Digital Filter Design VIs and Functions

June 2008, 371325C-01

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Digital Filter Design VIs and functions to create digital filters, analyze filter characteristics, process signals, convert floating-point filters to fixed-point filters, simulate fixed-point filtering processes, and generate code to implement filters on embedded platforms.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Subpalette	Description
<u>Conversion</u> <u>VIs</u>	Use the Conversion VIs to retrieve and convert filter structures and to cascade filters.
<u>Filter</u> Analysis <u>VIs</u>	Use the Filter Analysis VIs to analyze the characteristics of filters.
<u>Filter</u> Design VIs	Use the Filter Design VIs to create finite impulse response (FIR), infinite impulse response (IIR), and other types of filters.
<u>Fixed-Point</u> <u>Tools VIs</u>	Use the Fixed-Point Tools VIs to quantize coefficients, model the behavior of fixed-point filters, simulate filtering processes, generate statistics reports, and generate fixed- point target code.
<u>Multirate</u> <u>Filter</u> <u>Analysis</u> <u>VIs</u>	Use the Multirate Filter Analysis VIs to analyze characteristics of multirate filters.
<u>Multirate</u> <u>Filter</u> Design VIs	Use the Multirate Filter Design VIs to create multirate filters.
<u>Multirate</u> Fixed-Point Tools VIs	Use the Multirate Fixed-Point Tools VIs to quantize filter coefficients, model the behavior of fixed-point multirate filters, simulate multirate filtering processes, and generate

	fixed-point target code.
<u>Multirate</u> Processing VIs	Use the Multirate Processing VIs to filter signals with multirate digital filters.
<u>Multirate</u> <u>Utilities VIs</u>	Use the Multirate Utilities VIs to retrieve the multirate filter coefficients and parameters and to create multirate filters from the filter coefficients.
Processing VIs	Use the Processing VIs to filter signals with digital filters.
<u>Utilities VIs</u>	Use the Utilities VIs to retrieve the filter coefficients and and to create filters from the coefficients.

 $\ensuremath{\textcircled{\sc 0}}$ 2005–2008 National Instruments Corporation. All rights reserved.

Conversion VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Conversion VIs to retrieve and convert filter structures and to cascade filters.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
DFD Cascade Filters	Creates a new filter by cascading two or more filters.
DFD Convert Structure	Converts the <u>structure of a filter</u> to a new structure that you specify.
DFD Get Filter Structure	Retrieves the structure of a specified filter.
DFD Scale Filter	Scales the coefficients of a filter without changing the characteristics of the filter.

DFD Cascade Filters VI

Owning Palette: <u>Conversion VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a new filter by cascading two or more filters.

Example

filter in 1 🔤 🧟 🛠	er out
filter in 2	or out
error in (no error)	

■ Place on the block diagram ■ Find on the **Functions** palette

- **filter in 1** specifies the first filter you want to cascade.
- **filter in 2** specifies the second filter you want to cascade.
- **filters in** specifies one or more filters that you want to cascade. Use this input instead of the **filter in 1** and **filter in 2** inputs to cascade more than two filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **filter out** returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Cascade Lowpass and Highpass to Bandpass Filter VI in the labview\examples\Digital Filter Design\Getting Started\Design Filters directory for an example of using the DFD Cascade Filters VI.

■ Open example ■ Browse related examples

DFD Convert Structure VI

Owning Palette: <u>Conversion VIs</u>

Installed With: Digital Filter Design Toolkit

Converts the structure of a filter to a new structure that you specify.

Examples

filter in 🔤 📖 🛪	filter out
error in (no error)	error out

■ Place on the block diagram ■ Find on the **Functions** palette

- **filter in** specifies the input filter.
- **target structure** specifies the new structure you want to use to design the filter. You can select one of 23 structure types for FIR and IIR filters.

1	FIR Direct Form (default)
2	FIR Direct Form Transposed
3	FIR Symmetric
4	FIR Antisymmetric
5	IIR Direct Form I
6	IIR Direct Form I Transposed
7	IIR Direct Form II
8	IIR Direct Form II Transposed
9	IIR Cascaded Second-Order Sections Form I
10	IIR Cascaded Second-Order Sections Form I Transposed
11	IIR Cascaded Second-Order Sections Form II
12	IIR Cascaded Second-Order Sections Form II Transposed
13	Lattice Allpass (basic sections)
14	Lattice Allpass (one multiplier sections)
15	Lattice Allpass (normalized sections)
16	Lattice AR (basic sections)

- 17 Lattice AR (one multiplier sections)
 18 Lattice AR (normalized sections)
 19 Lattice MA (minimum phase)
 20 Lattice MA (maximum phase)
 21 Lattice ARMA (basic sections)
 22 Lattice ARMA (one multiplier sections)
 23 Lattice ARMA (normalized sections)
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the filter with the new structure.

error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.



status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD Convert Structure VI:

• Structure Selection and Quantization VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• Change Structure of Filter VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

■ Open example ■ Browse related examples

DFD Get Filter Structure VI

Owning Palette: <u>Conversion VIs</u>

Installed With: Digital Filter Design Toolkit

Retrieves the structure of a specified filter.

<u>Example</u>



■ Place on the block diagram ■ Find on the **Functions** palette

- **filter in** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- filter out returns the filter in unchanged.
- **structure** returns the structure of the filter.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Change Structure of Filter VI in the labview\examples\Digital Filter Design\Getting Started\Apply Filters directory for an example of using the DFD Get Filter Structure VI.

■ Open example ■ Browse related examples

DFD Scale Filter VI

Owning Palette: <u>Conversion VIs</u>

Installed With: Digital Filter Design Toolkit

Scales the coefficients of a filter without changing the characteristics of the filter.

Details Example



■ Place on the block diagram ■ Find on the **Functions** palette

scale type specifies the options you use to scale the filter coefficients. Refer to the <u>Details</u> section of this topic for more information about each scale type.

0 No Norm

1 **Time Domain 1-Norm** (default)

2 **Time Domain 2-Norm**

3 Time Domain Inf-Norm

- **filter in** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the scaled filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Scale Filter Details

This VI calculates the norm values for all impulse responses of the <u>summation outputs</u> in the filter. This VI then chooses an appropriate norm value to use as the reference for scaling the filter coefficients. You can use the following four norm calculation types:

• **Time Domain 1-Norm**—Specifies that for the impulse response of each summation output, this VI calculates the corresponding norm value by using the following equation:

 $\sum |h|$

where h_i is the impulse response.

This scale type ensures that the dynamic range of the output values is within an appropriate range. However, this scale type also reduces the signal-to-rounding-noise ratio.

• **Time Domain 2-Norm**—Specifies that for the impulse response of each summation output, this VI calculates the corresponding norm value by using the following equation:



This scale type returns a higher signal-to-rounding-noise ratio than the **Time Domain 1-Norm** type. However, this scale type cannot guarantee an appropriate dynamic range of the output values.

• **Time Domain Inf-Norm**—Specifies that for the impulse response of each summation output, this VI calculates the corresponding norm value by using the following equation:

 $\max|h_i|$

This scale type cannot guarantee an appropriate dynamic range of the output values.

• No Norm—Does not normalize the impulse response.

The four norm calculation types in the list above have decreasing requirements, which correspondingly result in a decreased overflowhandling capability for fixed-point operations such as quantization. Choose the appropriate scale type according to the specific requirements and applications.

Example

Refer to the Scale Filter before Targeting to FXP VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate directory for an example of using the DFD Scale Filter VI.

■ Open example ■ Browse related examples

Filter Analysis VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Filter Analysis VIs to analyze the characteristics of filters.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
<u>DFD Get Freq</u> <u>Response</u>	Calculates the frequency responses, including the magnitude and phase responses, of a filter at specified frequency points.
<u>DFD Plot Freq</u> <u>Response</u>	Plots the frequency responses, including the magnitude and phase responses, of a filter.
<u>DFD Plot Group</u> Delay	Plots the <mark>group delay response</mark> of a filter.
DFD Plot Impulse Response	Plots the impulse response of a filter.
<u>DFD Plot</u> <u>Narrowband</u> Freq Response	Plots the frequency responses, including the magnitude and phase responses, of a <u>narrowband</u> <u>filter</u> .
DFD Plot Phase Delay	Plots the phase delay response of a filter.
DFD Plot Pole- Zero	Plots the poles and zeroes of a filter in the z-plane.
DFD Plot Step Response	Plots the step response of a filter.
<u>Filter Analysis</u>	Analyzes the specified characteristics of a filter.

Filter Analysis Express VI

Owning Palette: Filter Analysis VIs Installed With: Digital Filter Design Toolkit Analyzes the specified characteristics of a filter.

Dialog Box Options

Block Diagram Inputs

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

Parameter	Description			
Show impulse response	Plots the impulse response if you place a checkmark in the checkbox.			
Show step response	Plots the step response if you place a checkmark in the checkbox.			
Show pole-zero plot	Plots the <u>poles and zeroes</u> if you place a checkmark in the checkbox.			
Show magnitude response	Plots the magnitude response if you place a checkmark in the checkbox.			
Show phase response	Plots the phase delay response if you place a checkmark in the checkbox.			
Show group delay	Plots the group delay response if you place a checkmark in the checkbox.			
Magnitude	 Contains the following options: Linear—Displays the magnitude response using a linear scale. dB—Displays the magnitude response in decibels. 			
Phase	 Contains the following options: Unwrap phase—Specifies whether this Express VI wraps or unwraps the phase. If you place a checkmark in the checkbox, this Express VI unwraps the phase and does not restrict the phase to [0, 2π]. Unit in degree—Specifies whether to display the phase in degrees or radians. If you place a checkmark in the checkbox, this Express VI displays the phase in degrees. 			

Block Diagram Inputs

Parameter	Description
filter in	Specifies the input filter.
error in (no error)	Describes error conditions that occur before this VI or function runs.

Block Diagram Outputs

Parameter	Description		
magnitude	Returns the magnitude response plot.		
phase	Returns the phase response plot.		
group delay	Returns the group delay response plot.		
Z Plane	Returns the pole-zero response plot.		
impulse	Returns the impulse response plot.		
step	Returns the step response plot.		
error out	Contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces.		

DFD Get Freq Response VI

Owning Palette: Filter Analysis VIs

Installed With: Digital Filter Design Toolkit

Calculates the frequency responses, including the magnitude and phase responses, of a filter at specified frequency points.

Examples



■ Place on the block diagram ■ Find on the **Functions** palette

output option specifies the frequency response that this VI plots.

0 **Floating-Point** (default)—If **filter in** is a floating-point filter, this VI calculates the frequency response of this filter. If **filter in** is a fixed-point filter, this VI calculates the frequency response of the reference floating-point filter.

1 **Fixed-Point**—If **filter in** is a floating-point filter, this VI returns an error. If **filter in** is a fixed-point filter, this VI calculates the frequency response of this filter.

- **freq points** specifies the frequency points, in hertz, at which you want to calculate the frequency response.
- **filter in** specifies the input filter.
- **phase view** specifies the phase response display settings.
 - **unwrap?** specifies whether this VI unwraps the phase. The default is FALSE, which specifies that the phase remains wrapped and is limited to $[0, 2\pi)$.
 - **degree?** specifies whether the phase appears in degrees or radians. The default is FALSE, which specifies that the phase appears in radians.
- **dB on?** specifies whether this VI uses decibels or a linear scale to

express the magnitude response. The default is TRUE, which specifies that this VI converts linear magnitude response to decibels.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The default is –1, which specifies that this VI uses the sampling frequency of the input filter.
- filter out returns the filter in unchanged.
- **magnitude** returns the resulting magnitude response at **freq points**.
- **phase** returns the resulting phase response at **freq points**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD Get Freq Response VI:

• Lpth Norm Complex Approximation-Compensate Channel Distortion VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

■ Open example ■ Browse related examples

• Analyze Frequency Response of Filter with Log Spaced Freq Bins VI: labview\examples\Digital Filter Design\Getting Started\Analyze Filters

■ Open example ■ Browse related examples

DFD Plot Freq Response VI

Owning Palette: Filter Analysis VIs

Installed With: Digital Filter Design Toolkit

Plots the frequency responses, including the magnitude and phase responses, of a filter.

Details Examples



■ Place on the block diagram ■ Find on the **Functions** palette

output option specifies the filter response that this VI plots.

- 0 **Auto** (default)—This VI automatically determines the responses of **filter in** to plot. If **filter in** is a floating-point filter, this VI plots the frequency response of this filter. If **filter in** is a fixed-point filter, this VI plots the frequency responses of both the fixedpoint filter and the reference floating-point filter.
- 1 **Floating-Point Only**—If **filter in** is a floating-point filter, this VI plots the frequency response of this filter. If **filter in** is a fixed-point filter, this VI plots the frequency response of the reference floating-point filter.
- 2 **Fixed-Point Only**—If **filter in** is a floating-point filter, this VI returns an empty graph. If **filter in** is a fixed-point filter, this VI plots the frequency response of this filter.
- **freq bins** specifies the number of frequency bins between 0 and **fs** that this VI plots in the frequency response. The default is –1, which specifies that this VI automatically determines the number of frequency bins.
- **filter in** specifies the input filter.
- **phase view** specifies the phase response display settings.
 - **unwrap?** specifies whether this VI unwraps the phase. The

default is FALSE, which specifies that the phase remains wrapped and is limited to $[0, 2\pi)$.

- **degree?** specifies whether the phase appears in degrees or radians. The default is FALSE, which specifies that the phase appears in radians.
- **dB on?** specifies whether this VI uses decibels or a linear scale to express the magnitude response. The default is TRUE, which specifies that this VI converts linear magnitude response to decibels.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The default is –1, which specifies that this VI uses the sampling frequency of the input filter.
- **filter out** returns the **filter in** unchanged.
- magnitude response returns the magnitude response of the filter.
- **phase response** returns the phase response of the filter.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Plot Freq Response Details

Given the transfer function of a filter H(z), the frequency response of the filter is defined by the following equation:

 $H(e^{j\omega}) = H(z)|_{z = e^{j\omega}}$

 $H(e^{j\omega})$ is a complex function that can be expressed using the following equation:

 $H(e^{j\omega}) = |H(e^{j\omega})|e^{j\theta(\omega)}$

where $|H(e^{j\omega})|$ is defined as the magnitude response, and $\theta(\omega)$ is defined as the phase response.

If *k* is an integer, $H(e^{j(\omega+2k\pi)}) = H(e^{j2k\pi}e^{j\omega}) = H(e^{j\omega})$, so $H(e^{j\omega})$ is periodic with a period 2π . $|H(e^{j\omega})|$ and $\theta(\omega)$ also are periodic with a period 2π . For one period $-\pi \le \omega \le \pi$, the magnitude response is even symmetric and the phase response is odd symmetric. The frequency response is calculated only for $0 \le \omega \le \pi$, meaning the frequency is between 0 and $f_s/2$, where f_s is the normalized sampling frequency.

Examples

Refer to the following VIs for examples of using the DFD Plot Freq Response VI:

• Frequency Analysis of a Filter Design - DFD VI: labview\examples\Digital Filter Design\AALXMPL

■ Open example ■ Browse related examples

• Analyze Frequency Response of Filter VI: labview\examples\Digital Filter Design\Getting Started\Analyze Filters

Open example Browse related examples

DFD Plot Group Delay VI

Owning Palette: Filter Analysis VIs

Installed With: Digital Filter Design Toolkit

Plots the group delay response of a filter.

Examples

filter in	.a. a.	filter out
error in (no error) fs [Hz] group delay unit	- 40 - 400 	error out

■ Place on the block diagram ■ Find on the **Functions** palette

output option specifies the group delay response that this VI plots.

0 **Auto** (default)—This VI automatically determines the responses of **filter in** to plot. If **filter in** is a floating-point filter, this VI plots the group delay response of this filter. If **filter in** is a fixed-point filter, this VI plots the group delay responses of both the fixedpoint filter and the reference floating-point filter.

- 1 **Floating-Point Only**—If **filter in** is a floating-point filter, this VI plots the group delay response of this filter. If **filter in** is a fixed-point filter, this VI plots the group delay response of the reference floating-point filter.
- 2 **Fixed-Point Only**—If **filter in** is a floating-point filter, this VI returns an empty graph. If **filter in** is a fixed-point filter, this VI plots the group delay response of this filter.
- # freq bins specifies the number of frequency bins between 0 and fs that this VI plots in the group delay response. The default is -1, which specifies that this VI automatically determines the number of frequency bins.
- filter in specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error

occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The default is –1, which specifies that this VI uses the sampling frequency of the input filter.
- **group delay unit** specifies the unit of measurement for the group delay response.

0	Samples (default)
1	Seconds

- **filter out** returns the **filter in** unchanged.
- **group delay** returns the group delay response of the filter in samples or seconds, depending on the value you specify for the **group delay unit** input.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

status is TRUE (X) if an error occurred or FALSE

(checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD Plot Group Delay VI:

Group Delay Compensation VI: labview\examples\Digital Filter
 Design\Floating-Point Filters\Conventional

■ Open example ■ Browse related examples

• Analyze Frequency Response of Filter VI: labview\examples\Digital Filter Design\Getting Started\Analyze Filters

Open example Browse related examples
DFD Plot Impulse Response VI

Owning Palette: Filter Analysis VIs

Installed With: Digital Filter Design Toolkit

Plots the impulse response of a filter.

Details Example



■ Place on the block diagram ■ Find on the **Functions** palette

output option specifies the impulse response that this VI plots.

- 0 **Auto** (default)—This VI automatically determines the responses of **filter in** to plot. If **filter in** is a floating-point filter, this VI plots the impulse response of this filter. If **filter in** is a fixed-point filter, this VI plots the impulse responses of both the fixed-point filter and the reference floating-point filter.
- 1 **Floating-Point Only**—If **filter in** is a floating-point filter, this VI plots the impulse response of this filter. If **filter in** is a fixed-point filter, this VI plots the impulse response of the reference floating-point filter.
- 2 **Fixed-Point Only**—If **filter in** is a floating-point filter, this VI returns an empty graph. If **filter in** is a fixed-point filter, this VI plots the impulse response of this filter.
- **# points** specifies the number of samples that this VI plots for the impulse response. If the value is an integer greater than zero, this VI uses the specified value. If the value is less than or equal to zero, this VI automatically determines the number of samples. The default is -1.
- **filter in** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value

to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The default is –1, which specifies that this VI uses the sampling frequency of the input filter.
- **filter out** returns the **filter in** unchanged.
- **impulse response** returns the impulse response of the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning.

