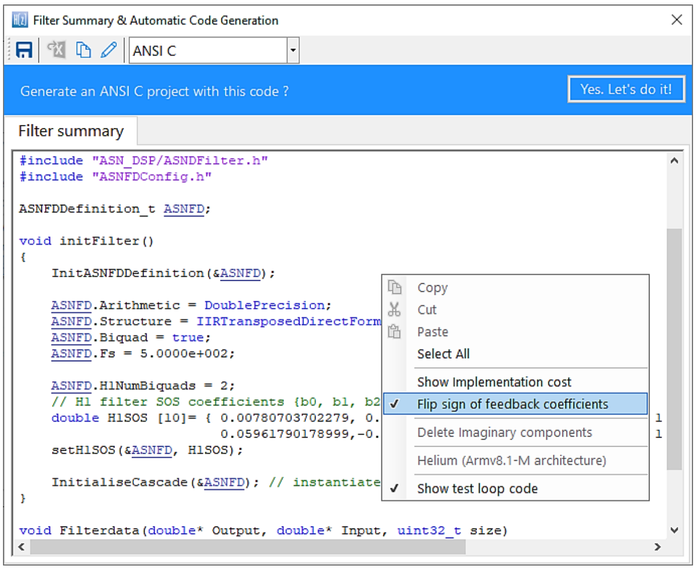


Documentation only

In order to assist with implementation in embedded devices, such as an FPGA, microcontroller or DSP, an implementation cost is available. This summary gives the designer a quick overview of the number of summers, multiplications and state variables needed for implementing the designed filter.

You may show or hide (default) the implementation cost summary by setting the option in the context menu (right mouse button).

5.2.2. Flipping the sign of the feedback coefficients



ANSI C and C# only

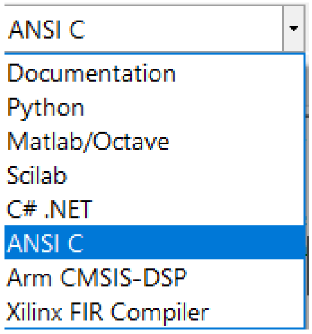
In order to provide a degree of flexibility with different technologies (FPGA, ASIC etc), you may enable (default) or disable the flipping of the sign of the feedback coefficients. This inversion means that additions need only be used, rather than subtractors.

If coefficient feedback is disabled, you cannot deploy your C or C# project via the ASN framework.

5.2.2.1. Deleting Imaginary components

You may delete the imaginary components of an H1 and/or H2 filter by selecting the **Delete Imaginary components** option. This is useful for removing rounding errors that may appear after optimising a design.

5.2.3. Support for 3rd party software development frameworks



Version 5 has a completely revamped filter summary UI, and now includes built in AI to analyse the filter cascade for any potential problems. Advice on how to correct the problem is given via the toolbar, and in many cases may be rectified by clicking on the 'Fix' button.

The project wizard bundles all of the necessary SDK framework files needed to run the designed filter cascade without the need for any other dependencies or 3rd party plugins. The supporting code frameworks support deployment of the complete filter cascade, and within a few clicks, the generated project files can be used in any industry standard IDE, such as Microsoft Visual Studio for C# projects.

Each code generator is supported by a detailed tutorial, that can be accessed by clicking on the '**show me how to use this code**' link.

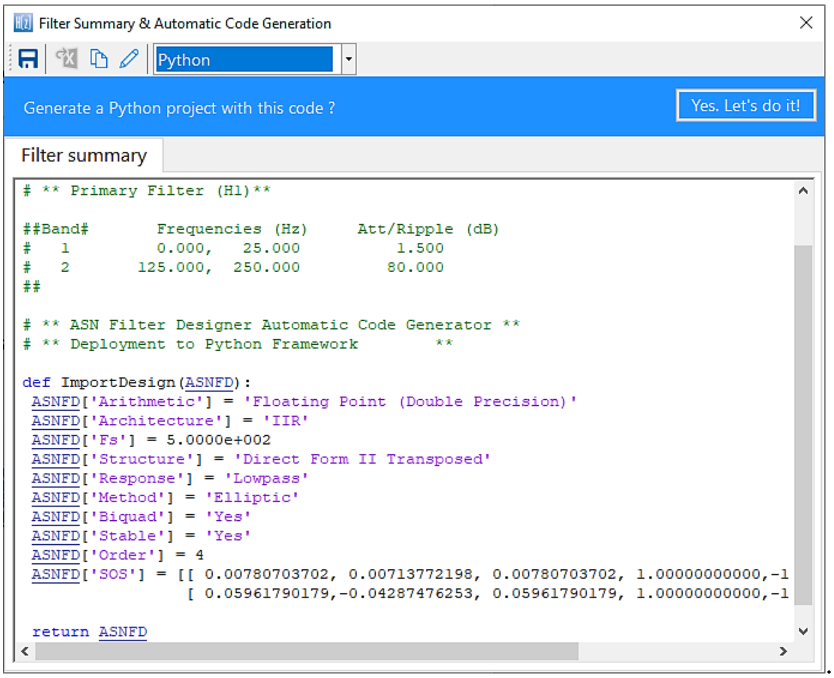
5.2.3.1. Agnostic C code and the ARM CMSIS-DSP library

For embedded developers, the supporting C SDK framework has been developed to be agnostic such that it will run on a variety of embedded systems, including all Arm Cortex-M processors, the popular ESP32 SoC, and many other platforms including Beagle Bone, Raspberry Pi and Arduino.

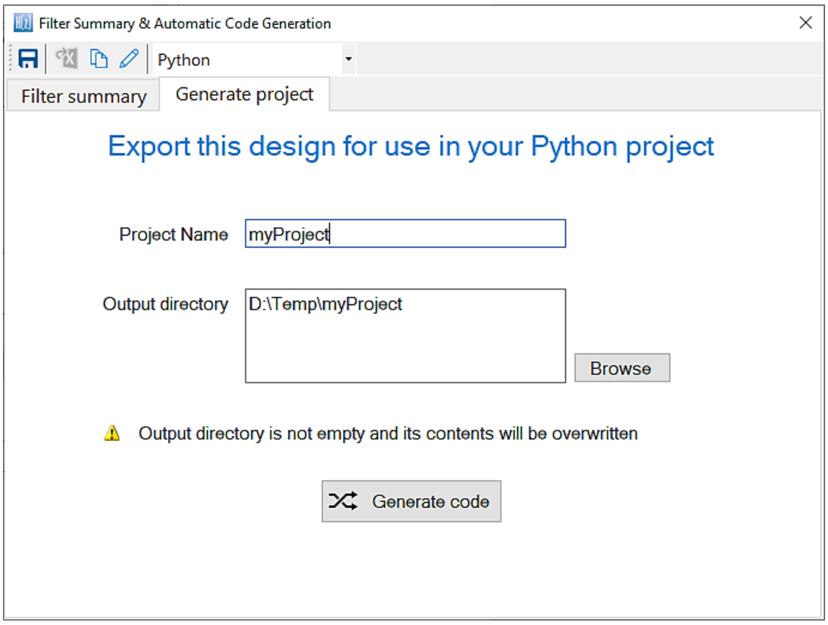
An overview video of how to design a biomedical filter and deploy its generated ANSI C code to STM32 Cube IDE is covered in the following video for a biomedical PPG application.