DFD Plot Impulse Response Details

Impulse response is the response of the filter to the impulse signal

```
\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}
```

A filter is considered stable if the filter impulse response h(n) approaches 0 as n goes to infinity; otherwise, the filter is unstable.

Example

Refer to the Analyze Impulse and Step Response of Filter VI in the labview\examples\Digital Filter Design\Getting Started\Analyze Filters directory for an example of using the DFD Plot Impulse Response VI.

■ Open example ■ Browse related examples

DFD Plot Narrowband Freq Response VI

Owning Palette: Filter Analysis VIs

Installed With: Digital Filter Design Toolkit

Plots the frequency responses, including the magnitude and phase responses, of a <u>narrowband filter</u>.

Examples



■ Place on the block diagram ■ Find on the **Functions** palette

freq bins specifies the number of frequency bins between 0 and fs that this VI plots in the frequency response. The default is -1, which specifies that this VI automatically determines the number of frequency bins.

narrowband filter in specifies the input narrowband filter.

- **multirate filters** contains the multirate filters this VI uses to construct the narrowband filter.
- filter type contains the type of filter.

0	Lowpass	(default)
---	---------	-----------

1 Highpass

- 2 Bandpass
- 3 Bandstop
- 4 Wideband-Lowpass
- 5 Wideband-Highpass
- **phase view** specifies the phase response display settings.
 - **unwrap?** specifies whether this VI unwraps the phase. The default is FALSE, which specifies that the phase remains wrapped and is limited to $[0, 2\pi)$.
 - degree? specifies whether the phase appears in degrees or

radians. The default is FALSE, which specifies that the phase appears in radians.

- **dB on?** specifies whether this VI uses decibels or a linear scale to express the magnitude response. The default is TRUE, which specifies that this VI converts linear magnitude response to decibels.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The default is –1, which specifies that this VI uses the sampling frequency of the input filter.
- narrowband filter out returns a duplicate filter of narrowband filter in.
 - **multirate filters** contains the multirate filters this VI uses to construct the narrowband filter.
 - **filter type** contains the type of filter.

- 0 **Lowpass** (default)
- 1 Highpass
- 2 Bandpass
- 3 Bandstop
- 4 Wideband-Lowpass
- 5 Wideband-Highpass
- magnitude response returns the magnitude response of the filter.
- **phase response** returns the phase response of the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD Plot Narrowband Freq Response VI:

• Narrowband Filtering - DFD VI: labview\examples\Digital Filter Design\AALXMPL

■ Open example ■ Browse related examples

- Narrowband Filter Design and Processing VI: labview\examples\Digital Filter Design\Floating-Point Filters\Multirate
 - Open example Browse related examples

DFD Plot Phase Delay VI

Owning Palette: Filter Analysis VIs

Installed With: Digital Filter Design Toolkit

Plots the phase delay response of a filter.



■ Place on the block diagram ■ Find on the **Functions** palette

- **output option** specifies the phase delay response that this VI plots.
 - 0 **Auto** (default)—This VI automatically determines the responses of **filter in** to plot. If **filter in** is a floating-point filter, this VI plots the phase delay response of this filter. If **filter in** is a fixed-point filter, this VI plots the phase delay responses of both this filter and the reference floating-point filter.
 - 1 **Floating-Point Only**—If **filter in** is a floating-point filter, this VI plots the phase delay response of this filter. If **filter in** is a fixed-point filter, this VI plots the phase delay response of the reference floating-point filter.
 - 2 **Fixed-Point Only**—If **filter in** is a floating-point filter, this VI returns an empty graph. If **filter in** is a fixed-point filter, this VI plots the phase delay response of this filter.
- # freq bins specifies the number of frequency bins between 0 and fs that this VI plots in the phase delay response. The default is -1, which specifies that this VI automatically determines the number of frequency bins.
- **filter in** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while

this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The default is –1, which specifies that this VI uses the sampling frequency of the input filter.
- **phase delay unit** specifies the unit of measurement for the phase delay response.

0	Samples (default)
1	Seconds

- **filter out** returns the **filter in** unchanged.
- **phase delay** returns the phase delay response of the filter in samples or seconds, depending on the value you specify for the **phase delay unit** input.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Plot Pole-Zero VI

Owning Palette: Filter Analysis VIs

Installed With: Digital Filter Design Toolkit

Plots the poles and zeroes of a filter in the z-plane.

Example



 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

output option specifies the pole-zero distribution that this VI plots.

- 0 **Auto (default)**—This VI automatically determines the pole-zero distribution of **filter in** to plot. If **filter in** is a floating-point filter, this VI plots the pole-zero distribution of this filter. If **filter in** is a fixed-point filter, this VI plots the pole-zero distributions of both this filter and the reference floating-point filter.
- 1 **Floating-Point Only**—If **filter in** is a floating-point filter, this VI plots the pole-zero distribution of this filter. If **filter in** is a fixed-point filter, this VI plots the pole-zero distribution of the reference floating-point filter.
- 2 **Fixed-Point Only**—If **filter in** is a floating-point filter, this VI returns an empty graph. If **filter in** is a fixed-point filter, this VI plots the pole-zero distribution of this filter.
- **filter in** specifies the input filter.
- **upper half plane only** specifies if this VI displays zeroes and poles on only the upper-half unit circle. The default is TRUE.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u>

Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the **filter in** unchanged.
- **Z-Plane** returns the zeroes and poles of the filter in an XY graph.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Get Zero-Pole-Gain of Filter VI in the labview\examples\Digital Filter Design\Getting Started\Analyze Filters directory for an example of using the DFD Plot Pole-Zero VI.

■ Open example ■ Browse related examples

DFD Plot Step Response VI

Owning Palette: Filter Analysis VIs

Installed With: Digital Filter Design Toolkit

Plots the step response of a filter.

The step response is the response to a unit-step input signal. You can use the step response to determine when the filter reaches the steady state for the step input.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

output option specifies the filter response that this VI plots.

0 **Auto** (default)—This VI automatically determines the responses of **filter in** to plot. If **filter in** is a floating-point filter, this VI plots the step response of this filter. If **filter in** is a fixed-point filter, this VI plots the step responses of both this filter and the reference floating-point filter.

1 **Floating-Point Only**—If **filter in** is a floating-point filter, this VI plots the step response of this filter. If **filter in** is a fixed-point filter, this VI plots the step response of the reference floating-point filter.

2 **Fixed-Point Only**—If **filter in** is a floating-point filter, this VI returns an empty graph. If **filter in** is a fixed-point filter, this VI plots the step response of this filter.

- **points** specifies the number of samples that this VI plots for the step response. If the value is an integer greater than zero, this VI uses the specified value. If the value is less than or equal to zero, this VI automatically determines the number of samples. The default is -1.
- **filter in** specifies the input filter.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The default is –1, which specifies that this VI uses the sampling frequency of the input filter.
- **filter out** returns the **filter in** unchanged.
- **step response** returns the step response of the filter.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Analyze Impulse and Step Response of Filter VI in the labview\examples\Digital Filter Design\Getting Started\Analyze Filters directory for an example of using the DFD Plot Step Response VI.

■ Open example ■ Browse related examples

Filter Design VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Filter Design VIs to create finite impulse response (FIR), infinite impulse response (IIR), and other types of filters.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
<u>Classical Filter</u> <u>Design</u>	Creates a lowpass, highpass, bandpass, or bandstop filter interactively.
DFD Classical Design	Creates a lowpass, highpass, bandpass, or bandstop filter.
<u>Pole-Zero</u> Placement	Adds, deletes, and moves poles and zeroes in a filter.

Subpalette	Description
<u>Advanced</u> <u>FIR Filter</u> Design VIs	Use the Advanced FIR Filter Design VIs to estimate finite impulse response (FIR) filter order and to create FIR filters from the filter specifications.
<u>Advanced</u> <u>IIR Filter</u> Design VIs	Use the Advanced IIR Filter Design VIs to estimate infinite impulse response (IIR) filter order and to create IIR filters from the filter specifications.
<u>Special</u> Filter Design VIs	Use the Special Filter Design VIs to create notch peak, infinite impulse response (IIR) comb, maximally flat, narrowband, and other special filters.

Classical Filter Design Express VI

Owning Palette: <u>Filter Design VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a lowpass, highpass, bandpass, or bandstop filter interactively.

<u>Details</u>

Dialog Box Options

Block Diagram Inputs

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

Parameter	Description
Main	Contains the following options:
Settings	 Filter type—Specifies the type of filter that this VI creates. The valid values include Lowpass, Highpass, Bandpass, and Bandstop. The default is Lowpass. Filter Specifications—Contains the following options: Sampling frequency—Specifies the
	sampling frequency of the filter in hertz. This input must contain a value greater than zero. The default is 1k Hz.
	 Passband edge frequency 1—Specifies the first passband edge frequency of the filter in hertz. This input must contain a value greater than zero and less than the Nyquist frequency. If you set Filter type to Lowpass or Bandstop, the default is 100 Hz. If you set Filter type to Highpass or Bandpass, the default is 200 Hz.
	 Passband edge frequency 2—Specifies the second passband edge frequency of the filter in hertz. This input must contain a value greater than Passband edge frequency 1 and less than the Nyquist frequency. This input is not available for lowpass or highpass filters. If you set Filter type to Bandpass, the default is 300 Hz. If you set Filter type to Bandstop, the default is 400 Hz. Passband ripple—Specifies the passband
	ripple of the filter in units that the Magnitude in dB option determines. If you place a checkmark in the Magnitude in dB checkbox, this input must contain a value greater than zero. The default is 0.1 dB. If

you remove the checkmark from the **Magnitude in dB** checkbox, the valid range of this input is (0, 1). The default then is 0.011447.

- Stopband edge frequency 1—Specifies the first stopband edge frequency of the filter in hertz. If you set Filter type to Lowpass or Bandstop, this input must contain a value greater than Passband edge frequency 1 and less than the Nyquist frequency. The default is 200 Hz. If you set Filter type to Highpass or Bandpass, this input must contain a value greater than zero and less than Passband edge frequency 1. The default then is 100 Hz.
- Stopband edge frequency 2—Specifies the second stopband edge frequency of the filter in hertz. If you set Filter type to Bandpass, this input must contain a value greater than Passband edge frequency 2 and less than the Nyquist frequency. The default is 400 Hz. If you set Filter type to Bandstop, this input must contain a value greater than Stopband edge frequency 1 and less than Passband edge frequency 2. The default then is 300 Hz. This input is not available for lowpass or highpass filters.
- Stopband attenuation—Specifies the stopband attenuation of the filter in units that the Magnitude in dB option determines. If you place a checkmark in the Magnitude in dB checkbox, this input must contain a value greater than zero. The default is 60 dB. If you remove the checkmark from the Magnitude in dB checkbox, the valid range of this input is (0, 1). The default then is 0.001.
- **Design method**—Specifies the <u>method that this</u>

	Express VI uses to design the filter. Options include Butterworth, Chebyshev, Inverse Chebyshev, Elliptic, Kaiser Window, Dolph-Chebyshev Window, and Equi-Ripple FIR. The default is Elliptic.
Design	Contains the following options:
Feedback	 Filter order—Returns the order of the designed filter.
	 Error message—Contains details about errors that occur during filter creation.
Magnitude in dB	Specifies whether this Express VI uses decibels or a linear scale to express the magnitude response. If you place a checkmark in the checkbox, this Express VI converts a linear magnitude response to decibels. This Express VI uses decibels by default.
Passband	Specifies the color of the lines in the magnitude plot that represent the passband response and the passband frequencies. The default is blue. Click the color box next to the parameter name to select a different color.
Stopband	Specifies the color of the lines in the magnitude plot that represent the stopband attenuation and the stopband frequencies. The default is red. Click the color box next to the parameter name to select a different color.
Magnitude Response	Contains the plot of the magnitude response. You can drag the cursors in the plot to change the specifications. The color you specify in Passband represents the passband response and the passband frequencies. The color you specify in Stopband represents the stopband attenuation and the stopband frequencies. The green vertical line in the graph represents the half sampling frequency, also known as the Nyquist frequency.
Z-Plane	Contains the plot of the zeroes and poles of the filter in the z-plane.

Block Diagram Inputs

Parameter	Description
error in (no error)	Describes error conditions that occur before this VI or function runs.

Block Diagram Outputs

Parameter	Description
filter out	Returns the new filter.
error out	Contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces.

Classical Filter Design Details

As you define a filter specification, you must adhere to a set of rules to maintain valid specifications. If any violations to the rules occur, the **Error message** indicator of the **Configure Classical Filter Design** dialog box displays a message with suggestions for repositioning the cursors. The rules are as follows:

- Keep horizontal cursors in the range (0, 1) in a linear scale or (- inf, 0 dB) in a logarithmic scale.
- Keep the horizontal passband cursor above the horizontal stopband cursor.

Pole-Zero Placement Express VI

Owning Palette: Filter Design VIs Installed With: Digital Filter Design Toolkit Adds, deletes, and moves poles and zeroes in a filter. You can load a filter from an existing file.

Dialog Box Options

Block Diagram Inputs

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

Parameter	Description
Zeroes	 Specifies the properties for each zero in the filter. Contains the following options: Real—Specifies the real part value if coordinates are rectangular, or the radius value if coordinates are polar.
	 Imaginary—Specifies the imaginary part value if coordinates are rectangular, or the angle value if coordinates are polar.
	 Is real?—Makes the specified pole or zero point real if you place a checkmark in the checkbox.
	 Is linear phase pair?—Makes the specified pole or zero a linear phase pair if you place a checkmark in the checkbox.
	 On unit circle?—Places the pole or zero point on the unit circle if you place a checkmark in the checkbox.
	• Order—Specifies the order of the pole or zero. The value of Order must be an integer greater than zero. The default is 1.
Poles	 Specifies the properties for each pole in the filter. Contains the following options: Real—Specifies the real part value if coordinates are rectangular, or the radius value if coordinates are polar. Imaginary—Specifies the imaginary part value if coordinates are rectangular, or the angle value if coordinates are polar. Is real?—Makes the specified pole or zero point real if you place a checkmark in the checkbox. Is linear phase pair?—Makes the specified pole or zero a linear phase pair if you place a checkmark in the checkbox. On unit circle?—Places the pole or zero point on

	the unit circle if you place a checkmark in the checkbox.
	• Order —Specifies the order of the pole or zero. The value of Order must be an integer greater than zero. The default is 1.
Settings	Contains the following options:
	 Gain—Sets the gain of the filter manually. You can use this control only if you remove the checkmark from the Normalized gain checkbox.
	• Normalized gain—Specifies if this Express VI automatically adjusts the gain of the filter. The default contains a checkmark in the checkbox, which specifies that this Express VI adjusts the gain so the maximum response is 1.0 (0 dB). Remove the checkmark from the checkbox to adjust the gain manually with the Gain control.
	• Sampling frequency —Specifies the sampling frequency in hertz. This input must contain a value greater than zero. The default is 1 Hz.
	• Coordinates —Specifies whether this Express VI displays poles and zeroes in rectangular or polar coordinates. The default is rectangular coordinates.
Load Filter from File	Opens a file dialog you can use to select a file of a filter that loads into this Express VI.
Zero	Specifies the color of the zeroes in the Z Plane plot. The default is blue. Click the color box next to the parameter name to select a different color.
Pole	Specifies the color of the poles in the Z Plane plot. The default is red. Click the color box next to the parameter name to select a different color.
Delete Selected	Deletes the selected pole or zero from the filter.
Add Zero	Adds a zero to the filter.
	Adds a note to the filter

Z-Plane	Plots the number and location of poles and zeroes.
Magnitude Response	Plots the filter magnitude response.
Magnitude in dB	Specifies whether this Express VI uses decibels or a linear scale in the magnitude plot. If a checkmark is in the checkbox, this Express VI converts linear magnitude response to decibels.

Block Diagram Inputs

Parameter	Description
error in (no error)	Describes error conditions that occur before this VI or function runs.

Block Diagram Outputs

Parameter	Description
filter out	Returns the new filter.
error out	Contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces.

DFD Classical Design VI

Owning Palette: Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a lowpass, highpass, bandpass, or bandstop filter.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

design method specifies the method that this VI uses to design the filter.

0	Butterworth

1 Chebyshev

2 Inverse Chebyshev

3 **Elliptic** (default)

4 Kaiser Window

5 Dolph-Chebyshev Window

6 Equi-Ripple FIR

filter type specifies the <u>type of filter</u> that this VI creates.

1 Highpass

2 Bandpass

3 Bandstop

freq specs specifies the band edge frequencies of the filter.

fpass 1 specifies the first passband edge frequency in hertz.

fstop 1 specifies the first stopband edge frequency in hertz.

- **fpass 2** specifies the second passband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **fstop 2** specifies the second stopband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The default is 60.
 - **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **filter out** returns a new filter.
- **order** returns the filter order.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
Example

Refer to the Classical Filter Design VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Conventional directory for an example of using the DFD Classical Design VI.

■ Open example ■ Browse related examples

Advanced FIR Filter Design VIs

Owning Palette: Filter Design VIs

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Advanced FIR Filter Design VIs to estimate finite impulse response (FIR) filter order and to create FIR filters from the filter specifications.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
DFD Dolph- Chebyshev Design	Creates a finite impulse response (FIR) filter by using the Dolph-Chebyshev Window method.
DFD Dolph- Chebyshev Order Estimation	Estimates the order of a finite impulse response (FIR) filter that you designed by using the <u>Dolph-Chebyshev Window</u> <u>method</u> .
<u>DFD</u> <u>Kaiser</u> Design	Creates a finite impulse response (FIR) filter by using the Kaiser Window method.
DFD Kaiser Order Estimation	Estimates the order of a finite impulse response (FIR) filter that you designed by using the <u>Kaiser Window method</u> .
<u>DFD Least</u> <u>Pth Norm</u> <u>Design</u>	Designs an infinite impulse response (IIR) or finite impulse response (FIR) filter with a frequency response that matches the response you request in terms of the least p^{th} norm algorithm. You can use either the iterative reweighted least square (IRLS) method or the Newton method that this VI provides to design a filter. You must <u>manually select the</u>

	polymorphic instance you want to use.
<u>DFD</u> <u>Remez</u> Design	Creates an equi-ripple filter using the Remez exchange method.
<u>DFD</u> <u>Remez</u> <u>Order</u> Estimation	Estimates the order of a Remez equi-ripple filter.
<u>DFD</u> Windowed FIR Design	Creates a finite impulse response (FIR) filter by using the window methods.

DFD Dolph-Chebyshev Design VI

Owning Palette: Advanced FIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a finite impulse response (FIR) filter by using the <u>Dolph-Chebyshev Window method</u>.

You can use the <u>DFD Dolph-Chebyshev Order Estimation</u> VI to estimate **order**.

Details



■ Place on the block diagram ■ Find on the **Functions** palette

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- **order** specifies the filter order. The value of **order** must be greater than zero. The default is 31. **order** +1 equals the number of coefficients or filter taps. Increasing the value can narrow the transition band.
- **high cutoff freq** specifies the high cutoff frequency in hertz. The value must be greater than **low cutoff freq**. The default is 0.45. This VI uses this input only for bandpass and bandstop filter design. The cutoff frequency is the frequency with one-half magnitude response.
- **Iow cutoff freq** specifies the low cutoff frequency in hertz. The default is 0.12. The cutoff frequency is the frequency with one-half magnitude response.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **ripple ratio** specifies the side-lobe attenuation provided by the Dolph-Chebyshev Window function in decibels. The value must be greater than zero. The default is 40. Increasing the value of **ripple ratio** decreases the ripples in the passband and stopband.
- filter out returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Dolph-Chebyshev Design Details

The Dolph-Chebyshev Window is a ripple-adjustable window, as described in the following equation:

$$W_{DC}(n) = \begin{cases} \frac{1}{M+1} \left\{ \frac{1}{r} + 2\sum_{i=1}^{M/2} C_M \left[\times_0 \cos(\frac{i\pi}{M+1}) \right] \cos(\frac{2ni\pi}{M+1}) \right\} & |n| \le M/2 \\ 0 & |n| > M/2 \end{cases}$$

where *r* is the relative side-lobe attenuation of the Dolph-Chebyshev Window in decibels, and $C_M(x)$ is the M^{th} order Chebyshev polynomial.

DFD Dolph-Chebyshev Order Estimation VI

Owning Palette: Advanced FIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Estimates the order of a finite impulse response (FIR) filter that you designed by using the <u>Dolph-Chebyshev Window method</u>.



Place on the block diagram Find on the Functions palette

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass
S	Bandhaaa

- 2 Bandpass
- 3 Bandstop
- **freq specs** specifies the band edge frequencies of the filter.
 - **fpass 1** specifies the first passband edge frequency in hertz.
 - **fstop 1** specifies the first stopband edge frequency in hertz.
 - **fpass 2** specifies the second passband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
 - **fstop 2** specifies the second stopband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The

default is 60.

- **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- order option specifies filter order requirements. The default is MinEven, which yields the minimum even order for the filter to meet the specifications you set.



- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized

sampling frequency.

- estimated order returns the minimum order value that the filter requires to meet the specifications you set.
- **high cutoff freq** returns the high cutoff frequency. The cutoff frequency is the frequency with one-half magnitude response.
- **Iow cutoff freq** returns the low cutoff frequency. The cutoff frequency is the frequency with one-half magnitude response.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **ripple ratio** returns the side-lobe attenuation provided by the Dolph-Chebyshev Window in decibels.

DFD Kaiser Design VI

Owning Palette: Advanced FIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a finite impulse response (FIR) filter by using the <u>Kaiser Window</u> <u>method</u>.

You can use the DFD Kaiser Order Estimation VI to estimate order.

<u>Details</u>



Place on the block diagram Find on the Functions palette

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- **order** specifies the filter order. The value of **order** must be greater than zero. The default is 31. **order** +1 equals the number of coefficients or filter taps. Increasing the value can narrow the transition band.
- **high cutoff freq** specifies the high cutoff frequency in hertz. The value must be greater than **low cutoff freq**. The default is 0.45. This VI uses this input only for bandpass and bandstop filter design. The cutoff frequency is the frequency with one-half magnitude response.
- **Iow cutoff freq** specifies the low cutoff frequency in hertz. The default is 0.12. The cutoff frequency is the frequency with one-half magnitude response.
- error in describes error conditions that occur before this VI or

function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **beta** controls the main lobe width and the ratio of the main lobe to secondary lobes. The default is 2.46. If you increase **beta** while **order** remains constant, the side lobe decreases in amplitude but the transition bandwidth increases.
- **filter out** returns a new filter.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code**

is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Kaiser Design Details

The Kaiser window is ripple-adjustable, as described by the following equation:

$$W_{Kaiser}(n) = \begin{cases} I_0 \left[\beta \sqrt{1 - \left(\frac{2n}{M}\right)^2} \right] & |n| \le M/2 \\ 0 & |n| > M/2 \end{cases}$$

where $I_0(x)$ is the zero-order modified Bessel function of the first kind, β is the **beta** input that controls the main lobe width and the ratio of the main lobe to secondary lobes, and *M* represents the size of the Kaiser window.

DFD Kaiser Order Estimation VI

Owning Palette: Advanced FIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Estimates the order of a finite impulse response (FIR) filter that you designed by using the <u>Kaiser Window method</u>.



Place on the block diagram Find on the Functions palette

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass
S	Bandhaaa

- 2 Bandpass
- 3 Bandstop
- **freq specs** specifies the band edge frequencies of the filter.
 - **fpass 1** specifies the first passband edge frequency in hertz.
 - **fstop 1** specifies the first stopband edge frequency in hertz.
 - **fpass 2** specifies the second passband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
 - **fstop 2** specifies the second stopband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The

default is 60.

- **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- order option specifies filter order requirements. The default is MinEven, which yields the minimum even order for the filter to meet the specifications you set.



- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized

sampling frequency.

- estimated order returns the minimum order value that the filter requires to meet the specifications you set.
- **high cutoff freq** returns the high cutoff frequency. The cutoff frequency is the frequency with one-half magnitude response.
- **Iow cutoff freq** returns the low cutoff frequency. The cutoff frequency is the frequency with one-half magnitude response.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **beta** returns the beta value that the Kaiser window requires.

DFD Least Pth Norm Design VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Designs an infinite impulse response (IIR) or finite impulse response (FIR) filter with a frequency response that matches the response you request in terms of the least p^{th} norm algorithm. You can use either the iterative reweighted least square (IRLS) method or the Newton method that this VI provides to design a filter. You must <u>manually select the polymorphic instance</u> you want to use.

Details Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

-

■ Place on the block diagram ■ Find on the **Functions** palette

DFD Least Pth Norm Design by IRLS



- **pole radius constraint** specifies the maximum pole radius for the new filter. The default is 0.99. A small pole radius decreases the possibility of filter instability resulting from <u>finite precision effects</u>. However, a small pole radius can affect the potential sharpness of the magnitude response adversely. The range of valid values for **pole radius constraint** is (0, 1]. If you specify an invalid value, this VI ignores this input and applies no constraint to the pole radius.
- **filter type** specifies the type of filter that this VI creates.
 - 0 **Symmetric** (default)—Creates a filter with a symmetric impulse response.
 - 1 **Antisymmetric**—Creates a filter with an antisymmetric impulse response.
 - 2 **Differentiator**—Creates a filter with an antisymmetric impulse response. This filter type differs from **Antisymmetric** by having an additional weighting of 1/¹/₁₀ on amplitude response.
 - 3 **Hilbert**—Creates a filter with an antisymmetric impulse response. This filter type differs from **Antisymmetric** because this option obtains the phase by adding π to the **phase** input.
 - 4 Minimum Phase—Creates a minimum phase filter.
 - 5 **Maximum Phase**—Creates a maximum phase filter.
- **order** specifies the filter numerator and denominator order. For FIR filters, **order**+1 equals the number of coefficients. For IIR filters, the numerator order+1 equals the number of forward coefficients and the denominator order+1 equals the number of reverse coefficients.

numerator specifies the numerator order. The value must

be greater than zero. The default is 5.

- **denominator** specifies the denominator order. The value must be greater than or equal to zero. The default is 5. If you set **denominator** to 0, this VI creates a digital FIR filter. Otherwise, this VI creates a digital IIR filter.
- **band specs** specifies the target frequency response that the filter frequency response fits. Each element in the array represents one frequency band specification. You can enter one or more points in ascending order to describe the frequency response in each band. This VI connects the points to form the continuous ideal frequency response for the band. The frequency range between two consecutive bands is a transition band.
 - **freq** specifies one frequency point in hertz.
 - **magnitude** specifies the magnitude in relation to **freq** using a linear scale.
 - weight specifies the relative importance of the ripple size. Increasing weight reduces the ripple size and brings the filter closer to the frequency response specified in freq. This VI linearly interpolates the weight values of the frequencies between points. For example, to design a lowpass filter whose passband ripple is half the stopband ripple, set the passband weight to 2 and the stopband weight to 1.
 - **phase** specifies the phase in relation to **freq**.
- **group delay** specifies the group delay for all bands. The default is 5. You can specify any real number. For a specific frequency, this VI adjusts the phase response using the **phase** input in the **band specs** in combination with **group delay**.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by

wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **p** specifies the order of norm. The value of **p** must be between 2 and 128. When **p** equals 2, this VI returns the least squares solution. As you increase the value of **p**, the solution asymptotically approaches an equi-ripple magnitude solution. When **p** equals 128, this VI returns a nearly equi-ripple magnitude response. The default is 128. This input corresponds to the *p* parameter in the equations in the <u>Details</u> section of this topic.
- **filter out** returns a new filter.
- **pth norm** returns the p^{th} norm for the design.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Least Pth Norm Design by Newton



- **pole radius constraint** specifies the maximum pole radius for the new filter. The default is 0.99. A small pole radius decreases the possibility of filter instability resulting from <u>finite precision effects</u>. However, a small pole radius can affect the potential sharpness of the magnitude response adversely. The range of valid values for **pole radius constraint** is (0, 1]. If you specify an invalid value, this VI ignores this input and applies no constraint to the pole radius.
- **filter type** specifies the type of filter that this VI creates.
 - 0 **Symmetric** (default)—Creates a filter with a symmetric impulse response.
 - 1 **Antisymmetric**—Creates a filter with an antisymmetric impulse response.
 - 2 **Differentiator**—Creates a filter with an antisymmetric impulse response. This filter type differs from **Antisymmetric** by having an additional weighting of 1/¹/₁₀ on amplitude response.
 - 3 **Hilbert**—Creates a filter with an antisymmetric impulse response. This filter type differs from **Antisymmetric** because this option obtains the phase by adding π to the **phase** input.
 - 4 Minimum Phase—Creates a minimum phase filter.
 - 5 **Maximum Phase**—Creates a maximum phase filter.
- **order** specifies the filter numerator and denominator order. For FIR filters, **order**+1 equals the number of coefficients. For IIR filters, the numerator order+1 equals the number of forward coefficients and the denominator order+1 equals the number of reverse coefficients.

numerator specifies the numerator order. The value must

be greater than zero. The default is 5.

- **denominator** specifies the denominator order. The value must be greater than or equal to zero. The default is 5. If you set **denominator** to 0, this VI creates a digital FIR filter. Otherwise, this VI creates a digital IIR filter.
- **band specs** specifies the target frequency response that the filter frequency response fits. Each element in the array represents one frequency band specification. You can enter one or more points in ascending order to describe the frequency response in each band. This VI connects the points to form the continuous ideal frequency response for the band. The frequency range between two consecutive bands is a transition band.
 - **freq** specifies one frequency point in hertz.
 - **magnitude** specifies the magnitude in relation to **freq** using a linear scale.
 - weight specifies the relative importance of the ripple size. Increasing weight reduces the ripple size and brings the filter closer to the frequency response specified in freq. This VI linearly interpolates the weight values of the frequencies between points. For example, to design a lowpass filter whose passband ripple is half the stopband ripple, set the passband weight to 2 and the stopband weight to 1.
 - **phase** specifies the phase in relation to **freq**.
- **group delay** specifies the group delay for all bands. The default is 5. You can specify any real number. For a specific frequency, this VI adjusts the phase response using the **phase** input in the **band specs** in combination with **group delay**.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by

wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **p** specifies the order of norm. The value of **p** must be between 2 and 128. When **p** equals 2, this VI returns the least squares solution. As you increase the value of **p**, the solution asymptotically approaches an equi-ripple magnitude solution. When **p** equals 128, this VI returns a nearly equi-ripple magnitude response. The default is 128. This input corresponds to the *p* parameter in the equations in the <u>Details</u> section of this topic.
- **filter out** returns a new filter.
- **pth norm** returns the p^{th} norm for the design.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Least Pth Norm Design Details

You can design either an IIR or FIR filter using this VI. The following equation shows the frequency response of an IIR filter with *N* zeroes and *M* poles:

$$H(\omega) = \frac{B(\omega)}{A(\omega)} = \frac{\sum_{n=0}^{N} b(n)e^{-j\omega n}}{1 + \sum_{n=1}^{M} \bar{e}(n)e^{-j\omega n}}$$

where $B(\omega)$ is the Fourier transform of the forward coefficients

 $A(\omega)$ is the Fourier transform of the reverse coefficients

b(n) is the set of forward coefficients

a(n) is the set of reverse coefficients

When M equals zero, the IIR filter reduces to an FIR filter. Usually, a(0) is normalized to one, as shown in the equation above.

Given a complex-valued ideal frequency response $D(\omega)$, the DFD Least Pth Norm Design VI designs optimal IIR filters using the least p^{th} norm algorithm. The VI uses either complex approximation or magnitude approximation to create the design. The following equation is the complex approximation:

$$\left\| E \right\|_{p} = \left(\sum_{i=0}^{L-1} (W(i) |\mathcal{H}(\omega_{i}) - D(\omega_{i})|)^{p} \right)^{\frac{1}{p}}$$

where W(i) is a positive weight at the *i*th frequency point

H is the response of the designed filter

D is the target response

L is the number of frequency points used to perform the calculation

p is the p^{th} norm

The following equation is the magnitude approximation:

$$\left\| \mathcal{E} \right\|_{p} = \left(\sum_{i=0}^{L-1} (W(i) \left\| \mathcal{H}(\omega_{i}) \right| - \left| \mathcal{D}(\omega_{i}) \right\| \right)^{p} \right)^{\frac{1}{p}}$$

The equations are minimized in terms of filter coefficients a(n) and b(n).

If you set the filter type input to **Minimum Phase** or **Maximum Phase**, the DFD Least Pth Norm Design VI performs magnitude approximation and ignores the phase information of $D(\omega)$. If you set the filter type input to **Symmetric**, **Antisymmetric**, **Differentiator**, or **Hilbert**, the DFD Least Pth Norm Design VI uses complex approximation.

The phase response of the filter $\theta_{\text{overall}}(\omega)$ is specified by **phase** in the **band specs** $\theta_{\text{specified}}(\omega)$ and **group delay** τ_{gp} , as shown in the following equation:

 $\theta_{\text{overall}}(\omega) = -\tau_{\text{gp}}\omega + \theta_{\text{specified}}(\omega)$

Examples

Refer to the following VIs for examples of using the DFD Least Pth Norm Design VI:

• Arbitrary Shape Lowpass Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

Open example Browse related examples

• Lpth Norm Complex Approximation-Compensate Channel Distortion VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

■ Open example ■ Browse related examples

• LPth Norm IIR Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

■ Open example ■ Browse related examples

• LPth Norm Weighting Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

Open example Browse related examples

DFD Remez Design VI

Owning Palette: Advanced FIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates an equi-ripple filter using the Remez exchange method.

Details Examples



■ Place on the block diagram ■ Find on the **Functions** palette

minimum order specifies whether this VI uses the filter order you specify or calculates the minimum filter order. The default is user defined. If you select minEven or minOdd, this VI ignores the order input and determines the minimum required filter order. You also must provide a ripple constraint for each band in the ripple constraint input of band specs.

0 **user defined**—Specifies to use the value in **order**.

1 **minEven**—Specifies to calculate the minimum even order value and use this value to design the filter.

2 **minOdd**—Specifies to calculate the minimum odd order value and use this value to design the filter.

filter type specifies the <u>type of filter</u> that this VI creates.

0 **Symmetric** (default)—Creates a filter with a symmetric impulse response.

1 Antisymmetric—Creates a filter with an antisymmetric impulse response. The Symmetric and Antisymmetric options determine the symmetry of the filter impulse response and consequently determine the symmetry of the zero phase frequency amplitude response specified by the band specifications.

- 2 **Differentiator**—Creates a filter with an antisymmetric impulse response. This filter type differs from **Antisymmetric** by having an additional weighting of 1/ ∞ on amplitude response. This option uses a built-in weighting function that is inversely proportional to frequency to achieve a constant percentage error ripple versus the amplitude of the frequency response.
- 3 **Hilbert**—Creates a filter with an antisymmetric impulse response that behaves in the same way as **Antisymmetric**.
- 4 **Minimum Phase**—Creates a minimum phase filter. Minimum phase filters sometimes are called minimum energy delay or minimum delay filters. A minimum phase FIR filter has all the zeroes inside or on the unit circle of the z-plane.
- 5 **Maximum Phase**—Creates a maximum phase filter. A maximum phase FIR filter has a time-reversed impulse response of a minimum phase filter, where the zeroes of a maximum phase filter are all outside or on the unit circle of the z-plane.
- **order** specifies the filter order. The value of **order** must be greater than zero. The default is 20. **order** +1 equals the number of coefficients or filter taps. Increasing the value can narrow the transition band.
- **band specs** specifies the target frequency response that the filter frequency response fits. Each element of the array represents one frequency band specification. You can enter one or more points in ascending order to describe the frequency response in each band. This VI connects the points to form the continuous ideal frequency response for the band. The frequency range between two consecutive bands is a transition band. The frequency response you describe with the **band specs** input is the signed amplitude response. You can provide negative target **amplitude** values. However, if the **filter type** input is **Minimum Phase** or **Maximum Phase**, the frequency response you describe with the **band specs** input is the magnitude response, and all target **amplitude** values must be positive.
 - **freq** specifies one frequency point in hertz.
 - **amplitude** specifies the amplitude on **freq** in linear scale.

- weight specifies the relative importance of the ripple size. Increasing weight reduces the ripple size and brings the filter closer to the frequency response specified in **freq**. This VI linearly interpolates the weight values of the frequencies between points. For example, to design a lowpass filter whose passband ripple is half the stopband ripple, set the passband weight to 2 and the stopband weight to 1.
- **ripple constraint** specifies the ripple constraint in the current band using linear scale. The default is 0, which means that no constraint is applied. If you enter a positive value, the ripple level in the current band will be at a level lower than the value you specify. You must leave at least one band unconstrained if you set **minimum order** to user defined.
- **freqs of exact gain** specifies frequency points where the amplitude must have exactly the same value as the **amplitude** input in **band specs**. If a frequency point does not appear in **band specs**, this VI interpolates the amplitude linearly.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- filter out returns a new filter.
- **actual ripples** returns the actual ripple magnitude in each band specified in **band specs**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Remez Design Details

In the Remez exchange method, the filter frequency response best fits the target response in the Chebyshev sense. The Remez Design VI employs either complex approximation or magnitude approximation to create the design.