5.2.3.2. Project code Wizard (all code generators)

The blue toolbar analyses the design, and provides user feedback in order to proceed or correct any errors required for project code generation.

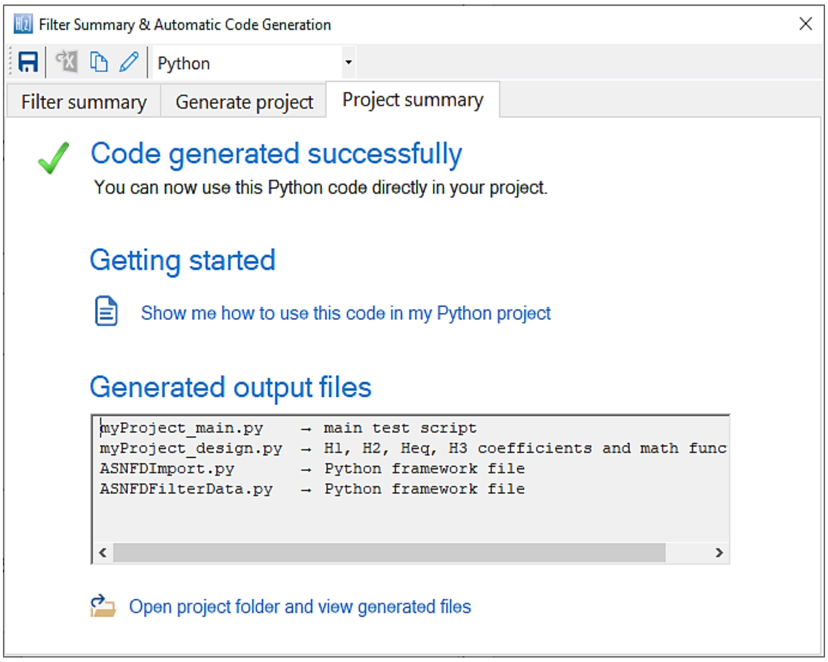


The project code generation wizard then appears, as shown below.



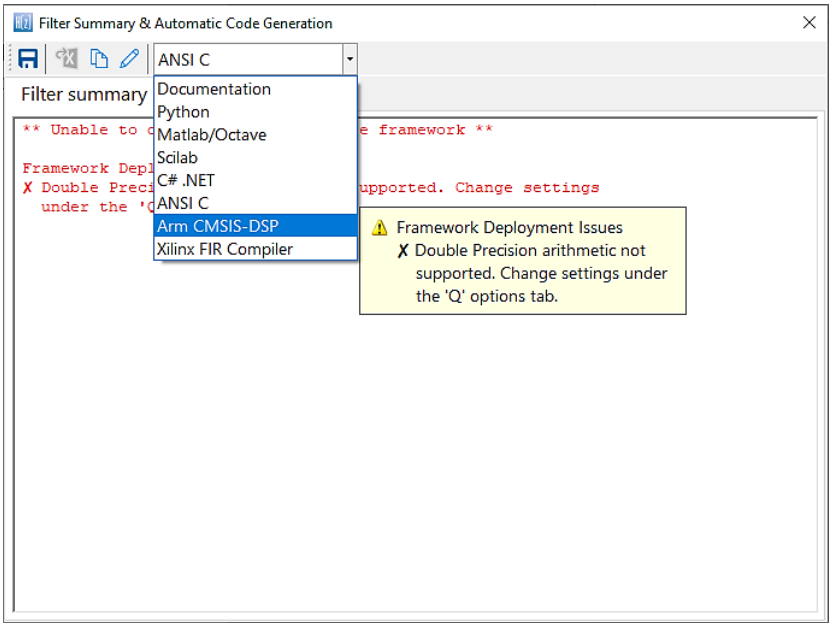
After entering a valid project name (minimum of two characters) and selecting an output directory, you may begin the code generation by clicking on the **Generator code** button.

If successful, the following tab will appear, with a list of generated output files and a link for supporting documentation on how to use the generated code in your project.

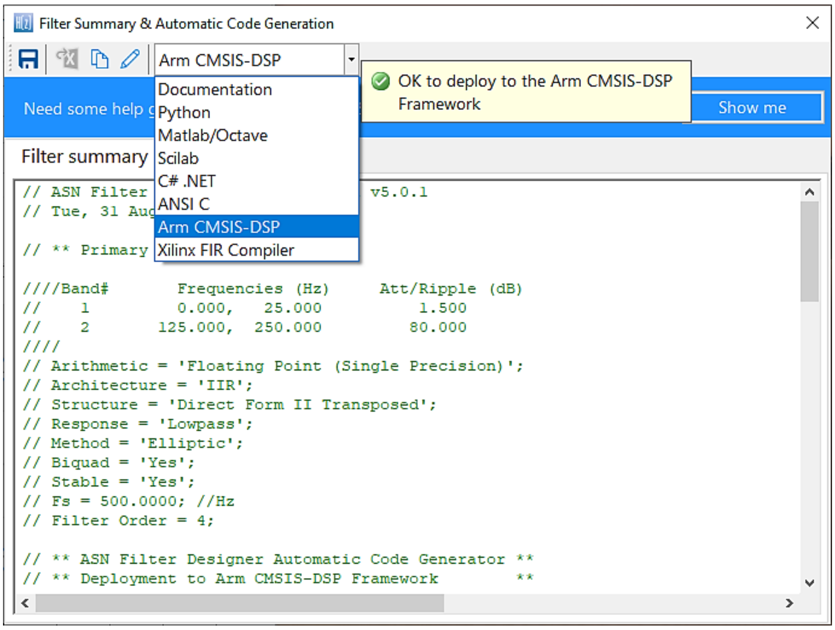


5.2.4. Automatic code generation for Arm CMSIS-DSP

ASN Filter Designer can automatically generate optimised SIMD (single instruction, multiple data) instruction set C code for ARM Cortex-M based processors via the ARM CMSIS DSP framework. The tool's inbuilt analytics automatically check specifications (such as filter structure and quantisation) in order to ensure that the generated code matches the design. The designer is then presented with a summary of 'issues to fix' if any problems are found. An example of this is shown below:



Upon solving any issues, the tool will automatically generate the code needed for your design:



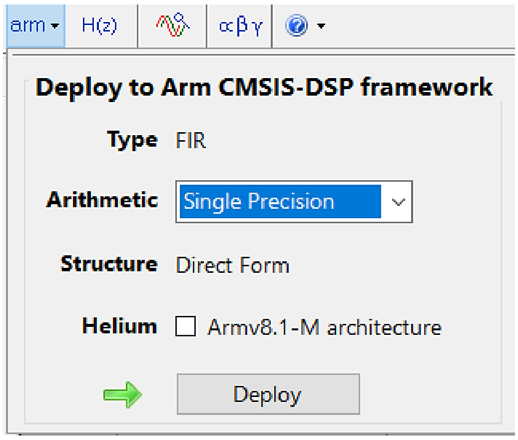
This code can now be copied and pasted into a development project and used directly. Finally, the tool produces code for the Cortex-M4 as default, please refer to the table below for **#define** definition required for other cores.