The design criterion for complex approximation is defined by the following equation:

 $\min\left\{\max\left\{\sum_{j=0}^{L-1} (W(j) \cdot \left| H(\omega_j) - D(\omega_j) \right|)\right\}\right\}$

The design criterion for magnitude approximation is defined by the following equation:

$$\min\left\{\max\left\{\sum_{j=0}^{L-1} (W(j) \cdot ||\mathcal{H}(\omega_j)| - |\mathcal{D}(\omega_j)||)\right\}\right\}$$

where D(wi) is the ideal frequency response, H(wi) is the frequency response of the designed filter, and W(i) is the positive weight at the *i*th frequency point. Symmetric, Antisymmetric, Differentiator, and Hilbert filter types use complex approximation. Minimum Phase and Maximum Phase filter types use magnitude approximation.

Examples

Refer to the following VIs for examples of using the DFD Remez Design VI:

• EquiRipple Filter Design - DFD VI: labview\examples\Digital Filter Design\AALXMPL

Open example Browse related examples

• Advanced Remez FIR Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

Open example Browse related examples

• Arbitrary Shape Lowpass Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

■ Open example ■ Browse related examples

DFD Remez Order Estimation VI

Owning Palette: Advanced FIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Estimates the order of a Remez equi-ripple filter.



Place on the block diagram Find on the Functions palette

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- freq specs specifies the band edge frequencies of the filter.
 fpass 1 specifies the first passband edge frequency in hertz.
 - **fstop 1** specifies the first stopband edge frequency in hertz.
 - **fpass 2** specifies the second passband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
 - **fstop 2** specifies the second stopband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The default is 60.
- **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- order option specifies filter order requirements. The default is MinEven, which yields the minimum even order for the filter to meet the specifications you set.



- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.

- estimated order returns the minimum order value that the filter requires to meet the specifications you set.
- **band specs** returns specifications such as frequency, amplitude, and weight in one or more bands.
 - **freq** returns one frequency point in hertz.
 - **amplitude** returns the amplitude on **freq** in linear scale.
 - weight returns the weight of the error on freq.
 - **ripple constraint** returns the ripple constraint in the current filter band in linear scale.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Windowed FIR Design VI

Owning Palette: Advanced FIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a finite impulse response (FIR) filter by using the window methods.

Example



 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

window specifies the time-domain window this VI uses.

0	None (default)
1	Hanning
2	Hamming
3	Blackman-Harris
4	Exact Blackman
5	Blackman
6	Flat Top
7	4 Term B-Harris
8	7 Term B-Harris
9	Low Sidelobe
30	Triangular

0	Lowpass (default)
1	Highpass
2	Bandpass

3 Bandstop

- **order** specifies the filter order. The value of **order** must be greater than zero. The default is 20. **order** +1 equals the number of coefficients or filter taps. Increasing the value can narrow the transition band.
- **high cutoff freq** specifies the high cutoff frequency in hertz. The value must be greater than **low cutoff freq**. The default is 0.45. This VI uses this input only for bandpass and bandstop filter design. The cutoff frequency is the frequency with one-half magnitude response.
- **Iow cutoff freq** specifies the low cutoff frequency in hertz. The default is 0.12. The cutoff frequency is the frequency with one-half magnitude response.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.

filter out returns a new filter.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the FIR Windowed Filter Design - DFD VI in the labview\examples\Digital Filter Design\AALXMPL directory for an example of using the DFD Windowed FIR Design VI.

■ Open example ■ Browse related examples

Advanced IIR Filter Design VIs

Owning Palette: Filter Design VIs

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Advanced IIR Filter Design VIs to estimate infinite impulse response (IIR) filter order and to create IIR filters from the filter specifications.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
<u>DFD</u> <u>Bessel</u> Design	Creates a digital Bessel infinite impulse response (IIR) filter.
<u>DFD</u> Butterworth Design	Creates a digital Butterworth infinite impulse response (IIR) filter.
DFD Butterworth Order Estimation	Estimates the Butterworth filter order.
<u>DFD</u> <u>Chebyshev</u> <u>Design</u>	Creates a digital Chebyshev infinite impulse response (IIR) filter.
DFD Chebyshev Order Estimation	Estimates the Chebyshev I filter order.
<u>DFD</u> <u>Elliptic</u> Design	Creates a digital Elliptic infinite impulse response (IIR) filter.
<u>DFD</u> <u>Elliptic</u>	Estimates the Elliptic filter order.

<u>Order</u> Estimation	
DFD Inverse Chebyshev Design	Creates a digital Inverse Chebyshev infinite impulse response (IIR) filter.
DFD Inverse Chebyshev Order Estimation	Estimates the Inverse Chebyshev filter order.
<u>DFD Least</u> <u>Pth Norm</u> <u>Design</u>	Designs an infinite impulse response (IIR) or finite impulse response (FIR) filter with a frequency response that matches the response you request in terms of the least p^{th} norm algorithm. You can use either the iterative reweighted least square (IRLS) method or the Newton method that this VI provides to design a filter. You must <u>manually select the</u> <u>polymorphic instance</u> you want to use.

DFD Bessel Design VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a digital Bessel infinite impulse response (IIR) filter.

Details Examples



■ Place on the block diagram ■ Find on the **Functions** palette

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- **order** specifies the digital filter order. The value of **order** must be greater than zero. The default is 2. If you set **filter type** to Lowpass or Highpass, the digital filter order you specify must equal the analog prototype filter order. If you set **filter type** to Bandstop or Bandpass, the digital filter order you specify must be an even number that equals two times the analog prototype filter order. Increasing the value can narrow the transition band.
- **high cutoff freq** specifies the high cutoff frequency in hertz. The value must be greater than **low cutoff freq**. The default is 0.45. This VI uses this input for bandpass and bandstop filter designs only. The cutoff frequency specifies the passband, or the region of an approximately linear phase response.
- **Iow cutoff freq** specifies the low cutoff frequency in hertz. The default is 0.12. The cutoff frequency specifies the passband, or the region of an approximately linear phase response.
- error in describes error conditions that occur before this VI or

function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **filter out** returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning.

DFD Bessel Design Details

Bessel filters have maximally flat response in both the passband and the stopband. The phase response in the passband of Bessel filters, which typically is the region of interest, is nearly linear. However, Bessel filters require high-order filters to provide a good approximation of the ideal filter response.

To achieve linear phase response without a Bessel filter, use FIR filter designs or use the <u>DFD Group Delay Compensator</u> VI to compensate the IIR filter group delay.

Examples

Refer to the following VIs for examples of using the DFD Bessel Design VI:

• IIR Filter Design - DFD VI: labview\examples\Digital Filter Design\AALXMPL

■ Open example ■ Browse related examples

• Bessel IIR Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

Open example Browse related examples

DFD Butterworth Design VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a digital Butterworth infinite impulse response (IIR) filter.

You can use the **DFD Butterworth Order Estimation** VI to estimate **order**.

Details Examples



■ Place on the block diagram ■ Find on the **Functions** palette

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- **order** specifies the digital filter order. The value of **order** must be greater than zero. The default is 2. If you set **filter type** to Lowpass or Highpass, the digital filter order you specify must equal the analog prototype filter order. If you set **filter type** to Bandstop or Bandpass, the digital filter order you specify must be an even number that equals two times the analog prototype filter order. Increasing the value can narrow the transition band.
- **high cutoff freq** specifies the high cutoff frequency in hertz. The value must be greater than **low cutoff freq**. The default is 0.45. This VI uses this input only for bandpass and bandstop filter design. The cutoff frequency that you specify corresponds to the half-power frequency or the 3 dB frequency.
- **Iow cutoff freq** specifies the low cutoff frequency in hertz. The default is 0.12. The cutoff frequency that you specify corresponds to the half-power frequency or the 3 dB frequency.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- filter out returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Butterworth Design Details

Butterworth filters have the following characteristics:

- Smooth frequency response at all frequencies and monotonically decreasing or increasing magnitude response in the transition band.
- Maximally flat at frequencies of zero and one-half the sampling frequency.

Butterworth filters do not always provide an acceptable accurate approximation of the ideal frequency response because the filter has slow roll off in the transition band. Use Chebyshev, Inverse Chebyshev, or Elliptic filters to achieve a sharper roll off.

Examples

Refer to the following VIs for examples of using the DFD Butterworth Design VI:

• Extract the Sine Wave - DFD VI: labview\examples\Digital Filter Design\AALXMPL

■ Open example ■ Browse related examples

• IIR Filter Design - DFD VI: labview\examples\Digital Filter Design\AALXMPL

Open example Browse related examples

DFD Butterworth Order Estimation VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Estimates the Butterworth filter order.



■ Place on the block diagram ■ Find on the **Functions** palette

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- **freq specs** specifies the band edge frequencies of the filter.
 - **fpass 1** specifies the first passband edge frequency in hertz.
 - **fstop 1** specifies the first stopband edge frequency in hertz.
 - **fpass 2** specifies the second passband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
 - **fstop 2** specifies the second stopband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The default is 60.

TF

dB/linear? specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- estimated order returns the minimum order value that the filter requires to meet the specifications you set.
- **high cutoff freq** returns the high cutoff frequency. The cutoff frequency corresponds to the half-power frequency or the 3 dB frequency.
- **Iow cutoff freq** returns the low cutoff frequency. The cutoff frequency corresponds to the half-power frequency or the 3 dB frequency.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Chebyshev Design VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a digital Chebyshev infinite impulse response (IIR) filter.

You can use the DFD Chebyshev Order Estimation VI to estimate order.

Details Example



■ Place on the block diagram ■ Find on the **Functions** palette

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- **order** specifies the digital filter order. The value of **order** must be greater than zero. The default is 2. If you set **filter type** to Lowpass or Highpass, the digital filter order you specify must equal the analog prototype filter order. If you set **filter type** to Bandstop or Bandpass, the digital filter order you specify must be an even number that equals two times the analog prototype filter order. Increasing the value can narrow the transition band.
- **high cutoff freq** specifies the high cutoff frequency in hertz. The value must be greater than **low cutoff freq**. The default is 0.45. This VI uses this input only for bandpass and bandstop filter design. The cutoff frequency that you specify corresponds to the edge frequency of the passband.
- **Iow cutoff freq** specifies the low cutoff frequency in hertz. The default is 0.12. The cutoff frequency that you specify corresponds to the edge frequency of the passband.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **passband ripple** specifies the ripple level in the passband in decibels. The value must be greater than zero. The default is 1.
- filter out returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code**

is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Chebyshev Design Details

Chebyshev filters have the following characteristics:

- Minimum peak error in the passband.
- Equi-ripple magnitude response in the passband.
- Monotonically decreasing or increasing magnitude response in the transition band and the stopband.
- For filters of the same order, the roll off in the transition band is sharper than that of Butterworth filters.

Example

Refer to the IIR Filter Design - DFD VI in the labview\examples\Digital Filter Design\AALXMPL directory for an example of using the DFD Chebyshev Design VI.

■ Open example ■ Browse related examples

DFD Chebyshev Order Estimation VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Estimates the Chebyshev I filter order.



■ Place on the block diagram ■ Find on the **Functions** palette

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- freq specs specifies the band edge frequencies of the filter.
 fpass 1 specifies the first passband edge frequency in hertz.
 - **fstop 1** specifies the first stopband edge frequency in hertz.
 - **fpass 2** specifies the second passband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
 - **fstop 2** specifies the second stopband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The default is 60.

- **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- estimated order returns the minimum order value that the filter requires to meet the specifications you set.
- **high cutoff freq** returns the high cutoff frequency. The cutoff frequency corresponds to the edge frequency of the passband.
- **Iow cutoff freq** returns the low cutoff frequency. The cutoff frequency corresponds to the edge frequency of the passband.
- error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **passband ripple** returns the ripple level in the passband in decibels.

DFD Elliptic Design VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a digital Elliptic infinite impulse response (IIR) filter.

You can use the **DFD Elliptic Order Estimation** VI to estimate **order**.

Details Examples



■ Place on the block diagram ■ Find on the **Functions** palette

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- **order** specifies the digital filter order. The value of **order** must be greater than zero. The default is 2. If you set **filter type** to Lowpass or Highpass, the digital filter order you specify must equal the analog prototype filter order. If you set **filter type** to Bandstop or Bandpass, the digital filter order you specify must be an even number that equals two times the analog prototype filter order. Increasing the value can narrow the transition band.
- **high cutoff freq** specifies the high cutoff frequency in hertz. The value must be greater than **low cutoff freq**. The default is 0.45. This VI uses this input only for bandpass and bandstop filter design. The cutoff frequency that you specify corresponds to the edge frequency of the passband.
- **Iow cutoff freq** specifies the low cutoff frequency in hertz. The default is 0.12. The cutoff frequency that you specify corresponds to the edge frequency of the passband.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **ripples** specifies the passband ripple and stopband attenuation of the filter in decibels.
 - **passband ripple** specifies the ripple level in the passband in decibels. The value must be greater than zero. The default is 1.
 - **stopband atten** specifies the stopband attenuation in decibels. The value must be greater than zero. The default is 60.
- filter out returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Elliptic Design Details

Elliptic filters have the following characteristics:

- Minimum peak error in the passband and stopband.
- Equi-ripple magnitude response in the passband and stopband.
- Compared with same-order Butterworth or Chebyshev filters, Elliptic filters have the sharpest transition band, which accounts for their widespread use.

Examples

Refer to the following VIs for examples of using the DFD Elliptic Design VI:

• IIR Filter Design - DFD VI: labview\examples\Digital Filter Design\AALXMPL

■ Open example ■ Browse related examples

• Elliptic IIR Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

Open example Browse related examples

DFD Elliptic Order Estimation VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Estimates the Elliptic filter order.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

0 Lowpass (default) 1 Highpass 2 Bandnass
1 Highpass
2 Bandnass
2 Dunupuss
3 Bandstop

- **freq specs** specifies the band edge frequencies of the filter.
 - **fpass 1** specifies the first passband edge frequency in hertz.
 - **fstop 1** specifies the first stopband edge frequency in hertz.
 - **fpass 2** specifies the second passband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
 - **fstop 2** specifies the second stopband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The

default is 60.

- **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- estimated order returns the minimum order value that the filter requires to meet the specifications you set.
- **high cutoff freq** returns the high cutoff frequency. The cutoff frequency corresponds to the edge frequency of the passband.
- **Iow cutoff freq** returns the low cutoff frequency. The cutoff frequency corresponds to the edge frequency of the passband.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **ripples** returns the passband ripple and stopband attenuation of the filter in decibels.
 - **passband ripple** returns the ripple level in the passband in decibels.
 - **stopband atten** returns the stopband attenuation in decibels.

Example

Refer to the Elliptic IIR Filter Design VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Conventional directory for an example of using the DFD Elliptic Order Estimation VI.

■ Open example ■ Browse related examples

DFD Inverse Chebyshev Design VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a digital Inverse Chebyshev infinite impulse response (IIR) filter.

You can use the DFD Inverse Chebyshev Order Estimation VI to estimate order.

Details Example



 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- **order** specifies the digital filter order. The value of **order** must be greater than zero. The default is 2. If you set **filter type** to Lowpass or Highpass, the digital filter order you specify must equal the analog prototype filter order. If you set **filter type** to Bandstop or Bandpass, the digital filter order you specify must be an even number that equals two times the analog prototype filter order. Increasing the value can narrow the transition band.
- **high cutoff freq** specifies the high cutoff frequency in hertz. The value must be greater than **low cutoff freq**. The default is 0.45. This VI uses this input only for bandpass and bandstop filter design. The cutoff frequency that you specify corresponds to the edge frequency of the stopband.
- **Iow cutoff freq** specifies the low cutoff frequency in hertz. The default is 0.12. The cutoff frequency that you specify corresponds

to the edge frequency of the stopband.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **stopband atten** specifies the stopband attenuation in decibels. The value must be greater than zero. The default is 60.
- **filter out** returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Inverse Chebyshev Design Details

Inverse Chebyshev filters, also known as Chebyshev II or Type II Chebyshev filters, have the following characteristics:

- Minimum peak error in the stopband.
- Equi-ripple magnitude response in the stopband.
- Monotonically decreasing or increasing magnitude response in the passband.
- For filters of the same order, the roll off in the transition band is sharper than that of Butterworth filters.

Example

Refer to the IIR Filter Design - DFD VI in the labview\examples\Digital Filter Design\AALXMPL directory for an example of using the DFD Inverse Chebyshev Design VI.

■ Open example ■ Browse related examples

DFD Inverse Chebyshev Order Estimation VI

Owning Palette: Advanced IIR Filter Design VIs

Installed With: Digital Filter Design Toolkit

Estimates the Inverse Chebyshev filter order.



■ Place on the block diagram ■ Find on the **Functions** palette

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- freq specs specifies the band edge frequencies of the filter.
 fpass 1 specifies the first passband edge frequency in hertz.
 - **fstop 1** specifies the first stopband edge frequency in hertz.
 - **fpass 2** specifies the second passband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
 - **fstop 2** specifies the second stopband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The default is 60.

- **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- estimated order returns the minimum order value that the filter requires to meet the specifications you set.
- **high cutoff freq** returns the high cutoff frequency. The cutoff frequency corresponds to the edge frequency of the stopband.
- **Iow cutoff freq** returns the low cutoff frequency. The cutoff frequency corresponds to the edge frequency of the stopband.
- error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **stopband atten** returns the stopband attenuation in decibels.

Special Filter Design VIs

Owning Palette: Filter Design VIs

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Special Filter Design VIs to create notch peak, infinite impulse response (IIR) comb, maximally flat, narrowband, and other special filters.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
DFD Arbitrary Group Delay Design	Creates an allpass filter with a <mark>group delay</mark> that you specify.
<u>DFD Group</u> <u>Delay</u> Compensator	Compensates the group delay of a filter.
<u>DFD IIR</u> <u>Comb</u> <u>Design</u>	Creates an infinite impulse response (IIR) <u>comb filter</u> . You must <u>manually select the polymorphic instance</u> you want to use.
<u>DFD IIR</u> <u>Notch Peak</u> Design	Designs a <u>notch or peak filter</u> in which the notch or peak is located at the center frequency. You must <u>manually</u> <u>select the polymorphic instance</u> you want to use.
DFD Maxflat Design	Creates a lowpass finite impulse response (FIR) or infinite impulse response (IIR) filter with a magnitude frequency response that is maximally flat at 0 and at half the sampling frequency.
<u>DFD</u> <u>Narrowband</u> <u>Filter Design</u>	Creates a narrowband filter using the interpolated finite impulse response (FIR) technique.