`ARM_MATH_CM0`	Cortex-M0 core.	`ARM_MATH_CM4`	Cortex-M4 core.
`ARM_MATH_CM0PLUS`	Cortex-M0+ core.	`ARM_MATH_CM7`	Cortex-M7 core.
`ARM_MATH_CM3`	Cortex-M3 core.		
`ARM_MATH_ARMV8MBL`	ARMv8M Baseline target (Cortex-M23 core).		
`ARM_MATH_ARMV8MML`	ARMv8M Mainline target (Cortex-M33 core).		

⚠ Single section and complex IIR filters are currently not supported. Please use the ANSI C framework for deploying these types of filters.

5.2.4.1. Quick deploy

For designers with a professional licence, the tool implements a shortcut wizard that automatically chooses the best settings for deploying to the Arm CMSIS-DSP framework. This requires little or no knowledge of framework settings, and is very handy for beginners.

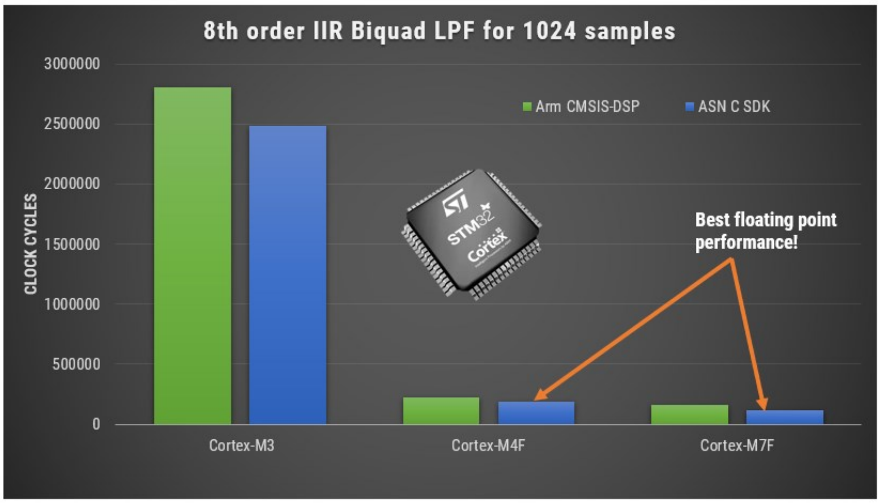


The deployment wizard also supports code generation for the Cortex-M52, Cortex-M55 and Cortex-M85 cores using the Helium Armv8.1 architecture.

H1, H2 and Heq filters will be automatically analysed and automatically converted to an H1 filter.

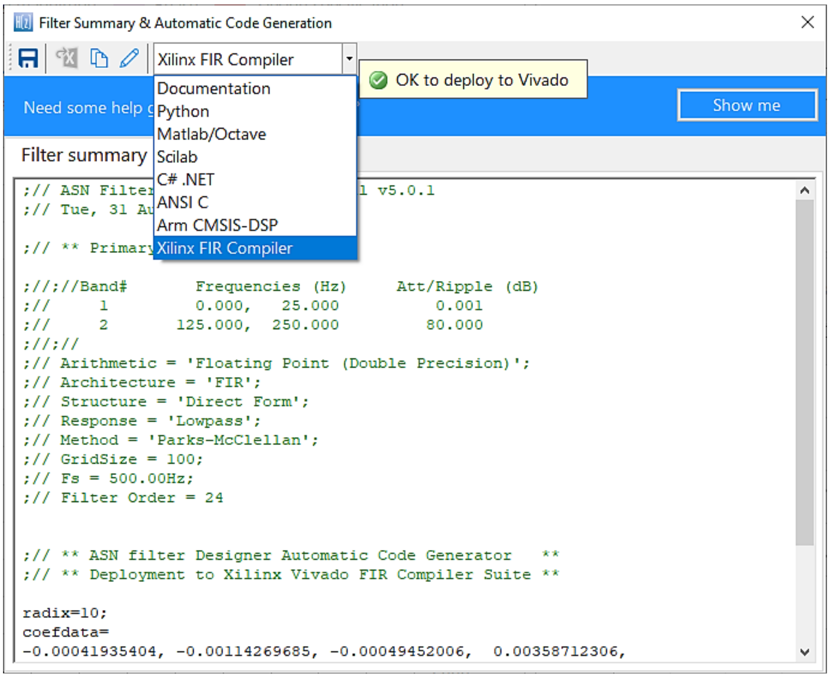
Complex IIR filters are automatically converted to real filters by deleting their imaginary components. Whereas, Complex FIR filters are supported by the framework, and implemented as two parallel filters respectively.

The tool will automatically analyse each filter in the filter cascade, including any math operators. If the design cannot be deployed with the Arm CMSIS-DSP framework, then the **ASN C SDK framework will be selected**. This Framework is only available for Single and Double precision floating point, and is slightly faster than the Arm CMSIS-DSP framework.



5.2.5. Automatic code generation for Xilinx FIR Compiler

FIR filters may be exported to the Xilinx Vivado FIR compiler suite. The data format is given below, and in the case of complex FIR filters, two sets of coefficients are generated.

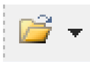


A supporting tutorial is available to get you up and running.

6. Other options: Project files and Design notes

The ASN filter designer allows licensed users to save any project design notes as part of the project file, providing a powerful documentation solution, suitable for peer review and/or project handover. The tool has been designed in order to allow non-licensed (demo) users to view the design notes and filter frequency response of a design created with either the educational or professional version, which is ideal for students and demonstrating your design to clients.

6.1. Opening project files



All versions of the software allow you to load a project file. For the demo version, this has the added advantage of allowing you to load a project file created in the either the professional or educational version for evaluation purposes.

Although the filter's frequency response may always be analysed for all versions, the signal generator (see section 4) places a limit on the maximum licensed filter order. This usually impacts users of the demo version, as a project file created in either the educational or professional version may be loaded, but it may not always be possible to use the signal generator.

6.1.1. Opening project files from Window's command line prompt interface

Project files `(*.afd)` may be loaded into the ASN filter designer via Window's command line prompt interface using the following syntax:

```
ASNFilterDesigner.exe  <filename>
```

6.1.2. Drag and drop

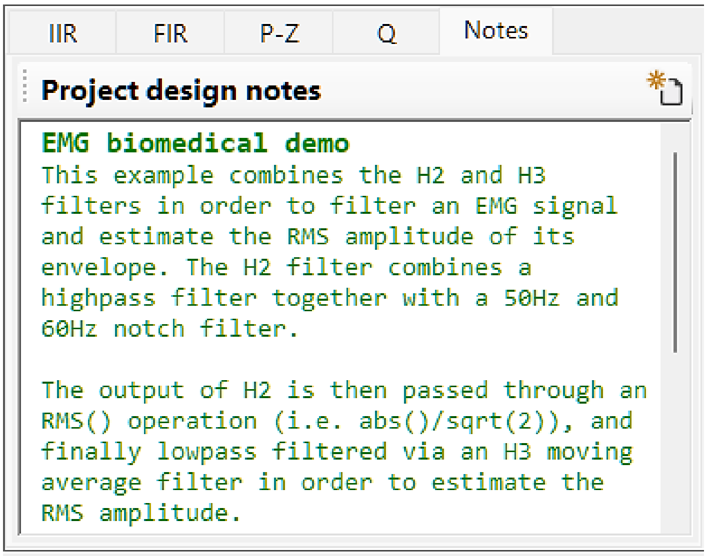
As with many Windows based applications, dragging and dropping an `afd` project file onto the main chart will automatically open it.

6.2. Saving project files

Only licensed (i.e. educational and professional) users of the tool may create project files. Clicking on the arrow will allow you to **Save As**

6.3. Project design notes

Any project design notes may be entered in the **Project design notes** textbox. The tool will automatically enhance the title (i.e., all of the text from the beginning to the first carriage return). Although formatted body text, such as bold and italics is currently not supported, but hyperlinks are supported and are a useful means of helping document and maintain a design.



The project design notes are automatically saved when saving a project file (licensed users only), and automatically updated when a project file is loaded.

Part II

Bespoke filter design and the H2 Filter

7. Introduction

The H1 (primary) filters considered in Part I are all designed via standard prototype methods, such as Butterworth, Chebyshev for IIR filters and Parks-McClellan for FIR filters. Although these design methods are adequate for many applications, they are limited in their flexibility. As an example, consider the transfer function of an IIR notch filter:

$$H(z) = \frac{1 - 2\cos w_c z^{-1} + z^{-2}}{1 - 2r\cos w_c z^{-1} + r^2 z^{-2}}$$

where, $w_c = \frac{2\pi f_c}{fs}$ controls the centre frequency, f_c of the notch, and r controls the bandwidth of the notch. Clearly, this cannot be implemented with a standard IIR prototype.

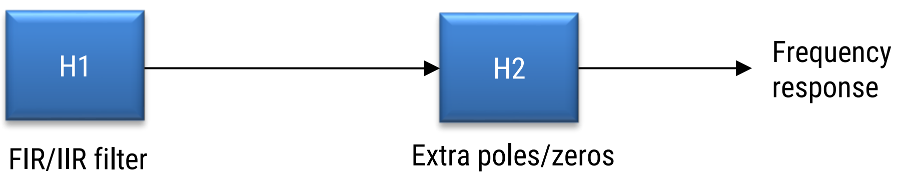
The ASN Filter designer offers designers two powerful methods for designing bespoke (specialised) filters:

- **P-Z editor**: A fully interactive pole-zero editor allowing designers to zoom, pan and graphically fine-tune designs to their exact requirements. The corresponding frequency response is updated in real-time allowing for instant evaluation of the new pole-zero positions. The zooming and panning feature is also available in the pole-zero chart, allowing designers to easily fine tune the pole-zero positions with the mouse and see the effects in real-time on the frequency response chart.

- **ASN Filter Script**: A scripting language supporting over 82 scientific commands, 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 interface 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.

7.1. The primary (H1) and secondary (H2) filter

ASN Filter Designer allows designers to add extra poles and zeros to any IIR or FIR filter via the P-Z editor. In order to facilitate this, the main FIR/IIR filter is assigned to the *primary filter, H1* and any extra pole/zeros are added to a *secondary filter block, H2*. The H2 filter block implements the filter as a **Direct Form II Transposed** single section IIR or a **Transposed Direct Form** FIR (if no poles are present). Notice that this degree of flexibility has the advantage of assigning poles to an FIR primary filter.



It should be noted that a direct form (single section) implementation may become problematic (due to numerical stability issues) for higher filter orders, especially when poles are near to the unit circle.

7.2. Design methods and customisation

All H1 (primary) filters are designed via standard prototype methods, such as Butterworth, Chebyshev for IIR filters and Parks-McClellan for FIR filters. The pole-zero positions of these 'standard filters' may be modified by the user via the P-Z editor in order to customise the design.

As will be discussed in the re-optimise design section, the P-Z editor allows designers to combine and H1 and H2 filters in order to accommodate more advanced design requirements. This is especially useful for adding extra characteristics to a standard filter with minimal effort, for example, adding an extra null (zero) to a lowpass filter. H2 filters may be specified by the P-Z editor (i.e. manually adding poles and zeros one-by-one to the z-plane), or by the symbolic math script language (ASN FilterScript). Where, the latter allows designers to specify and experiment with the H2 transfer function symbolically.

7.3. Errors in high order polynomials

The tool will for FIR filters and the filter script use the given **Num** and **Den** polynomials for computation. However, if these positions are modified via the P-Z editor, they will be handled via the roots-to-poly algorithm which will attempt to reconstruct the polynomial from the presented roots using double precision arithmetic. For lower orders this will generally result in an almost identical polynomial, but as a consequence of the errors inherent to the root finding algorithm, higher order polynomials (> 60 or so) may significantly deviate from the ideal result.

7.4. Programming the H1 filter via ASN FilterScript

Although FilterScript is primarily aimed at the H2 filter, designers may also program the H1 filter via the **H1Num**, **H1Den** and **H1Gain** commands. This provides a great deal of flexibility, as two independent filters can be customised and cascaded.

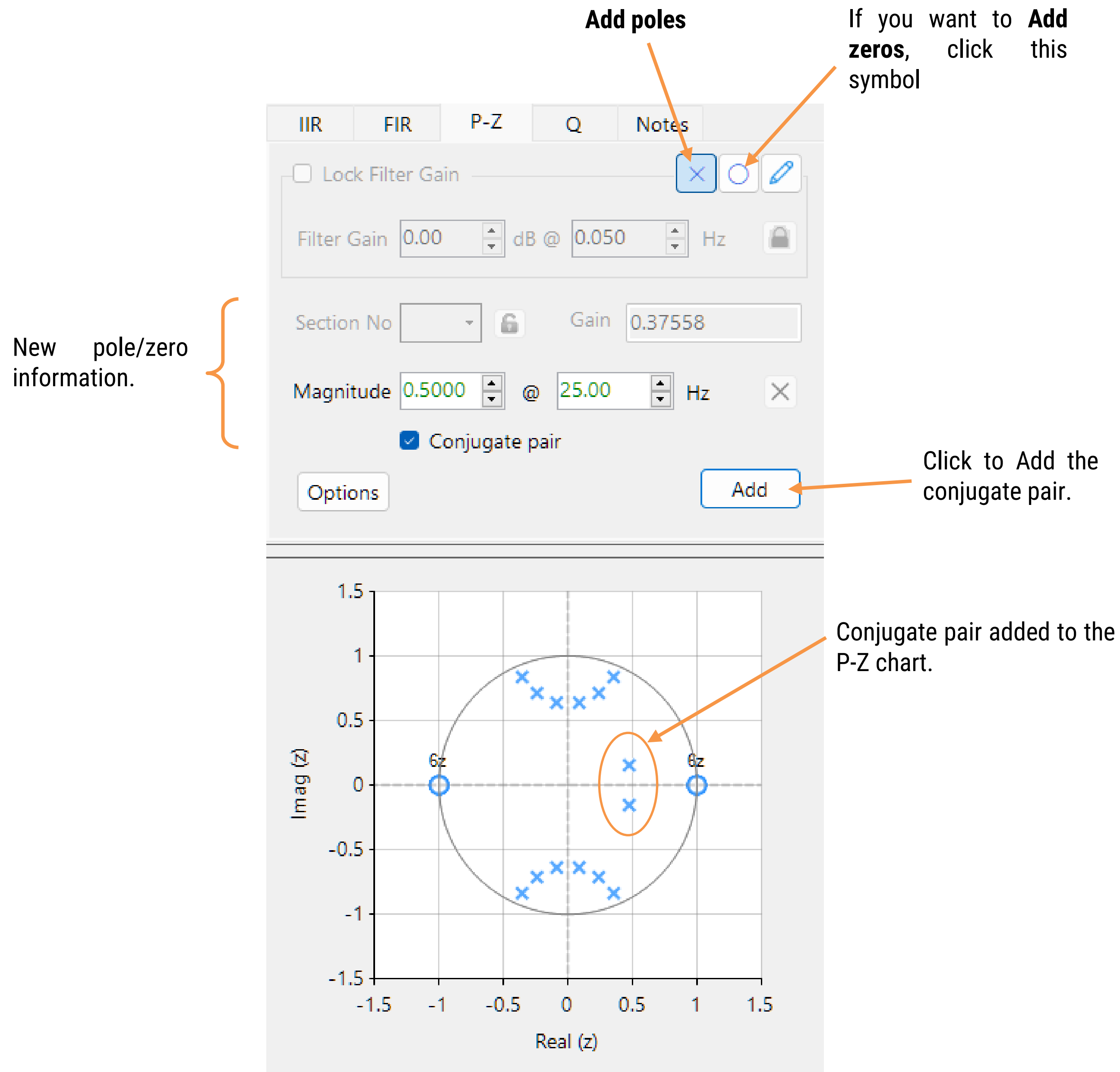
For more details, please see the ASN FilterScript reference guide for a complete explanation of the scripting language interface, including detailed practical examples.