DFD Arbitrary Group Delay Design VI

Owning Palette: Special Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates an allpass filter with a group delay that you specify.

The group delay in the new filter is optimal in terms of the least pth norm.

<u>Example</u>



■ Place on the block diagram ■ Find on the **Functions** palette

- **pole radius constraint** specifies the maximum pole radius for the new filter. The default is 0.99. A small pole radius constraint decreases the possibility of filter instability caused by finite precision effects. However, a small value of the pole radius constraint can adversely affect the potential sharpness of the group delay response. The valid values for **pole radius constraint** are within the range (0, 1]. This VI ignores the input and applies no constraint to the radius if a value is beyond the range.
- **ITT integer offset?** specifies whether the group delay offset between the group delay of the filter and predefined specifications is an integer or any real number. The default is TRUE. If you set **integer offset?** to FALSE, the group delay offset is any real number. Using an integer can slightly increase delay.
- **order** specifies the filter order. The value of **order** must be greater than zero. The default is 10.
- **band specs** specifies the group delay response you want for the filter, for one or more frequency bands.
 - **freq** specifies the frequency point.
 - **group delay** specifies the group delay you want for the frequency point.
 - weight defines the relative significance of the group delay

you want. Increasing the value of **weight** adds precision to group delay for the frequency point.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- filter out returns a new filter.
- **offset** returns the group delay offset between the group delay of **filter out** and the group delay specified in **band specs**. The **integer offset?** input determines whether **offset** is an integer or a floating-point number.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

TF	status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
132	code is the error or warning code. If status is TRUE, code is a nonzero <u>error code</u> . If status is FALSE, code is 0 or a warning code.
Jabc	source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Arbitrary Group Delay Filter Design VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Conventional directory for an example of using the DFD Arbitrary Group Delay Design VI.

Open example Browse related examples

DFD Group Delay Compensator VI

Owning Palette: Special Filter Design VIs

Installed With: Digital Filter Design Toolkit

Compensates the group delay of a filter.

The resulting filter, **compensated filter**, has the same magnitude response as the original filter but includes a constant group delay in the frequency ranges you specify.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

- **pole radius constraint** specifies the maximum pole radius for the new filter. The default is 0.99. A small pole radius constraint decreases the possibility of filter instability caused by finite precision effects. However, a small value of the pole radius constraint can adversely affect the potential sharpness of the group delay response. The valid values for **pole radius constraint** are within the range (0, 1]. This VI ignores the input and applies no constraint to the radius if a value is beyond the range.
- **integer group delay?** specifies whether group delay for the new filter is an integer. The default is TRUE. If you set **integer group delay?** to FALSE, group delay is any real number.
- **filter in** specifies the input filter.
- **compensator order** specifies the order of the allpass filter that **compensator** returns, which compensates group delay. **compensator order** must be an even number.
- **freq ranges** specifies one or more frequency ranges in which this VI compensates the group delay.
 - **Iow** specifies the low end of the frequency range. The default is 0.

- **high** specifies the high end of the frequency range. The default is 0.2.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **compensated filter** returns the new filter with compensated group delay.
- compensator returns the allpass filter that the VI uses to compensate the group delay.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Group Delay Compensation VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Conventional directory for an example of using the DFD Group Delay Compensator VI.

■ Open example ■ Browse related examples

DFD IIR Comb Design VI

Owning Palette: Special Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates an infinite impulse response (IIR) <u>comb filter</u>. You must <u>manually</u> <u>select the polymorphic instance</u> you want to use.

•

Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD IIR Comb Design by N and Bandwidth



filter type specifies the type of IIR comb filter.

0	Notch Type I (default)
1	Notch Type II
2	Peak Type I
3	Peak Type II

- # notches/peaks specifies the number of notches or peaks in the full frequency band from 0 to fs. The default is 10.
- **Df** defines the full bandwidth at the level of -Ab. The default is 0.02. **Df** represents Δf .
- **Ab** specifies the attenuation that corresponds to the bandwidth. The default is 3.0103, which corresponds to 3 dB bandwidth, the commonly used bandwidth of a filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**

is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- filter out returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD IIR Comb Design by f0 and Bandwidth



filter type specifies the type of IIR comb filter.

0	Notch Type I (default)
1	Notch Type II
2	Peak Type I
3	Peak Type II

- **f0** specifies the center frequency of the first nonzero notch or peak. The default is 0.1. The value of **fs/f0** must be an integer for a Type I comb filter design. The value of **fs/(f0***2) must be an integer for a Type II comb filter design.
- **Df** defines the full bandwidth at the level of -Ab. The default is 0.02. **Df** represents Δf .
- **Ab** specifies the attenuation that corresponds to the bandwidth. The default is 3.0103, which corresponds to 3 dB bandwidth, the commonly used bandwidth of a filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- filter out returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD IIR Comb Design VI:

• IIR Comb Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

■ Open example ■ Browse related examples

• Noise Cancellation for ECG Signal by Notch Filter VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

■ Open example ■ Browse related examples

DFD IIR Notch Peak Design VI

Owning Palette: <u>Special Filter Design VIs</u>

Installed With: Digital Filter Design Toolkit

Designs a <u>notch or peak filter</u> in which the notch or peak is located at the center frequency. You must <u>manually select the polymorphic instance</u> you want to use.

•

Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD IIR Notch Peak Design By Q Factor





filter type specifies the type of filter that this VI creates.



f0 specifies the center frequency of the notch or peak. The value of f0 must be greater than 0 but less than half the value of fs. The default is 0.10.

- **Q factor** specifies the Q factor of the filter, which equals the quotient of center frequency and corresponding bandwidth. The default is 20. The Q factor reflects the relative sharpness of the filter notch or peak. Increasing the Q factor sharpens the notch or peak.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **filter out** returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD IIR Notch Peak Design By Bandwidth



filter type specifies the <u>type of filter</u> that this VI creates.



- f0 specifies the center frequency of the notch or peak. The value of f0 must be greater than or equal to 0 but less than half the value of fs. The default is 0.10.
- **Df** defines the full bandwidth at the level of -Ab. The default is 0.02. **Df** represents Δf .
- **Ab** specifies the attenuation that corresponds to the bandwidth. The default is 3.0103, which corresponds to 3 dB bandwidth, the commonly used bandwidth of a filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- filter out returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD IIR Notch Peak Design VI:

• IIR Notch Peak Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

Open example Browse related examples

 Noise Cancellation for ECG Signal by Notch Filter VI: labview\examples\Digital Filter Design\Floating-Point Filters\Conventional

■ Open example ■ Browse related examples

DFD Maxflat Design VI

Owning Palette: Special Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a lowpass finite impulse response (FIR) or infinite impulse response (IIR) filter with a magnitude frequency response that is maximally flat at 0 and at half the sampling frequency.

You can specify whether this VI creates an FIR or IIR filter.



■ Place on the block diagram ■ Find on the **Functions** palette

filter type specifies the <u>type of filter</u> that this VI creates.



- **numerator order** sets the FIR filter order or the numerator order of the IIR filter. The value must be greater than zero. The default is 2.
- denominator order sets the denominator order of the IIR filter. The default is 2. The value must be greater than or equal to zero if you set filter type to IIR. If you set filter type to FIR or Symmetric FIR, this VI ignores this input.

3dB cutoff freq specifies the cutoff frequency in hertz at which the magnitude response of the filter equals –3 dB. The default is 0.25.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- filter out returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Narrowband Filter Design VI

Owning Palette: Special Filter Design VIs

Installed With: Digital Filter Design Toolkit

Creates a narrowband filter using the interpolated finite impulse response (FIR) technique.

To filter data, wire the **narrowband filter out** output to the **narrowband filter** input of the <u>DFD Narrowband Filtering</u> VI.

Examples



■ Place on the block diagram ■ Find on the **Functions** palette

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass
2	Bandpass
3	Bandstop

- **freq specs** specifies the band edge frequencies of the filter.
 - **fpass 1** specifies the first passband edge frequency in hertz.
 - **fstop 1** specifies the first stopband edge frequency in hertz.
 - **fpass 2** specifies the second passband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
 - **fstop 2** specifies the second stopband edge frequency in hertz. This VI ignores this input for lowpass and highpass filters.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.

- **passband** specifies the ripple level in the passband. The default is 0.1.
- **stopband** specifies the ripple level in the stopband. The default is 60.
- **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs** specifies the sampling frequency in hertz. The value must be greater than zero. The default is 1, which is the normalized sampling frequency.
- **narrowband filter out** contains the narrowband filter.
 - **multirate filters** contains the multirate filters this VI uses to construct the narrowband filter.

filter type contains the type of narrowband filter.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
Examples

Refer to the following VIs for examples of using the DFD Narrowband Filter Design VI:

• Narrowband Filtering - DFD VI: labview\examples\Digital Filter Design\AALXMPL

■ Open example ■ Browse related examples

- Narrowband Filter Design and Processing VI: labview\examples\Digital Filter Design\Floating-Point Filters\Multirate
 - Open example Browse related examples

Fixed-Point Tools VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Fixed-Point Tools VIs to quantize coefficients, model the behavior of fixed-point filters, simulate filtering processes, generate statistics reports, and generate fixed-point target code.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
DFD Convert FXP to Integer	Converts fixed-point numbers to integers. You must manually select the polymorphic instance you want to use.
DFD Convert Integer to FXP	Converts integers to fixed-point numbers. Wire data to the I16 input to determine the polymorphic instance to use or manually select the instance.
<u>DFD FXP</u> <u>Code</u> <u>Generator</u>	Generates fixed-point code from a fixed-point filter, including fixed-point LabVIEW field-programmable gate array (FPGA) code, integer LabVIEW code, and C code. You must <u>manually select the polymorphic instance</u> you want to use.
DFD FXP Coef Report	Generates a text report about the coefficients of a fixed- point filter.
<u>DFD FXP Get</u> Quantizer	Retrieves quantizer settings from a fixed-point filter. You must <u>manually select the polymorphic instance</u> you want to use.
DFD FXP Modeling	Creates a fixed-point filter model according to the input and output word length settings.
DFD FXP Postprocessing	Converts the output signal of a fixed-point filter from a fixed-point integer representation to a floating-point representation. You must <u>manually select the</u> polymorphic instance you want to use.
DFD FXP	Quantizes the coefficients of a floating-point filter and

<u>Quantize Coef</u>	generates a fixed-point filter. You must <u>manually select</u> <u>the polymorphic instance</u> you want to use.
<u>DFD FXP Set</u> Quantizer	Sets a quantizer or quantizers in a fixed-point filter. You must <u>manually select the polymorphic instance</u> you want to use.
DFD FXP Simulation Report	Creates a text report of filtering statistics from the <u>DFD</u> <u>FXP Simulation</u> VI or the <u>DFD FXP Simulation with</u> <u>State</u> VI.
<u>DFD FXP</u> <u>Simulation with</u> <u>State</u>	Simulates the filtering process with initial internal states and generates the filtering statistics report for a fixed- point filter. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.
DFD FXP Simulation	Simulates the filtering process continuously and generates a filtering statistics report for a fixed-point filter. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.

DFD Convert FXP to Integer VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Converts fixed-point numbers to integers. You must <u>manually select the</u> <u>polymorphic instance</u> you want to use.

▼

Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD Convert FXP to I16 (scalar)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients	a/k	(default)
---	--------------	-----	-----------

1 Input

2 Output

- 3 Multiplicand
- 4 Product
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0	Saturation
1	Wrap (default)

0	Nearest
1	Truncation (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **FXP value** specifies the fixed-point number you want to convert.
- **I16** returns a 16-bit signed integer.

DFD Convert FXP to I32 (scalar)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients	a/k (defau	lt)
---	--------------	-------------------	-----

1 Input

2 Output

- 3 Multiplicand
- 4 **Product**
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0	Saturation
1	Wrap (default)

0	Nearest	
1	Truncation	(default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **FXP value** specifies the fixed-point number you want to convert.
- **I32** returns a 32-bit signed integer.

DFD Convert FXP to I8 (scalar)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients	a/k	(default)
---	--------------	-----	-----------

1 Input

2 Output

- 3 Multiplicand
- 4 Product
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0	Saturation
1	Wrap (default)

0	Nearest	
1	Truncation	(default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **FXP value** specifies the fixed-point number you want to convert.
- **I8** returns an 8-bit signed integer.

DFD Convert FXP to I16 (vector)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)

1 Input

- 2 **Output**
- 3 Multiplicand
- 4 **Product**
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0	Saturation
1	Wrap (default)

0	Nearest
1	Truncation (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **FXP values** specifies the fixed-point numbers you want to convert.
- **[116 values** returns an array of 16-bit signed integers.

DFD Convert FXP to I32 (vector)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)

1 Input

2 Output

- 3 Multiplicand
- 4 Product
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0	Saturation
1	Wrap (default)

0	Nearest
1	Truncation (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **FXP values** specifies the fixed-point numbers you want to convert.
- **I32 values** returns an array of 32-bit signed integers.

DFD Convert FXP to I8 (vector)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)

1 Input

2 **Output**

- 3 Multiplicand
- 4 **Product**
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0	Saturation
1	Wrap (default)

0	Nearest
1	Truncation (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **FXP values** specifies the fixed-point numbers you want to convert.
- **I8 values** returns an array of 8-bit signed integers.

Examples

Refer to the following VIs for examples of using the DFD Convert FXP to Integer VI:

• Conversion between FXP and Integer VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• How to Build Coefficients Quantizer VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

■ Open example ■ Browse related examples

DFD Convert Integer to FXP VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Converts integers to fixed-point numbers. Wire data to the **I16** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

•

Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD Convert I16 to FXP (scalar)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)

1 Input

- 2 **Output**
- 3 Multiplicand
- 4 **Product**
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0		Saturation
1	,	Wrap (default)

- **rounding mode** specifies the <u>rounding mode</u> this VI uses in the quantizer.
 - 0 Nearest
 - 1 **Truncation** (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **I16** specifies a 16-bit signed integer.
- **FXP value** returns a fixed-point number.

DFD Convert I32 to FXP (scalar)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)

1 Input

2 **Output**

- 3 Multiplicand
- 4 Product
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0	Saturation
1	Wrap (default)

rounding mode specifies the <u>rounding mode</u> this VI uses in the quantizer.

0	Nearest	
	-	

1 Truncation (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **I32** specifies a 32-bit signed integer.
- **FXP value** returns a fixed-point number.

DFD Convert I8 to FXP (scalar)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)

1 Input

- 2 **Output**
- 3 Multiplicand
- 4 Product
- 5 **Sum**
- 6 Delay
- 7 **Coefficients b/v**
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0		Saturation	
1	,	Wrap (default)	

0	Nearest
1	Truncation (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **I8** specifies an 8-bit signed integer.
- **FXP value** returns a fixed-point number.

DFD Convert I16 to FXP (vector)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)

1 Input

- 2 **Output**
- 3 Multiplicand
- 4 **Product**
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0	Saturation	
1	Wrap (default)	

rounding mode specifies the <u>rounding mode</u> this VI uses in the quantizer.

0	Nearest	
4	— 1 ²	<u> </u>

1 **Truncation** (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **I16 values** specifies an array of 16-bit signed integers.
- **FXP values** returns an array of fixed-point numbers.

DFD Convert I32 to FXP (vector)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)

1 Input

2 **Output**

- 3 Multiplicand
- 4 Product
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.
- **overflow mode** specifies how this VI <u>handles overflows and</u> <u>underflows</u> in the quantizer.

0	Saturation	
1	Wrap (default)	

0	Nearest	
1	Truncation	(default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **I32 values** specifies an array of 32-bit signed integers.
- **FXP values** returns an array of fixed-point numbers.

DFD Convert I8 to FXP (vector)



quantization settings specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)

1 Input

- 2 **Output**
- 3 Multiplicand
- 4 **Product**
- 5 **Sum**
- 6 Delay
- 7 Coefficients b/v
- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.

0		Saturation	
1	,	Wrap (default)	

rounding mode specifies the <u>rounding mode</u> this VI uses in the quantizer.

0	Nearest	
1	T	

1 **Truncation** (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **I8 values** specifies an array of 8-bit signed integers.
- **FXP values** returns an array of fixed-point numbers.

Examples

Refer to the following VIs for examples of using the DFD Convert Integer to FXP VI:

• Conversion between FXP and Integer VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• How to Build Coefficients Quantizer VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

■ Open example ■ Browse related examples

DFD FXP Code Generator VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Generates fixed-point code from a fixed-point filter, including fixed-point LabVIEW field-programmable gate array (FPGA) code, integer LabVIEW code, and C code. You must <u>manually select the polymorphic instance</u> you want to use.



Note To avoid errors in generating code from a fixed-point filter, ensure that you configure the quantizers according to the guidelines in the <u>Details</u> section of the DFD FXP Set Quantizer VI.

•

Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

Place on the block diagram Find on the Functions palette

DFD FXP LabVIEW FPGA Code Generator



- **open project?** specifies if this VI opens the project file after generating the code. The default is FALSE, which means that you must open the project file manually after this VI generates the code.
- # channels specifies the number of channels that you want the generated code to process. The default is 1.
- **filter in** specifies the input filter.
- destination folder specifies the folder in which you want to save the generated code. This VI returns an error if you do not specify a valid path to the folder.
- **filter name** specifies a name for the filter code that this VI generates. This VI also uses this value as the filename of the project file that contains the generated filter code. You can use only letters and digits in the **filter name** input. This VI ignores other characters. If you specify an invalid name, this VI creates a string that starts with Unknown.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **confirm?** specifies if you want this VI to ask you for confirmation before replacing an existing file. If the value is TRUE, this VI displays a dialog box asking for confirmation to replace the existing file. If the value is FALSE, this VI replaces the existing file automatically. The default is TRUE.
- **Ivproj path** returns the path to the generated project file.
- **sampling frequency/FPGA clock** returns a ratio. You can multiply this ratio with a specific FPGA clock rate to calculate the maximum input sampling frequency per channel that the generated FPGA code can process at the FPGA clock rate. For example, if the ratio is 0.05 and the FPGA clock rate is 40 MHz, then the maximum input sampling frequency per channel that the generated FPGA code can process is 2 MHz.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Integer LabVIEW Code Generator



- **open project?** specifies if this VI opens the project file after generating the code. The default is FALSE, which means that you must open the project file manually after this VI generates the code.
- # channels specifies the number of channels that you want the generated code to process. The default is 1.
- **filter in** specifies the input filter.
- destination folder specifies the folder in which you want to save the generated code. This VI returns an error if you do not specify a valid path to the folder.
- **filter name** specifies a name for the filter code that this VI generates. This VI also uses this value as the filename of the project file that contains the generated filter code. You can use only letters and digits in the **filter name** input. This VI ignores other characters. If you specify an invalid name, this VI creates a string that starts with Unknown.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **confirm?** specifies if you want this VI to ask you for confirmation before replacing an existing file. If the value is TRUE, this VI displays a dialog box asking for confirmation to replace the existing file. If the value is FALSE, this VI replaces the existing file automatically. The default is TRUE.
- **Ivproj path** returns the path to the generated project file.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
DFD FXP C Code Generator



filter in specifies the input filter.