8. P-Z editor

The P-Z editor was introduced in section 3 for editing the properties of an H1 filter. However, the editor also allows designers to add poles and zeros to a design that are implemented in the H2 filter.

Example

The following example illustrates the ease at which a conjugate pole pair can be added to the H2 filter:



Use **Edit mode** to edit the properties of the new conjugate pole pair.

8.1. Section number and section lock

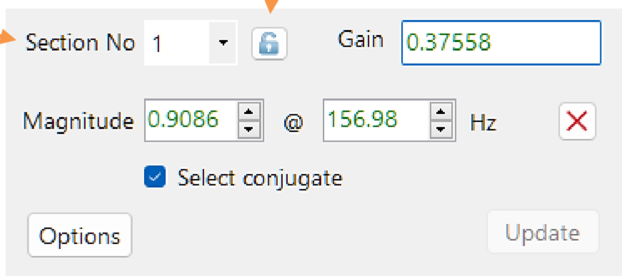
Section number

This allows you to highlight the pole and zeros of a specific section in the H1 (the primary filter) or the pole-zeros of the H2 (secondary) filter:

- For FIR filters or single section IIR filters: the section number will always be equal to **1**.
- For biquad IIR filters: this will be a list of all the biquad sections in the filter cascade.
- The secondary filter (H2): is represented as **H2**.

Section lock

Clicking on the section lock, allows you to focus on a specific section by highlighting all of the poles-zeros of the selected section number and minimising the rest.

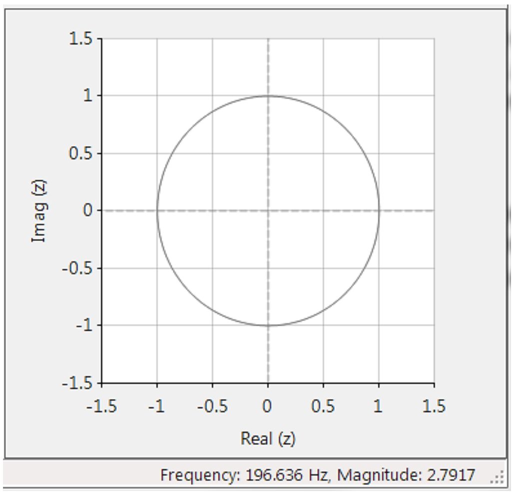


8.2. Options

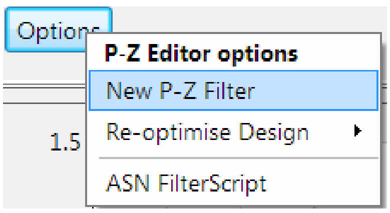
The options menu extends on the functionality of a simple pole-zero editor by allowing designers to design and experiment with any combination of poles and/or zeros of their choice. There are three options as discussed below:

8.2.1. New P-Z filter

The new pole-zero filter option deletes all poles and zeros from the design, and in essence is a blank sheet.



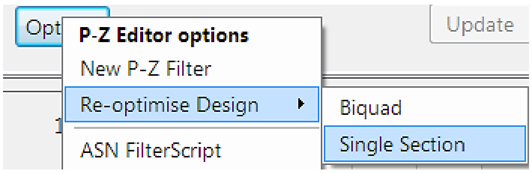
You may add pole and zeros to the design as required via the **Add pole** and **Add zero** options, as discussed at the beginning of this section. Where, all new poles and zeros are added to the H2 filter only.



The **New P-Z filter** option is especially useful for classroom examples, whereby the effects of moving a single pole or zero around the z-plane and the resulting frequency response can be seen in real-time. In essence, students can graphically see the effects of a pole/zero on the overall frequency response.

8.2.2. Re-optimise design

The re-optimise design option allows for the analysis and re-optimisation of all H1, H2 and Heq (to be discussed in section 9) poles and zeros into an H1 filter. This is especially useful for IIR biquads, as any extra poles/zeros that may have been added to H2 will be analysed and allocated to the most suitable biquad. Notice that the sub-menu allows you to choose between a single section or biquad.



In the event that you have added poles to an FIR filter, this option allows you to convert your FIR/IIR design into an H1 IIR filter.

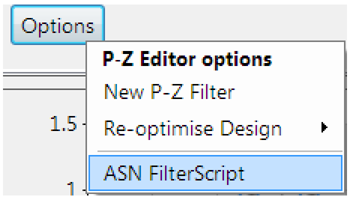
The IIR biquad section optimisation algorithm groups conjugate pole pairs with their closest conjugate zero pairs - where the conjugate pair that is closest to the unit circle is placed at the end of the filter cascade. The optimisation method also attempts to group any non-conjugate poles/zeros to any remaining conjugate pole/zero pairs.

⚠ Care should be taken when optimising a design with ASN FilterScript, as the optimisation algorithm decomposes H2's **Num** and **Den** polynomials into their poles and zeros and then combines them with H1's poles and zeros. This optimisation may lead to a slightly different frequency response than the original design for high-order polynomials due to errors in the root finding algorithm and different pole-zero pairing combinations.

ℹ It is advised in the case of high-order polynomials to first fine-tune the pole-zero positions in the P-Z editor before applying re-optimisation.