- **destination folder** specifies the folder in which you want to save the generated code. This VI returns an error if you do not specify a valid path to the folder.
- **filter name** specifies a name for the filter code that this VI generates. This VI also uses this value as the filename of the project file that contains the generated filter code. You can use only letters and digits in the **filter name** input. This VI ignores other characters. If you specify an invalid name, this VI creates a string that starts with Unknown.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **confirm?** specifies if you want this VI to ask you for confirmation before replacing an existing file. If the value is TRUE, this VI displays a dialog box asking for confirmation to replace the existing file. If the value is FALSE, this VI replaces the existing file automatically. The default is TRUE.
- **output files** contains the absolute path to the generated C source files. The VI generates three files: *filter name*.c, which contains C source code, *filter name*.h, which contains function prototypes, and nidfdtyp.h, which is a National Instruments proprietary file.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD FXP Code Generator VI:

• LabVIEW FPGA Code Generation VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• Generate LabVIEW FPGA Code for Exponentially Weighted Moving Average Filter VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• Integer LabVIEW Code Generation VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• LabVIEW C Code Generation VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

DFD FXP Coef Report VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Generates a text report about the coefficients of a fixed-point filter.

<u>Examples</u>



■ Place on the block diagram ■ Find on the **Functions** palette

- **filter in** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- filter out returns the filter in unchanged.
- coefficients report returns a string that contains a text report on the fixed-point filter coefficients. The coefficients report output is

a single string that contains the following sections:

- **Reference Value**—Contains the floating-point coefficients before quantization.
- **Quantized Value**—Contains the fixed-point coefficients after quantization.
- **Note**—Indicates whether the quantized coefficients have overflows, underflows, or are zeroes, and provides the number of overflows, underflows, and zeroes that the quantizing generates.

Use the **coefficients report** output to view the number of overflows and underflows so you can adjust the quantization settings.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD FXP Coef Report VI:

• Analyze Coefficients-Quantized Filter VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• Structure Selection and Quantization VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

DFD FXP Get Quantizer VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Retrieves quantizer settings from a fixed-point filter. You must <u>manually</u> <u>select the polymorphic instance</u> you want to use.

▼

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD FXP Get Quantizer (All)



filter in specifies the input filter.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the **filter in** unchanged.
- **quantizers** returns all quantizer settings in the fixed-point filter.
 - **source** returns the quantizer source.
 - wl returns the word length, in number of bits, that the quantizer uses to represent a fixed-point number.
 - **iwl** returns the integer word length, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number.

- **overflow mode** returns the operation mode for overflow and underflow in the quantizer.
- **rounding mode** returns the mode for rounding numbers in the quantizer.
- **signed?** is TRUE if the fixed-point number is a signed number. **signed?** is FALSE if the fixed-point number is an unsigned number.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Get Quantizer (One)



filter in specifies the input filter.

quantizer source specifies the source for the quantizer.

0	Coefficients a/k (default)
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Delay
7	Coefficients b/v

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **filter out** returns the **filter in** unchanged.
- **quantizer** returns the settings of one quantizer.
 - **source** returns the quantizer source.
 - wl returns the word length, in number of bits, that the quantizer uses to represent a fixed-point number.
 - **iwl** returns the integer word length, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number.
 - **overflow mode** returns the operation mode for overflow and underflow in the quantizer.
 - **rounding mode** returns the mode for rounding numbers in the quantizer.
 - **signed?** is TRUE if the fixed-point number is a signed number. **signed?** is FALSE if the fixed-point number is an unsigned number.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Get Quantizer (Group)



filter in specifies the input filter.

quantizer sources specifies the source for each quantizer.



- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **filter out** returns the **filter in** unchanged.
- **quantizers** returns all quantizer settings in the fixed-point filter.
 - **source** returns the quantizer source.
 - wl returns the word length, in number of bits, that the quantizer uses to represent a fixed-point number.
 - **iwl** returns the integer word length, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number.
 - **overflow mode** returns the operation mode for overflow and underflow in the quantizer.
 - **rounding mode** returns the mode for rounding numbers in the quantizer.
 - **signed?** is TRUE if the fixed-point number is a signed number. **signed?** is FALSE if the fixed-point number is an unsigned number.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Easy Fixed-Point Filter Modeling and Simulation VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate directory for an example of using the DFD FXP Get Quantizer VI.

■ Open example ■ Browse related examples

DFD FXP Modeling VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a fixed-point filter model according to the input and output word length settings.



Note To avoid errors in generating code from a fixed-point filter, ensure that you configure the quantizers according to the guidelines in the <u>Details</u> section of the DFD FXP Set Quantizer VI.



■ Place on the block diagram ■ Find on the **Functions** palette

filter in specifies the input filter.

- input word length specifies the word length, in number of bits, that this VI uses to represent the input signal. The valid range is [1, 32]. The default is 16.
- **output word length** specifies the word length, in number of bits, that this VI uses to represent the output signal. The valid range is [1, 32]. The default is 16.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **output rounding mode** specifies the <u>rounding mode</u> this VI uses in the output quantizer.

0 **Nearest**—Rounds to the closest representable number.

1 **Truncation** (default)—Rounds to the closest representable number less than the original value.

filter out returns a fixed-point filter.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Postprocessing VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Converts the output signal of a fixed-point filter from a fixed-point integer representation to a floating-point representation. You must <u>manually</u> <u>select the polymorphic instance</u> you want to use.

•

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD FXP Postprocessing (I32, nCh)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- # channels specifies the number of channels that signal in contains. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **history in** specifies the data from the last iteration of the

postprocessing process.

- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **history out** returns the remaining data for the next iteration of postprocessing. You can wire this output to the **history in** input of the next call to this VI if you want to process the data continuously.

DFD FXP Postprocessing (I16, nCh)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- # channels specifies the number of channels that signal in contains. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **history in** specifies the history data from the last iteration of

postprocessing.

- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **history out** returns the remaining data for the next iteration of postprocessing. You can wire this output to the **history in** input of the next call to this VI if you want to process the data continuously.

DFD FXP Postprocessing (I32, 1Ch)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Postprocessing (I16, 1Ch)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Simulation with Integer Inputs VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate directory for an example of using the DFD FXP Postprocessing VI.

■ Open example ■ Browse related examples

DFD FXP Quantize Coef VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Quantizes the coefficients of a floating-point filter and generates a fixedpoint filter. You must <u>manually select the polymorphic instance</u> you want to use.

Use the <u>DFD FXP Coef Report</u> VI to generate a report on the effect that quantization has on the filter.

•

Details Examples

Use the pull-down menu to select an instance of this VI.

■ Place on the block diagram ■ Find on the **Functions** palette

DFD FXP Quantize Coef (Easy)



- **coefficients b/v word length** specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent coefficients *b/v*.
- **coefficients a/k word length** specifies the word length, in number of bits, that the quantizer uses to represent coefficients *a/k*.
- **filter in** specifies the input floating-point filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **gain settings** specifies the settings for the filter gain.
 - **gain processing** specifies whether you want to process the filter gain on a host machine or a fixed-point target, such as an NI Reconfigurable I/O (RIO) target. If you want to

generate C code from the resulting fixed-point filter, you must set **gain processing** to On Target.

- 0 **On Target**—Specifies to process the filter gain on a fixed-point target.
- 1 **On Host** (default)—Specifies to process the filter gain on a host machine.
- **gain word length** specifies the word length, in number of bits, that this VI uses to represent the filter gain if you set **gain processing** to On Target. The valid range is [1, 32]. The default is 16.
- **filter out** returns a fixed-point filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Quantize Coef (Advanced)



coefficients b/v quantizer specifies the settings for the coefficients *b/v* quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Delay
7	Coefficients b/v (default)

- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The valid range is [1, 32]. If you specify an invalid value, this VI uses the same quantizer settings as you specified in the **Coefficients a/k quantizer**. The default is –1.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.
- **overflow mode** specifies how this VI <u>handles overflows and</u> <u>underflows</u> in the quantizer.

0 Saturation

1 Wrap (default)

rounding mode specifies the <u>rounding mode</u> this VI uses in

the quantizer.

0 Nearest

1 **Truncation** (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **coefficients a/k quantizer** specifies the settings for the coefficients *a/k* quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Delay
7	Coefficients b/v

- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The valid range is [1, 32]. If you specify an invalid value, this VI uses the same quantizer settings as you specified in the **Coefficients b/v quantizer**. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.
- overflow mode specifies how this VI handles overflows and underflows in the quantizer.



1 Wrap (default)

rounding mode specifies the <u>rounding mode</u> this VI uses in the quantizer.

0	Nearest
1	Truncation (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- **filter in** specifies the input floating-point filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **gain settings** specifies the settings for the filter gain.
 - **gain processing** specifies whether you want to process the filter gain on a host machine or a fixed-point target, such as

an NI Reconfigurable I/O (RIO) target. If you want to generate C code from the resulting fixed-point filter, you must set **gain processing** to On Target.

- 0 **On Target**—Specifies to process the filter gain on a fixed-point target.
- 1 **On Host** (default)—Specifies to process the filter gain on a host machine.
- **gain word length** specifies the word length, in number of bits, that this VI uses to represent the filter gain if you set **gain processing** to On Target. The valid range is [1, 32]. The default is 16.
- **filter out** returns a fixed-point filter.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Quantize Coef Details

You can represent an integer, or binary number, with a length **wl** as $b_1b_2b_3b_{wl}$ where b_i is a binary digit. Both **wl** and **iwl** determine the position of the binary point and the range the binary number can represent.

For a signed fixed-point number in two's complement, the value of the fixed-point number is defined by the following equation:

$$\hat{x} = Q[x] = 2^{iwl} \left(-b_1 \cdot 2^{-1} + \sum_{i=2}^{wl} b_i \cdot 2^{-i} \right)$$

The highest representable number is $2^{iwl}(2^{-1}-2^{-wl})$ and the lowest representable number is -2^{iwl-1} .

For an unsigned fixed-point number, the value of the fixed-point number is defined by the following equation:

$$\hat{x} = Q[x] = 2^{iwl} \sum_{i=1}^{wl} b_i \cdot 2^{-i}$$

For unsigned fixed-point numbers, the highest representable number is $2^{iwl}(1-2^{-wl})$ and the lowest representable number is 0.

The **rounding mode** input controls the operation of quantizing to **wl** bits. If you set **rounding mode** to Nearest, the quantizer rounds the result to the closest representable number. If the two closest representable numbers are equidistant, the quantizer rounds the result to the closest representable number with a least significant bit of 0. If you set **rounding mode** to Truncation, the quantizer rounds to the closest representable number lower than the original value.

overflow mode determines the quantized value when an overflow or underflow occurs. If you set **overflow mode** to Saturation, the quantizer converts the specified value to the highest representable number for overflow or to the lowest representable number for underflow. If you set **overflow mode** to Wrap, the quantizer wraps around the specified value from the highest representable number to the lowest representable number for overflow or from the lowest representable number to the highest representable number for underflow. The size of the error does not increase as abruptly with **Saturation** as the size does with **Wrap** when overflow or underflow occurs.

Examples

Refer to the following VIs for examples of using the DFD FXP Quantize Coef VI:

• How to Build Coefficients Quantizer VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

Open example Browse related examples

• Analyze Coefficients-Quantized Filter VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

Open example Browse related examples

• Easy Fixed-Point Filter Modeling and Simulation VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• Structure Selection and Quantization VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

DFD FXP Set Quantizer VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Sets a quantizer or quantizers in a fixed-point filter. You must <u>manually</u> <u>select the polymorphic instance</u> you want to use.

•

Details Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette
DFD FXP Set Quantizer (One)



filter in specifies the input filter.

quantizer specifies the settings of the quantizer.

source specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Dolov

6 Delay

7 Coefficients b/v

- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- **iwl** specifies the <u>integer word length</u>, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number. The default is 1. **iwl** can be any integer value.
- **overflow mode** specifies how this VI <u>handles overflows and</u> <u>underflows</u> in the quantizer.

0	Saturation	
1	Wrap (default)	

rounding mode specifies the <u>rounding mode</u> this VI uses in the quantizer.

0	Nearest

1 **Truncation** (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns a fixed-point filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code**

is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

source describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Set Quantizer (Group)



filter in specifies the input filter.

- **quantizers** specifies the settings for a group of quantizers.
 - **quantizer** specifies the settings of the quantizer.
 - **source** specifies the <u>quantizer</u> source.

0	Coefficients a/k (default)
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Delay
7	Coefficients b/v

- wl specifies the <u>word length</u>, in number of bits, that the quantizer uses to represent a fixed-point number. The default is 16.
- iwl specifies the integer word length, in number of bits, within wl that the quantizer uses to represent the integer part of a fixed-point number. The default is 1.
 iwl can be any integer value.
- overflow mode specifies how this VI <u>handles</u> <u>overflows and underflows</u> in the quantizer.

0	Saturation
1	Wrap (default)

rounding mode specifies the <u>rounding mode</u> this VI uses in the quantizer.

0 Nearest

1 **Truncation** (default)

- **signed?** specifies if the fixed-point number is a signed number. This VI supports signed numbers only. If you remove the checkmark from the **signed?** checkbox, the result you obtain might not be correct.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns a fixed-point filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Set Quantizer Details

Depending on whether you use the fixed-point filter model for simulation or for code generation, the DFD FXP Set Quantizer VI has different restrictions on the word lengths, the overflow mode, and the rounding mode of the quantizers. The following table lists the restrictions on each quantizer for simulation.

Quantizer	Word Length	Rounding Mode	Overflow Mode
Coefficients alk	132	No Restriction	No Restriction
Coefficients b/v	132	No Restriction	No Restriction
Input	132	No Restriction	No Restriction
Output	132	No Restriction	No Restriction
Multiplicand	132	No Restriction	No Restriction
Product	164	No Restriction	No Restriction
Sum	164	No Restriction	No Restriction
Delay	132	No Restriction	No Restriction
Gain	132	Nearest	Saturation

The following table lists the restrictions on each quantizer for code generation.

Quantizer	Word Length	Rounding Mode	Overflow Mode
Coefficients alk	132	No Restriction	No Restriction
Coefficients b/v	132	No Restriction	No Restriction
Input	132	No Restriction	No Restriction
Output ¹	132	Truncation	Wrap
Multiplicand ²	132	Truncation	Wrap
Product	32	Truncation	Wrap
Sum	32	Truncation	Wrap
Delay ³	Equal to Input/Sum	Truncation	Wrap
Gain	132	Nearest	Saturation

¹For LabVIEW code generation, such as integer LabVIEW code and LabVIEW FPGA code, the rounding and overflow modes of the output

quantizer do not have any restrictions. However, if you want to generate C code, the rounding mode must be Truncation and the overflow mode must be Wrap.

²For <u>FIR structures</u>, the word length of the multiplicand quantizer must conform to the restrictions as the following table shows:

Filter Structure	Word Length of Multiplicand
FIR Direct Form	Equal to Input
FIR Direct Form Transposed	Equal to Input
FIR Symmetric	Equal to min(Input+1, 32)
FIR Antisymmetric	Equal to min(Input+1, 32)

The integer word length of the multiplicand quantizer must be greater than or equal to the value in the default fixed-point model.

³For FIR structures other than the FIR Direct Form Transposed structure, the word length of the delay quantizer must be equal to that of the input quantizer. For the FIR Direct Form Transposed structure and all IIR and lattice filter structures, the word length of the delay quantizer must be equal to that of the sum quantizer.

Example

Refer to the Customized Fixed-Point Filter Modeling and Simulation VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate directory for an example of using the DFD FXP Set Quantizer VI.

■ Open example ■ Browse related examples

DFD FXP Simulation VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Simulates the filtering process continuously and generates a filtering statistics report for a fixed-point filter. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

You can use the <u>DFD FXP Simulation Report</u> VI to generate a report about the performance of quantizers during a simulation.

Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

▼

DFD FXP Simulation (DBL In)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- init? specifies how you want to initialize the internal states. The default is TRUE, which specifies that this VI initializes the internal states to zero. If init? is FALSE, this VI initializes the internal states from the final states of the previous call to the current VI instance. To process a large data sequence, split the sequence into smaller blocks, set init? to TRUE for the first block, and set init? to FALSE for the remaining blocks.
- signal in is the input signal you want to process. You can wire an impulse pattern, step pattern, uniform white noise, or a user-defined signal to this input. The input word length value you set on the DVD FXP Modeling VI determines the range of signal in. The range equals [-2^(input word length-1), 2^(input word length-1)-1]. For example, if you specify 16 as the input word length value, the corresponding range is [-32768, 32767].
- **filter in** specifies the input filter.
- **filtering statistics in** specifies the statistical information of quantizers in the **filter in** input before the simulation.
 - **quantizer source** specifies the source for the quantizer.

0	Coefficients a/k (default)
1	Input
2	Output
3	Multiplicand
4	Product

5	Sum
6	Delay
7	Coefficients b/v

statistical information contains the statistical information for **quantizer source**.

- **max value** contains the maximum value that appeared in **quantizer source** during simulation.
- **min value** contains the minimum value that appeared in **quantizer source** during simulation.
- **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
- #underflows contains the number of underflows that occurred in quantizer source during simulation.
- **#operations** contains the number of operations that occurred in **quantizer source** during simulation.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **DBL signal out** returns the filtered floating-point signal.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- **filtering statistics report** returns the statistical information of quantizers in the **filter in** input after the simulation occurs. Use the DFD FXP Simulation Report VI to generate a text report from the **filtering statistics report** output.
 - **quantizer source** returns the quantizer source.
 - **statistical information** contains the statistical information for **quantizer source**.
 - **max value** contains the maximum value that appeared in **quantizer source** during simulation.
 - **min value** contains the minimum value that appeared in **quantizer source** during simulation.
 - **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
 - **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
 - **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning.

DFD FXP Simulation (I32 In)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- init? specifies how you want to initialize the internal states. The default is TRUE, which specifies that this VI initializes the internal states to zero. If init? is FALSE, this VI initializes the internal states from the final states of the previous call to the current VI instance. To process a large data sequence, split the sequence into smaller blocks, set init? to TRUE for the first block, and set init? to FALSE for the remaining blocks.
- **signal in** specifies the input signal that you want to process.
- **filter in** specifies the input filter.
- **filtering statistics in** specifies the statistical information of quantizers in the **filter in** input before the simulation.
 - **quantizer source** specifies the source for the quantizer.

0	Coefficients a/k (default)
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Delay
7	Coefficients b/v

statistical information contains the statistical information for **quantizer source**.

- **max value** contains the maximum value that appeared in **quantizer source** during simulation.
- **min value** contains the minimum value that appeared in **quantizer source** during simulation.
- **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
- #underflows contains the number of underflows that occurred in quantizer source during simulation.
- **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DBL signal out** returns the filtered floating-point signal.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- **filtering statistics report** returns the statistical information of

quantizers in the **filter in** input after the simulation occurs. Use the DFD FXP Simulation Report VI to generate a text report from the **filtering statistics report** output.

- **quantizer source** returns the quantizer source.
- **statistical information** contains the statistical information for **quantizer source**.
 - **max value** contains the maximum value that appeared in **quantizer source** during simulation.
 - **min value** contains the minimum value that appeared in **quantizer source** during simulation.
 - **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
 - **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
 - **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Simulation (I16 In)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **III init?** specifies how you want to initialize the internal states. The default is TRUE, which specifies that this VI initializes the internal states to zero. If **init?** is FALSE, this VI initializes the internal states from the final states of the previous call to the current VI instance. To process a large data sequence, split the sequence into smaller blocks, set **init?** to TRUE for the first block, and set **init?** to FALSE for the remaining blocks.
- **signal in** specifies the input signal that you want to process.
- **filter in** specifies the input filter.
- **filtering statistics in** specifies the statistical information of quantizers in the **filter in** input before the simulation.
 - **quantizer source** specifies the source for the quantizer.

0	Coefficients a/k (default)
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Delay
7	Coefficients b/v

statistical information contains the statistical information for **quantizer source**.

- **max value** contains the maximum value that appeared in **quantizer source** during simulation.
- **min value** contains the minimum value that appeared in **quantizer source** during simulation.
- **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
- #underflows contains the number of underflows that occurred in quantizer source during simulation.
- **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DBL signal out** returns the filtered floating-point signal.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- **filtering statistics report** returns the statistical information of

quantizers in the **filter in** input after the simulation occurs. Use the DFD FXP Simulation Report VI to generate a text report from the **filtering statistics report** output.

- **quantizer source** returns the quantizer source.
- **statistical information** contains the statistical information for **quantizer source**.
 - **max value** contains the maximum value that appeared in **quantizer source** during simulation.
 - **min value** contains the minimum value that appeared in **quantizer source** during simulation.
 - **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
 - **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
 - **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD FXP Simulation VI:

• Easy Fixed-Point Filter Modeling and Simulation VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

Open example Browse related examples

- Customized Fixed-Point Filter Modeling and Simulation VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate
 - Open example Browse related examples
- Continuous Overflow Statistics VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• Simulation with Integer Inputs VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

Open example Browse related examples

DFD FXP Simulation Report VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a text report of filtering statistics from the <u>DFD FXP Simulation</u> VI or the <u>DFD FXP Simulation with State</u> VI.

Wire the **filtering statistics report** output of the DFD FXP Simulation VI or the DFD FXP Simulation with State VI to the **filtering statistics report** input of this VI.

Details Examples



■ Place on the block diagram ■ Find on the **Functions** palette

filtering statistics report specifies the statistical information of quantizers after a simulation.

quantizer source specifies the source for the quantizer.

0	Coefficients a/k (default)
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Delay
7	Coefficients b/v

- **statistical information** contains the statistical information for **quantizer source**.
 - **max value** contains the maximum value that appeared in **quantizer source** during simulation.
 - **min value** contains the minimum value that appeared in **quantizer source** during simulation.

132

#overflows contains the number of overflows that occurred in **quantizer source** during simulation.

- **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
- **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filtering text report** returns a text report with the filtering statistics from **filtering statistics report**.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE

(checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Simulation Report Details

The filtering text report output lists filtering statistics for the quantizers of Input, Output, Multiplicand, Product, Sum, and Delay during the simulation process. The output does not include Coefficients a/k and Coefficients b/v. Each quantizer includes five entries: max value, min value, #overflows, #underflows, and #operations.

You can use **filtering text report** to monitor the performance of quantizers. Use the **max value** and **min value** to monitor the effect of overflow and underflow and adjust quantization settings accordingly.

Examples

Refer to the following VIs for examples of using the DFD FXP Simulation Report VI:

• Customized Fixed-Point Filter Modeling and Simulation VI: labview/examples/Digital Filter Design/Fixed-Point Filters/Single-Rate

■ Open example ■ Browse related examples

• Continuous Overflow Statistics VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

Open example Browse related examples

DFD FXP Simulation with State VI

Owning Palette: <u>Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Simulates the filtering process with initial internal states and generates the filtering statistics report for a fixed-point filter. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

You can use the <u>DFD FXP Simulation Report</u> VI to generate a text report about the performance of quantizers during a simulation.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

DFD FXP Simulation with State (DBL In)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [-10, 10V]. The default is 1.
- signal in is the input signal you want to process. You can wire an impulse pattern, step pattern, uniform white noise, or a user-defined signal to this input. The input word length value you set on the DVD FXP Modeling VI determines the range of signal in. The range equals [-2^(input word length-1), 2^(input word length-1)-1]. For example, if you specify 16 as the input word length value, the corresponding range is [-32768, 32767].
- **filter in** specifies the input filter.
- **filtering statistics in** specifies the statistical information of quantizers in the **filter in** input before the simulation.
 - **quantizer source** specifies the source for the quantizer.

1 Input

2 Output

3 Multiplicand

- 4 Product
- 5 **Sum**
- 6 Delay
- 7 **Coefficients b/v**
- **statistical information** contains the statistical information for **quantizer source**.

max value contains the maximum value that

appeared in quantizer source during simulation.

- **min value** contains the minimum value that appeared in **quantizer source** during simulation.
- **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
- **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
- **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **DBL signal out** returns the filtered floating-point signal.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.

[263]

filtering statistics report returns the statistical information of quantizers in the filter in input after the simulation occurs. Use the DFD FXP Simulation Report VI to generate a text report from the filtering statistics report output.

- **quantizer source** returns the quantizer source.
- **statistical information** contains the statistical information for **quantizer source**.
 - **max value** contains the maximum value that appeared in **quantizer source** during simulation.
 - **min value** contains the minimum value that appeared in **quantizer source** during simulation.
 - **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
 - **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
 - **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD FXP Simulation with State (I32 In)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [-10, 10V]. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **filter in** specifies the input filter.
- **filtering statistics in** specifies the statistical information of quantizers in the **filter in** input before the simulation.
 - **quantizer source** specifies the source for the quantizer.

0	Coefficients a/k (default)
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Delay
7	Coefficients b/v

- **statistical information** contains the statistical information for **quantizer source**.
 - **max value** contains the maximum value that appeared in **quantizer source** during simulation.
 - **min value** contains the minimum value that appeared in **quantizer source** during simulation.
 - **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.

- **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
- **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **DBL signal out** returns the filtered floating-point signal.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- **filtering statistics report** returns the statistical information of quantizers in the **filter in** input after the simulation occurs. Use the DFD FXP Simulation Report VI to generate a text report from the **filtering statistics report** output.
 - **quantizer source** returns the quantizer source.
 - **statistical information** contains the statistical information

for quantizer source.

- **max value** contains the maximum value that appeared in **quantizer source** during simulation.
- **min value** contains the minimum value that appeared in **quantizer source** during simulation.
- **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
- **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
- **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD FXP Simulation with State (I16 In)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [-10, 10V]. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **filter in** specifies the input filter.
- **filtering statistics in** specifies the statistical information of quantizers in the **filter in** input before the simulation.
 - **quantizer source** specifies the source for the quantizer.

0	Coofficiento alle (defeuilt)
U	Coefficients ark (default)
1	Input
2	Output
3	Multiplicand
4	Product
5	Sum
6	Delay
7	Coefficients b/v

- **statistical information** contains the statistical information for **quantizer source**.
 - **max value** contains the maximum value that appeared in **quantizer source** during simulation.
 - **min value** contains the minimum value that appeared in **quantizer source** during simulation.
 - **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.

- **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
- **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **DBL signal out** returns the filtered floating-point signal.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- **filtering statistics report** returns the statistical information of quantizers in the **filter in** input after the simulation occurs. Use the DFD FXP Simulation Report VI to generate a text report from the **filtering statistics report** output.
 - **quantizer source** returns the quantizer source.
 - **statistical information** contains the statistical information

for quantizer source.

- **max value** contains the maximum value that appeared in **quantizer source** during simulation.
- **min value** contains the minimum value that appeared in **quantizer source** during simulation.
- **#overflows** contains the number of overflows that occurred in **quantizer source** during simulation.
- **#underflows** contains the number of underflows that occurred in **quantizer source** during simulation.
- **#operations** contains the number of operations that occurred in **quantizer source** during simulation.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.
Multirate Filter Analysis VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Multirate Filter Analysis VIs to analyze characteristics of multirate filters.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
DFD Plot MRate Freq Response	Plots the frequency responses, including the magnitude and phase responses, of multirate filters.
<u>DFD Plot NStage</u> <u>MRate Freq</u> <u>Response</u>	Plots the frequency responses, including the magnitude and phase responses, of multistage multirate filters.

DFD Plot MRate Freq Response VI

Owning Palette: Multirate Filter Analysis VIs

Installed With: Digital Filter Design Toolkit

Plots the frequency responses, including the magnitude and phase responses, of multirate filters.

Example

132



■ Place on the block diagram ■ Find on the **Functions** palette

output option specifies the filter response that this VI plots.

- 0 **Auto** (default)—This VI automatically determines the responses of multirate filter in to plot. If multirate filter in is a floatingpoint multirate filter, this VI plots the frequency response of this multirate filter. If multirate filter in is a fixed-point multirate filter, this VI plots the frequency responses of both this multirate filter and the reference floating-point multirate filter. 1 **Floating-Point Only**—If **multirate filter in** is a floating-point multirate filter, this VI plots the frequency response of this multirate filter. If **multirate filter in** is a fixed-point multirate filter, this VI plots the frequency response of the reference floating-point multirate filter. 2 **Fixed-Point Only**—If **multirate filter in** is a floating-point multirate filter, this VI returns an empty graph. If multirate filter in is a fixed-point multirate filter, this VI plots the frequency response of this multirate filter. # freq bins specifies the number of frequency bins between 0 and the highest sampling frequency in the frequency response of the multirate filter. The default is -1, which specifies that this VI
- automatically determines the number of frequency bins. **multirate filter in** specifies the input multirate filter.

- **phase view** specifies the phase response display settings.
 - **unwrap?** specifies whether this VI unwraps the phase. The default is FALSE, which specifies that the phase remains wrapped and is limited to $[0, 2\pi)$.
 - **degree?** specifies whether the phase appears in degrees or radians. The default is FALSE, which specifies that the phase appears in radians.
- **dB on?** specifies whether this VI uses decibels or a linear scale to express the magnitude response. The default is TRUE, which specifies that this VI converts linear magnitude response to decibels.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs in** specifies the input sampling frequency of the multirate filter in hertz. If the value of **fs in** is equal to or less than zero, this VI uses the sampling frequency of the input filter. The default is –1.

- multirate filter out returns the multirate filter in unchanged.
- magnitude response returns the magnitude response of the filter.
- **phase response** returns the phase response of the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Analyze Coefficients-Quantized Multirate Filter VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate directory for an example of using the DFD Plot MRate Freq Response VI.

■ Open example ■ Browse related examples

DFD Plot NStage MRate Freq Response VI

Owning Palette: <u>Multirate Filter Analysis VIs</u>

Installed With: Digital Filter Design Toolkit

Plots the frequency responses, including the magnitude and phase responses, of multistage multirate filters.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

output option specifies the filter responses that this VI plots.

- Auto (default)—This VI automatically determines the responses of multirate filters in to plot. If multirate filters in are floating-point multirate filters, this VI plots the frequency responses of these multirate filters. If multirate filters in are fixed-point multirate filters, this VI plots the frequency responses of both these multirate filters and the reference floating-point multirate filters.
 Floating-Point Only—If multirate filters in are floating-point multirate filters, this VI plots the frequency responses of these multirate filters. If multirate filters in are floating-point multirate filters.
 Floating-Point Only—If multirate filters in are floating-point multirate filters, this VI plots the frequency responses of these multirate filters. If multirate filters in are fixed-point multirate filters, this VI plots the frequency responses of only the reference floating-point multirate filters.
 Fixed-Point Only—If multirate filters in are floating-point multirate filters, this VI plots the frequency responses of only the reference floating-point multirate filters.
- **freq bins** specifies the number of frequency bins between 0 and the highest sampling frequency in the frequency response of the multirate filter. The default is –1, which specifies that this VI automatically determines the number of frequency bins.

frequency responses of these multirate filters.

- **multirate filters in** specifies the input multistage multirate filters.
- **phase view** specifies the phase response display settings.
 - **unwrap?** specifies whether this VI unwraps the phase. The default is FALSE, which specifies that the phase remains wrapped and is limited to $[0, 2\pi)$.
 - **degree?** specifies whether the phase appears in degrees or radians. The default is FALSE, which specifies that the phase appears in radians.
- **dB on?** specifies whether this VI uses decibels or a linear scale to express the magnitude response. The default is TRUE, which specifies that this VI converts linear magnitude response to decibels.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs in** specifies the input sampling frequency of the multirate filter in hertz. If the value of **fs in** is equal to or less than zero, this VI uses the sampling frequency of the input filter. The default is –1.

multirate filters out returns the **multirate filters in** unchanged.

- **magnitude responses** returns the overall magnitude response and the magnitude response of each stage of the **multirate filters in**. For fixed-point multistage multirate filters, this VI returns the magnitude responses of all stages by interleaving the fixed-point magnitude responses and reference floating-point magnitude responses in the following format: [*MR*_{overall}, *MF*_{overall}, *MR*₁, *MF*₁, *MR*₂, *MF*₂, ..., *MR*_n, *MF*_n]
 - where *MR*_{overall} represents the overall magnitude response of the reference floating-point multistage multirate filter

*MF*_{overall} represents the overall magnitude response of the fixed-point multistage multirate filter

 MR_n represents the magnitude response of the n^{th} stage of the reference floating-point multistage multirate filter

 MF_n represents the magnitude response of the n^{th} stage of the fixed-point multistage multirate filter

phase responses returns the overall phase response and the phase response of each stage of the multirate filters in **multirate filters in**. For fixed-point multistage multirate filters, this VI returns the phase responses of all stages by interleaving the fixed-point phase responses and reference floating-point phase responses in the following format: $[PR_{overall}, PF_{overall}, PR_1, PF_1, PR_2, PF_2, ..., PR_n, PF_n]$

where *PR*_{overall} represents the overall phase response of the reference floating-point multistage multirate filter

*PF*_{overall} represents the overall phase response of the fixed-point multistage multirate filter

 PR_n represents the phase response of the n^{th} stage of the reference floating-point multistage multirate filter

 PF_n represents the phase response of the n^{th} stage of the fixed-point multistage multirate filter

error out contains error information. If error in indicates that an

error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Analyze Coefficients-Quantized Multistage Multirate Filter VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate directory for an example of using the DFD Plot NStage MRate Freq Response VI.

Open example Browse related examples

Multirate Filter Design VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Multirate Filter Design VIs to create multirate filters.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
<u>DFD Halfband</u> <u>Design</u>	Creates a <u>halfband filter</u> with an automatic order estimation.
<u>DFD MRate</u> <u>Filter Design</u>	Creates a single-stage multirate filter. You must manually select the polymorphic instance you want to use.
<u>DFD NStage</u> <u>MRate Filter</u> <u>Design</u>	Creates multistage multirate filters, which meet the requirements you specify by cascading filters in multirate filters out .
<u>DFD Nyquist</u> <u>Design</u>	Creates a <u>Nyquist filter</u> using the window or equi-ripple methods with an automatic order estimation.
<u>DFD Raised</u> Cosine Design	Creates a <u>raised cosine or a root-raised cosine</u> finite impulse response (FIR) filter.
<u>Multirate CIC</u> Design	Creates a <u>cascaded integrator comb (CIC) filter</u> .
<u>Multirate FIR</u> Design	Creates a finite impulse response (FIR) multirate filter.
<u>Multistage</u> <u>Multirate Filter</u> <u>Design</u>	Creates a multistage multirate filter.