8.2.3. ASN FilterScript

The third and final option provides designers with a powerful symbolic math scripting language IDE αβγ



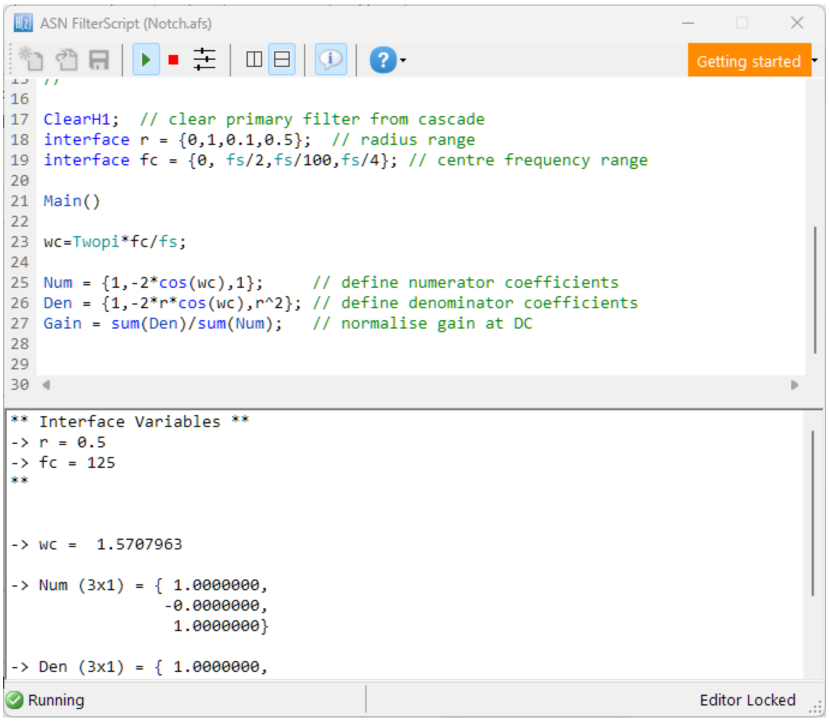
The scripting language supports over 82 scientific commands and allows you to implement complex symbolic mathematical expressions for your filtering applications. The scripting language offers the unique and powerful ability to modify parameters on the fly with the so-called interface variables, allowing for real-time updates of the resulting frequency response.

The H2 filter implements the **Num** and **Den** polynomials as defined in the filter script, rather than a re-construction of the roots presented in the P-Z chart. This is particularly useful for high order FIR filters, as no errors are introduced from the root finding algorithm.

However, in the event that any modifications are made to the pole-zero positions via the P-Z editor, the tool will automatically re-construct H1 and H2's polynomials by calling the roots-to-poly function.

Example

Revisiting the transfer function presented at the beginning of Part II, we see that almost any symbolic mathematical transfer function can be easily implemented in the FilterScript language, as shown below.



$$H(z) = \frac{1 - 2\cos w_c z^{-1} + z^{-2}}{1 - 2r\cos w_c z^{-1} + r^2 z^{-2}}$$

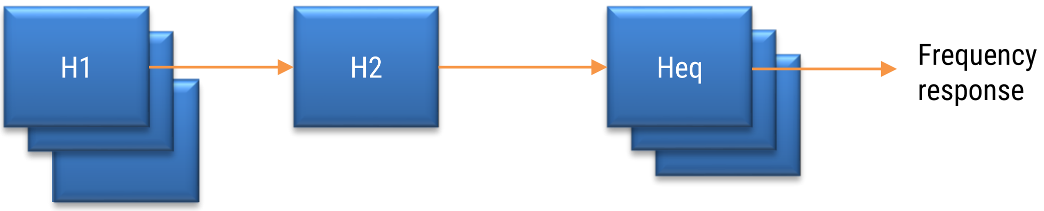
where, $w_c = \frac{2\pi f_c}{fs}$ controls the centre frequency, f_c of the notch, and r controls the bandwidth of the notch.

Setting $r = 0.5$ and $fc = 125$

Please see the ASN FilterScript reference guide for a complete explanation of the scripting language interface, including detailed practical examples.

9. Phase equalisation via the Heq filter cascade

An all-pass equalisation filter cascade (Heq) is available for equalising the phase response of the H1 filter cascade and H2 filter respectively.

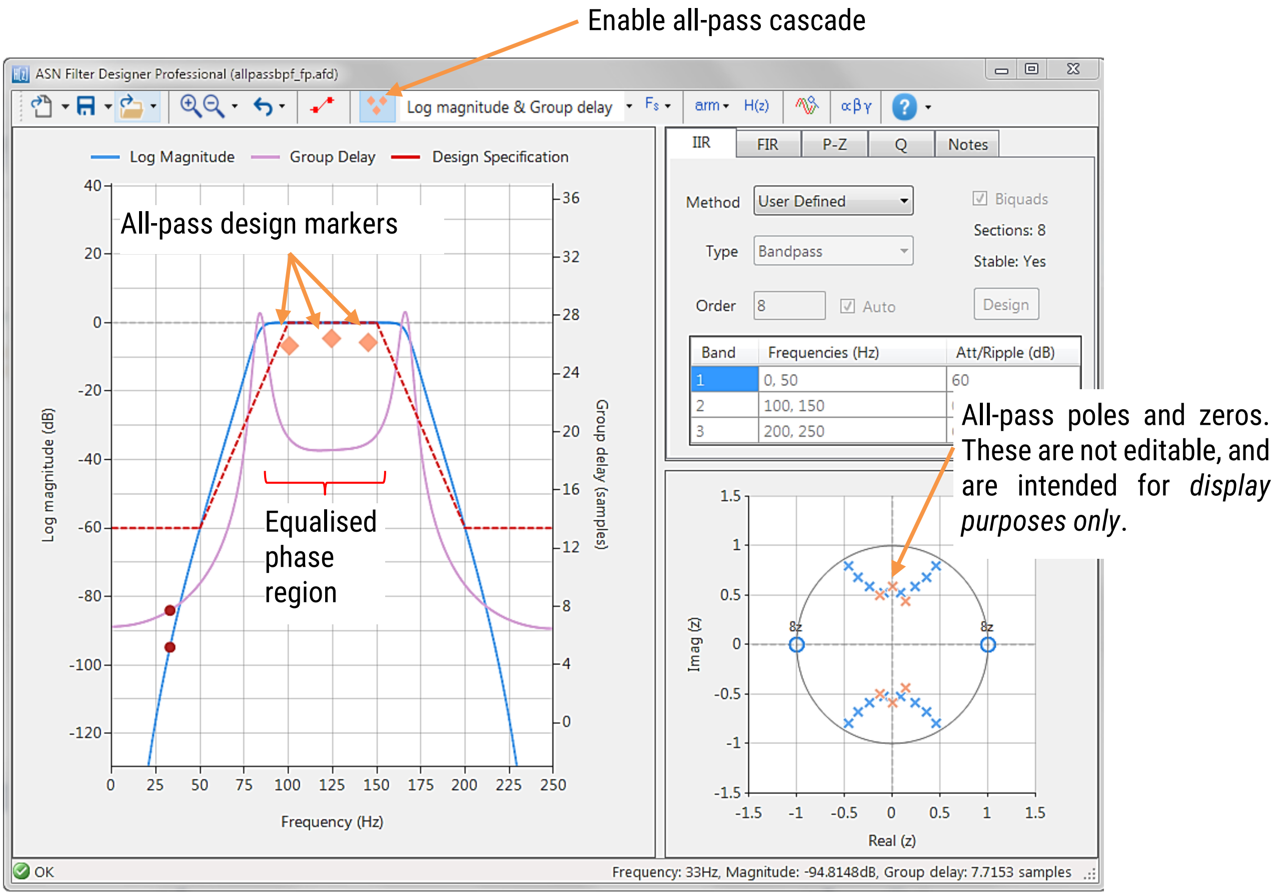


The cascade is comprised of up to ten second order (biquad) IIR all-pass filters, as defined by:

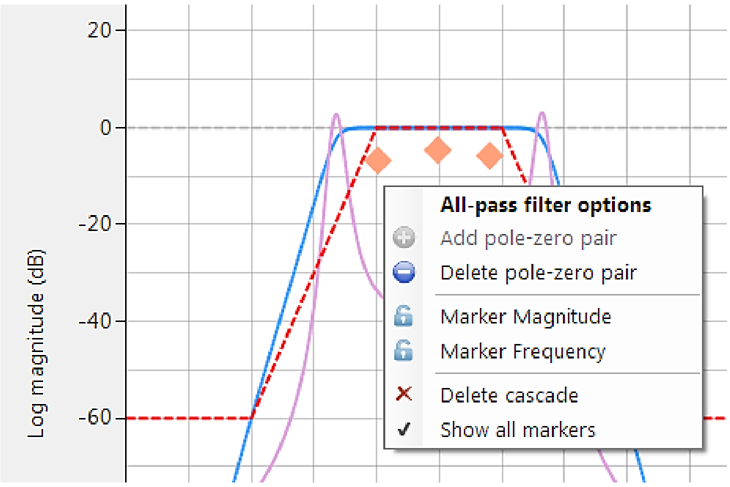
$$H(z) = \frac{r^2 - 2r\cos\left(\frac{2\pi fc}{fs}\right)z^{-1} + z^{-2}}{1 - 2r\cos\left(\frac{2\pi fc}{fs}\right)z^{-1} + r^2 z^{-2}}$$

Analysing $H(z)$, notice how the numerator and denominator coefficients are arranged as a mirror image pair of one another. This mirror image property has a highly desirable characteristic, namely the modification of phase without affecting the magnitude spectrum, hence its usefulness for phase equalisation.

As with the H1 filter design paradigm, the positions of the conjugate poles may be placed and moved interactively with the mouse. The tool will automatically compute the associated zeros and update the frequency response.



9.1. Menu options



Right clicking on the frequency response chart or on an existing all-pass design marker displays an options menu, as shown on the left. You may add up to 10 biquads (professional version only) and as with the H1 design markers, restrict the movement of the design markers.

Disabling the filter cascade *does not delete the filter cascade*, but instead allows you to see the effects of the all-pass cascade on your filter.

The Heq filter is *always implemented* using a Direct Form II Transposed (IIR) in either double or single precision arithmetic. Use the Re-optimise P-Z options to convert an equalised filter cascade into an H1 filter for deployment.

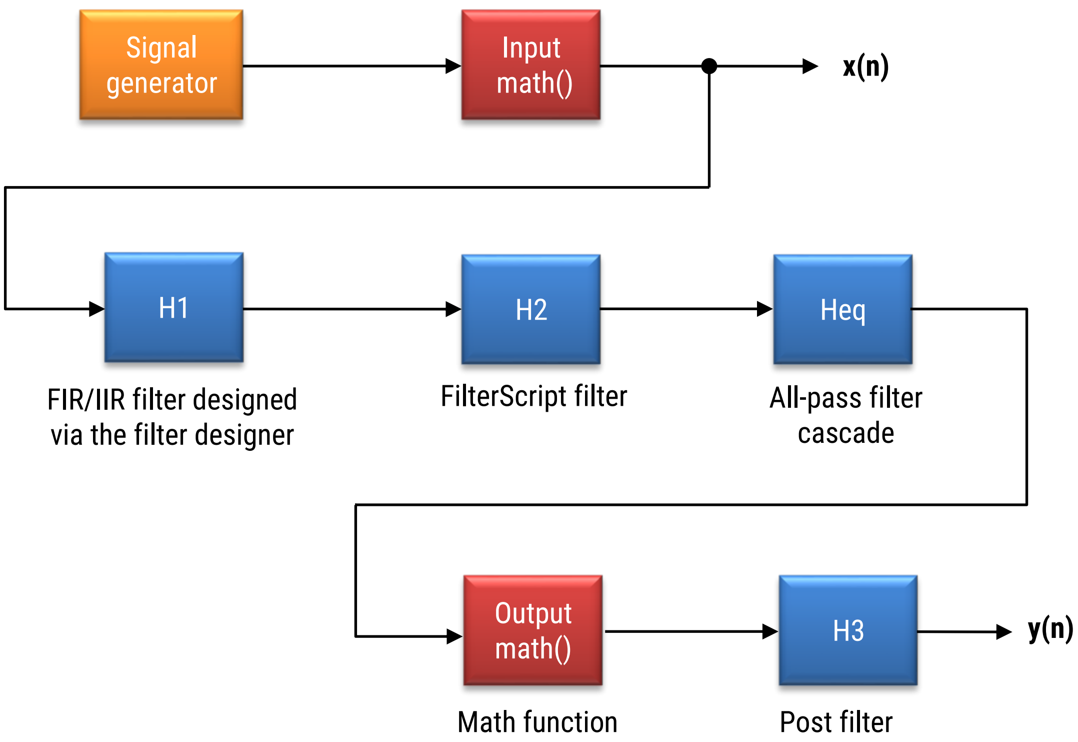
10. H1, H2, Heq, H3 filters and the signal analyser

The signal analyser GUI is comprised of a time/frequency domain analyser and a signal generator. The GUI allows designers to explore the time and frequency characteristics of both H1 and H2 type filters for various types of quantisation and inputs. However, when evaluating a fixed point design, only H1 filters may be implemented.

An extra post filter (H3) is also available for post filtering operations, as discussed below.

10.1. H3 post filtering

The signal analyser implements an extra post filter, H3. Unlike the H1 and H2 filters, the H3 filter is *always lowpass*[1] and is preceded by an optional mathematical function operation (i.e. **Abs, Angle, Ln, RMS, Sqr or Sqrt** and **TKEO**). The complete filtering chain is shown below together with the signal generator and the input/output math function blocks.



10.1.1. The TKEO algorithm

For biomedical and predictive maintenance applications, the TKEO (Teager-Kaiser energy operator) function is available,

$$y(n) = x^2(n-1) - x(n)(x-2)$$

This function removes baselines, increases signals of interest and reduces noise all in one. This method is a good alternative to using the **RMS** or **Abs** functions.

[1] *The median filter is actually a non-linear noise reduction technique.*

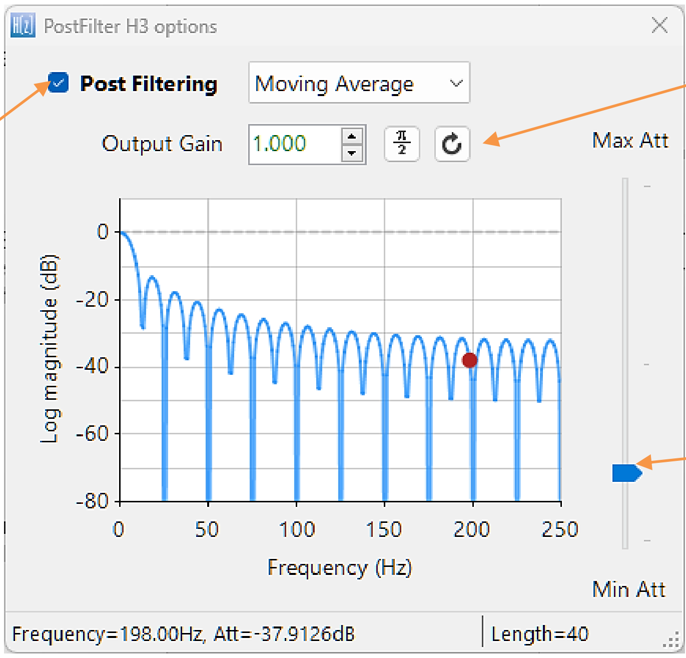
10.1.2. Types of H3 filters

The following four filters are supported:

Type	Transfer function	Gain at DC	Order
IIR	$H_3(z) = \frac{1+2z^{-1}+z^{-2}}{1+2\alpha z^{-1}+\alpha^2 z^{-2}}$	$\frac{1+2\alpha+\alpha^2}{4}$	2
FIR	$H_3(z) = 1 + z^{-1} + z^{-2} \ldots + z^{-M}$	$\frac{1}{(M+1)}$	1-300
Feed through	$H_3(z) = 1$	1	-
Median	*data window*	-	3-195

Enable/disable post filtering.

The post filter is disabled by default.



The output gain (range: `0.1-100`) proceeds the H3 filter, and is useful for fine tuning the output amplitude. The $\frac{\pi}{2}$ gain option is useful for Fullwave rectification applications.

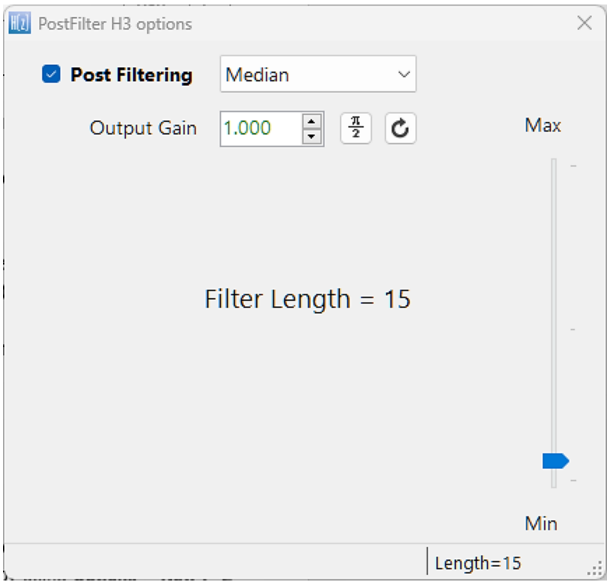
Adjust the slider to change alpha (IIR), or the filter length (FIR/Median)

NB. Unlike the main chart, no zooming functionality is provided and panning is currently limited to the Y-Axis.

⚠ As the H3 filter does not form part of the main filter designer, it is *always implemented* using a Direct Form II Transposed (IIR) or Direct Form Transposed (FIR) structure with double precision arithmetic.

10.1.2.1. The median filter

Median filters are a class of non-linear noise reduction filters, which are very good at removing spikes and retaining the sharp edges of signals. The Median filter implemented with the tool does not use a filter structure as such, and just simply computes the median of the data within its buffer (window) using double precision arithmetic.

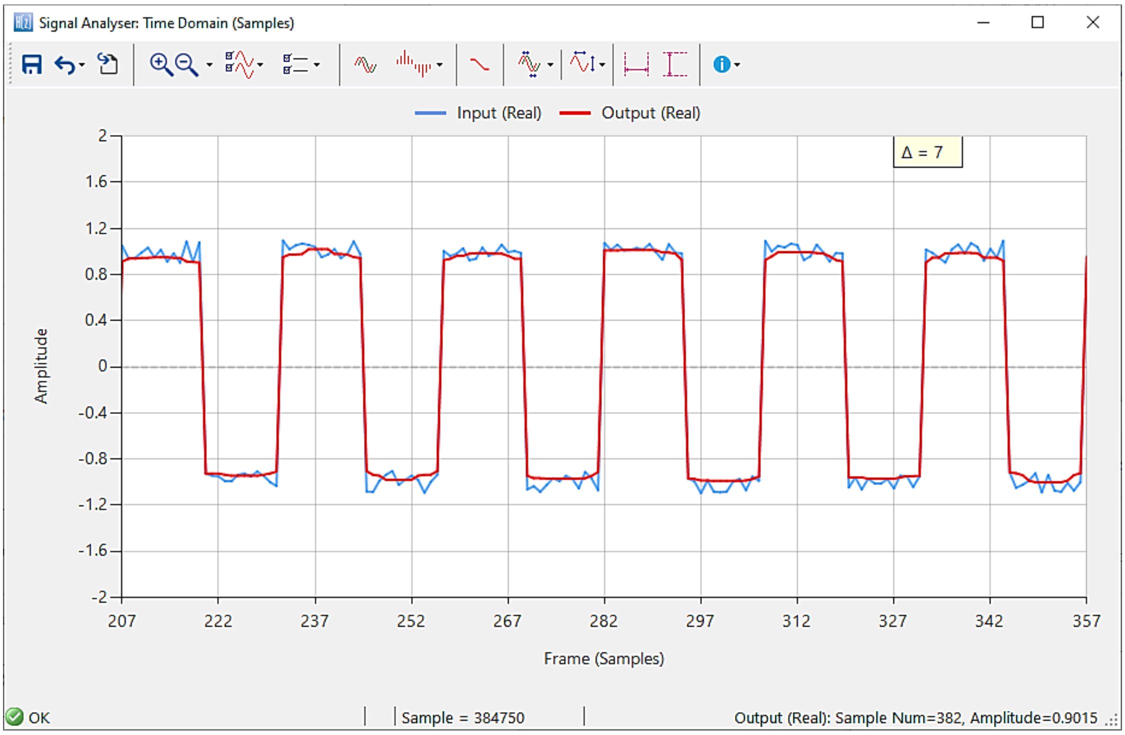


You may use the delay waveform shifting function (as discussed in section 4.3.1.2) by setting the delay to `(Filter Length – 1)/2` samples. Depending on the filter length, this allows designers to align the input and filtered output data streams respectively.

If you just want to implement a median filter, delete the H1 and H2 filters by using **Options → New P-Z filter**.

Time aligned median filtering Example

The following example demonstrates filtering perform with a median filter of **Length** `15` on a 20Hz square wave sampled at 500Hz. The **Input Delay** has been set to `7`, which aligns the datasets respectively.



Document Revision Status

Rev.	Description	Date
1	Document released for version 5.	01/09/2021
2	Document updated for version 5.1.0	27/10/2021
3	Document updated for version 5.1.1	01/02/2022
4	Document updated for version 5.1.2	27/04/2022
5	Document updated for version 5.2.1	07/11/2022
6	Document updated for version 5.2.2	09/03/2023
7	Document updated for version 5.3.1	03/08/2023
8	Document updated for version 5.3.2	26/01/2024
9	Document updated for version 5.3.3	06/02/2024
10	Document updated for version 5.3.4	23/04/2024