Multirate CIC Design Express VI

Owning Palette: <u>Multirate Filter Design VIs</u> Installed With: Digital Filter Design Toolkit Creates a <u>cascaded integrator comb (CIC) filter</u>. <u>Dialog Box Options</u>

Block Diagram Outputs

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

Dialog Box Options

Parameter	Description
Magnitude Response	Contains the plot of the magnitude response. You can drag the cursors in the plot to change the specifications.
	 Magnitude in dB—Specifies whether this Express VI uses decibels or a linear scale to express the magnitude response. If you place a checkmark in the checkbox, this Express VI converts a linear magnitude response to decibels. This Express VI uses decibels by default. Passband—Specifies the color of the lines in the magnitude plot that represent the passband response and the passband edge frequency. The default is blue
	 Maximum aliasing frequency—Specifies the color of the lines in the magnitude plot that represent the aliasing distortion and the maximum aliasing frequency. The default is red. This option is available only if you set Filter type to Decimation or No Rate Change. Maximum imaging frequency—Specifies the color of the lines in the magnitude plot that represent the images and the maximum image frequency. The default is available only if you set Filter type to Interpolation.
Main	Contains the following options:
Settings	 Filter type—Specifies the type of CIC filter that this VI creates. The valid values include No Rate Change, Interpolation, and Decimation. The default is Decimation. Number of stages—Specifies the number of stages in the CIC filter. The valid range is [1, 8]. The default is 4. Filter factor—Specifies the sampling frequency conversion factor of the CIC filter. If you set Filter
	type to No Rate Change, the value of this input must

	 be an integer greater than or equal to 1. If you set Filter type to Interpolation or Decimation, the value of this input must be an integer greater than or equal to 2. The default is 2. Differential delay—Specifies the number of null values in the frequency response. Increasing the value of the differential delay increases the number of null values and the sharpness of the transition band in the frequency response. A larger differential delay value also requires more hardware resources. The valid values include 1 and 2. The default is 1. Input sampling frequency—Specifies the sampling frequency, in hertz, of an input signal for the CIC filter. This input must contain a value greater than zero. The default is 1 Hz.
Analysis Settings	 Contains the following options: Analyze CIC filter—Specifies if you want to analyze the passband distortion and aliasing distortion for the CIC filter. Passband frequency—Specifies the passband frequency of the CIC filter. This input is available only if you place a checkmark in the Analyze CIC filter checkbox. This Express VI calculates the default value as 80% of <i>f</i>_N/4. If you set Filter type to Interpolation or No Rate Change, <i>f</i>_N is the Nyquist frequency of the input sampling frequency. If you set Filter type to Decimation, <i>f</i>_N is the Nyquist frequency of the output sampling frequency in a unit that the Magnitude in dB option determines. This output displays the magnitude distortion value only if you place a checkmark in the Analyze CIC filter checkbox. Aliasing distortion—Returns the magnitude distortion at the magnitude distortion value only if you place a checkmark in the Analyze CIC filter checkbox. Aliasing distortion—Returns the magnitude distortion at the magnitude distortion value only if you place a checkmark in the Analyze CIC filter checkbox. Aliasing distortion—Returns the magnitude distortion at the maximum aliasing frequency in a unit that the Magnitude in dB option determines.

This output displays the magnitude distortion value only if you place a checkmark in the Analyze CIC filter checkbox.
--

Block Diagram Outputs

Parameter	Description
CIC filter out	Returns the new CIC multirate filter.
error out	Describes the error status that this VI or function produces.

Multirate FIR Design Express VI

Owning Palette: <u>Multirate Filter Design VIs</u> Installed With: Digital Filter Design Toolkit Creates a finite impulse response (FIR) multirate filter. <u>Details</u> <u>Dialog Box Options</u>

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

Parameter	Description
Magnitude Response	Contains the plot of the magnitude response. You can drag the cursors in the plot to change the specifications. The color you specify in Passband represents the passband response and the passband edge frequency. The color you specify in Stopband represents the stopband attenuation and the stopband edge frequency. The color you specify in Stopband limit represents the stopband limit. The color you specify in Nyquist represents half the sampling frequency, also known as the Nyquist frequency.
	Magnitude in dB—Specifies whether this
	Express VI uses decibels or a linear scale to express the magnitude response. If you place a checkmark in the checkbox, this Express VI converts a linear magnitude response to decibels. This Express VI uses decibels by default.
	 Nyquist—Specifies the color of the line in the magnitude plot that represents the Nyquist frequency. The default is green.
	 Passband—Specifies the color of the lines in the magnitude plot that represent the passband response and the passband edge frequency. The default is blue.
	• Stopband limit —Specifies the color of the line in the magnitude plot that represents the maximum stopband edge frequency. The default is yellow.
	• Stopband —Specifies the color of the lines in the magnitude plot that represent the stopband attenuation and the stopband edge frequency. The default is red.
Flow Diagram	Returns the flow diagram of the multirate filter you specify in Filter type .

	This input is available only if you set
	decimation or rational resampling filter.
	 L—Specifies the interpolation factor for an <u>interpolation</u> or <u>rational resampling</u> filter. This input is available only if you set Filter type to Interpolation or Rational Resampling. The value of this input must be an integer greater than or equal to 2. The default is 1.
	• Filter factor—Specifies the sampling frequency conversion factor of the multirate filter. Contains the following options:
	• Filter type—Specifies the type of multifate litter that this VI creates. The valid values include No Rate Change , Interpolation , Decimation , and Rational Resampling . The default is Decimation.
	 Design method—Specifies the method that this Express VI uses to design the multirate filter. The valid values include Kaiser Window, Dolph-Chebyshev Window, and Equi-Ripple FIR. The default is Equi-Ripple FIR.
Main Settings	Contains the following options:
Coefficients	 to the settings you specify on the Floating-Point Design page. Filter order—Displays the order of the multirate filter you design. The value of Filter order+1 equals the number of coefficients.

	 sampling frequency, in hertz, of an input signal for the multirate filter. This input must contain a value greater than zero. The default is 1k Hz. Passband edge frequency—Specifies the passband edge frequency of the multirate filter. The default is 200 Hz. Stopband edge frequency—Specifies the stopband edge frequency of the multirate filter. The default is 250 Hz. Output sampling frequency—Returns the sampling frequency of the output signal for the multirate filter. Passband ripple—Specifies the passband ripple of the multirate filter in a unit that the Magnitude in dB option determines. If you place a checkmark in the Magnitude in dB checkbox, Passband ripple must be greater than zero. The default is 0.11. Stopband attenuation—Specifies the stopband attenuation of the multirate filter in a unit that the Magnitude in dB option determines. If you place a checkmark from the Magnitude in dB checkbox, the valid range of Passband ripple is (0, 1). The default then is 0.011. Stopband attenuation—Specifies the stopband attenuation of the multirate filter in a unit that the Magnitude in dB checkbox, Stopband attenuation must be greater than zero. The default is 60 dB. If you remove the checkmark from the Magnitude in dB checkbox, the valid range of Stopband attenuation is (0, 1). The default is 60 dB. If you remove the checkmark from the Magnitude in dB checkbox, the valid range of Stopband attenuation is (0, 1). The default then is 0.011.
Transition band aliasing allowed	Specifies if you allow frequency aliasing in the transition band.
Quantization Settings	 Contains the following options: Quantize filter—Specifies if you want to use this Express VI to quantize the floating-point multirate filter you design. If you enter the

multirate filter specifications, you can place a checkmark in the **Quantize filter** checkbox after you click the **Update Design** button.

- **Coefficients word length**—Specifies the word <u>length</u>, in number of bits, that this Express VI uses to represent the filter coefficients. This option is available only if you place a checkmark in the **Quantize filter** checkbox. The valid range is [1, 32]. The default is 16.
- Gain word length—Specifies the word length, in number of bits, that this Express VI uses to represent the multirate filter gain if the gain processing occurs on an NI Reconfigurable I/O (RIO) target. This option is available only if you place a checkmark in the Quantize filter checkbox and select On Target from the Gain processing pull-down menu. The valid range is [1, 32]. The default is 16.
- Coefficients scale type—Specifies how this Express VI scales the multirate filter coefficients. This option is available only if you place a checkmark in the Quantize filter checkbox. The valid values include No Norm, Time Domain-1 Norm, Time Domain-2 Norm, and Time Domain-Inf Norm. The default is Time Domain-1 Norm.
- Scale by power of 2—Specifies whether this Express VI scales the multirate filter with the original norm value or the smallest power of 2 value that is greater than the norm value. This option is available only if you place a checkmark in the Quantize filter checkbox. If you place a checkmark in the Scale by power of 2 checkbox, this Express VI scales the multirate filter with the smallest power of 2 value. If you remove the checkmark from the Scale by power of 2 checkbox, this Express VI scales the multirate filter with the original norm value.

	 Gain processing—Specifies whether the gain processing occurs on a host machine or an NI-RIO target. This option is available only if you place a checkmark in the Quantize filter checkbox. Postprocessing filter gain—Displays the value of the postprocessing gain. Manually multiply
	the fixed-point output signal by the postprocessing gain to obtain the floating-point output signal or automatically use the <u>DFD FXP</u> <u>MRate Postprocessing</u> VI to handle the postprocessing gain.
Tips	Displays tips and error messages that help you use this Express VI to design a multirate filter.
Update Design	Updates the floating-point multirate filter design with the specifications you entered. When you click the Update Design button, you enable the Quantize filter option on the Fixed-Point Quantization page.

Block Diagram Outputs

Parameter	Description
error out	Describes the error status that this VI or function produces.
multirate filter out	Returns the new multirate filter.
postprocessing filter gain	Returns the value of the postprocessing gain. Manually multiply the fixed-point output signal by the postprocessing gain to obtain the floating-point output signal or automatically use DFD FXP MRate Postprocessing VI to handle the postprocessing gain.

Multirate FIR Design Details

As you define a filter specification, you must adhere to a set of rules to maintain valid specifications. If you do not adhere to the following rules, the **Configure Multirate Filter Design** dialog box displays a message in the **Tips** indicator with suggestions for repositioning the cursors.

- Keep the horizontal cursors in the range (0, 1) in a linear scale or (–inf, 0 dB) in a logarithmic scale.
- Keep the horizontal passband cursor above the horizontal stopband cursor.
- The **Passband edge frequency** value must be less than the Nyquist frequency, or you must keep the vertical passband cursor to the left of the Nyquist cursor.
- The **Stopband edge frequency** value must be greater than the **Passband edge frequency** value, or you must keep the vertical passband cursor to the left of the stopband cursor.
- If you remove the checkmark from the Transition band aliasing allowed checkbox to avoid aliasing in the transition band, keep the Stopband edge frequency value between the Passband edge frequency value and the Nyquist frequency, or keep the vertical stopband cursor between the vertical passband and Nyquist cursors. If you place a checkmark in the Transition band aliasing allowed checkbox to allow aliasing in the transition band, keep the vertical stopband cursor between the vertical passband cursor and the vertical stopband limit cursor.

Multistage Multirate Filter Design Express VI

Owning Palette: <u>Multirate Filter Design VIs</u> Installed With: Digital Filter Design Toolkit Creates a multistage multirate filter.

Dialog Box Options

Block Diagram Outputs

■ Place on the block diagram ■ Find on the **Functions** palette

Dialog Box Options

Parameter	Description
Magnitude Response	Contains the plot of the magnitude response. You can drag the cursors in the plot to change the specifications. The color you specify in Passband represents the passband response and the passband edge frequency. The color you specify in Stopband represents the stopband attenuation and the stopband edge frequency. The color you specify in Stopband limit represents the stopband limit. The color you specify in Nyquist represents the half sampling frequency, also known as the Nyquist frequency.
	 Magnitude in dB—Specifies whether this Express VI uses decibels or a linear scale to express the magnitude response. If you place a checkmark in the checkbox, this Express VI converts a linear magnitude response to decibels. This Express VI uses decibels by default.
	 Nyquist—Specifies the color of the line in the magnitude plot that represents the Nyquist frequency. The default is green. Passband—Specifies the color of the lines in the magnitude plot that represent the passband response and the passband edge frequency. The default is blue. Stopband limit—Specifies the color of the line in the magnitude plot that represents the maximum stopband edge frequency. The default is yellow. Stopband—Specifies the color of the lines in the magnitude plot that represent the stopband attenuation and the stopband edge frequency. The default is yellow. Response selection—Specifies the stage for which you want to check the magnitude

	response.
Flow Diagram	Returns the flow diagram of the multirate filter you specify in Filter type .
Filter Coefficients	Returns the coefficients of the multirate filter according to the settings you specify on the Floating-Point Design page.
	• Filter order —Displays the order of the multirate filter you design. The value of Filter order +1 equals the number of coefficients.
	• Coefficients selection —Specifies the stage for which you want to check the filter coefficients.
Main Settings	Contains the following options:
	• Filter type—Specifies the type of multirate filter that this VI creates. The valid values include No Rate Change, Interpolation, and Decimation. The default is Decimation.
	• Filter factor—Specifies the sampling frequency conversion factor of the multirate filter. If you set Filter type to No Rate Change, the value of this input must be an integer greater than or equal to 1. The default is 1. If you set Filter type to Decimation or Interpolation, the value of this input must be an integer greater than or equal to 2. The default then is 2.
	• Number of stages—Specifies the number of stages of the multirate filter that this VI creates. The valid value of Number of stages is an integer between 1 and 3. Within this range, the value of Number of stages must be less than or equal to the number of elements in the prime factorization of Filter factor.
	• Using CIC—Specifies if you want to use the <u>cascaded integrator comb (CIC) filter</u> design method to design one stage of the multistage multirate filters. This option is available only if Filter factor is greater than 4 and divisible by 4.

	 The default is FALSE, which means this Express VI does not use the CIC filter design method. If the value is TRUE, this Express VI uses the CIC filter design method to design the first stage of the filter when you set filtering mode to Decimation, or the last stage of the filter when you set filtering mode to Interpolation. Subfactors—Specifies the factors of each stage of the multirate filter. Contains the following options: Subfactor 1—Specifies the sampling frequency conversion factor of the first stage of the multirate filter. This option is available only if Number of stages is equal to or greater than 2 and Filter factor is not a prime number. Subfactor 2—Specifies the sampling frequency conversion factor of the second stage of the multirate filter. This option is available only if Number of stages is equal to or greater than 2 and Filter factor is not a prime number. Subfactor 3—Specifies the sampling frequency conversion factor of the second stage of the multirate filter. This option is available only if Number of stages is equal to or greater than 2 and if the number of elements in the prime factorization of Filter factor is equal to or greater than 2. Subfactor 3—Specifies the sampling frequency conversion factor of the third stage of the multirate filter. This option is available only if Number of stages is equal to 3 and if the number of elements in the prime factorization of Filter factor is greater than 2.
Filter	Contains the following options:
Specifications	 Input sampling frequency—Specifies the sampling frequency, in hertz, of an input signal for the multirate filter. This input must contain a value greater than zero. The default is 1k Hz. Passband edge frequency—Specifies the

	 passband edge frequency of the multirate filter. The default is 200 Hz. Stopband edge frequency—Specifies the stopband edge frequency of the multirate filter. The default is 250 Hz. Output sampling frequency—Returns the sampling frequency of the output signal for the multirate filter. Passband ripple—Specifies the passband ripple of the multirate filter in a unit that the Magnitude in dB option determines. If you place a checkmark in the Magnitude in dB checkbox, Passband ripple must be greater than zero. The default is 0.1 dB. If you remove the checkmark from the Magnitude in dB checkbox, the valid range of Passband ripple is (0, 1). The default then is 0.011. Stopband attenuation—Specifies the stopband attenuation of the multirate filter in a unit that the Magnitude in dB option determines. If you place a checkmark from the Magnitude in dB checkbox, the valid range of Passband ripple is (0, 1). The default then is 0.011. Stopband attenuation—Specifies the stopband attenuation of the multirate filter in a unit that the Magnitude in dB option determines. If you place a checkmark in the Magnitude in dB checkbox, Stopband attenuation must be greater than zero. The default is 60 dB. If you remove the checkmark from the Magnitude in dB checkbox, the valid range of Stopband attenuation is (0, 1). The default then is 0.011.
Transition band aliasing allowed	Specifies if you allow frequency aliasing in the transition band.
Quantization Settings	 Contains the following options: Quantize filter—Specifies if you want to use this Express VI to quantize the floating-point multirate filter you design. If you enter the multirate filter specifications, you can place a checkmark in the Quantize filter checkbox after you click the Update Design button. Coefficients word lengths—Specifies the word

lengths, in number of bits, that this Express VI uses to represent the filter coefficients of each stage. Contains the following options:

- Coefficients word length 1—Specifies the word length of the first stage filter coefficients of the multirate filter. This option is available only if you place a checkmark in the Quantize filter checkbox. The valid range is [1, 32]. The default is 16.
- Coefficients word length 2—Specifies the word length of the second stage filter coefficients of the multirate filter. This option is available only if Number of stages is equal to or greater than 2 and if you place a checkmark in the Quantize filter checkbox. The valid range is [1, 32]. The default is 16.
- Coefficients word length 3—Specifies the word length of the third stage filter coefficients of the multirate filter. This option is available only if Number of stages is equal to 3 and if you place a checkmark in the Quantize filter checkbox. The valid range is [1, 32]. The default is 16.
- Gain word length—Specifies the word length, in number of bits, that this Express VI uses to represent the multirate filter gain if the gain processing occurs on an NI Reconfigurable I/O (RIO) target. This option is available only if you place a checkmark in the Quantize filter checkbox and select On Target from the Gain processing pull-down menu. The valid range is [1, 32]. The default is 16.
- **Coefficients scale type**—Specifies how this Express VI scales the multirate filter coefficients. This option is available only if you place a

	 checkmark in the Quantize filter checkbox. The valid values include No Norm, Time Domain-1 Norm, Time Domain-2 Norm, and Time Domain-Inf Norm. The default is Time Domain-1 Norm. Gain processing—Specifies whether the gain processing occurs on a host machine or an NI-RIO target. This option is available only if you place a checkmark in the Quantize filter checkbox. Scale by power of 2—Specifies whether this Express VI scales the multirate filter with the original norm value or the smallest power of 2 value that is greater than the norm value. This option is available only if you place a checkmark in the Quantize filter checkbox. If you place a checkmark in the Scale by power of 2 checkbox, this Express VI scales the multirate filter with the smallest power of 2 checkbox, this Express VI scales the multirate filter with the smallest power of 2 checkbox, this Express VI scales the multirate filter with the smallest power of 2 checkbox, this Express VI scales the multirate filter with the smallest power of 2 checkbox, this Express VI scales the multirate filter with the original norm value. Postprocessing filter gain—Displays the value of the postprocessing gain to obtain the floating-point output signal or automatically use the DFD FXP MRate Postprocessing VI to handle the postprocessing gain.
Tips	Displays tips and error messages that help you use this Express VI to design a multirate filter.
Update Design	Updates the floating-point multirate filter design with the specifications you entered. When you click the Update Design button, you enable the Quantize filter option on the Fixed-Point Quantization page.

Block Diagram Outputs

Parameter	Description
error out	Describes the error status that this VI or function produces.
multirate filters out	Returns the new multirate filters.
postprocessing filter gain	Returns the value of the postprocessing gain. Manually multiply the fixed-point output signal by the postprocessing gain to obtain the floating-point output signal or automatically use DFD FXP MRate Postprocessing VI to handle the postprocessing gain.

DFD Halfband Design VI

Owning Palette: <u>Multirate Filter Design VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a <u>halfband filter</u> with an automatic order estimation.

You can use the filter as a single-rate filter, an interpolation filter, or a decimation filter.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

method specifies the method this VI uses to create the filter.

0 Kaiser Window

1 Dolph-Chebyshev Window

2 Equi-Ripple

3 Positive Equi-Ripple (default)

4 Max Flat

filter type specifies the <u>type of filter</u> that this VI creates.

0	Lowpass (default)
1	Highpass

- **order** specifies the filter order. The value of **order** must be an even integer that is greater than zero. The default is -1. If **order** is not greater than zero, this VI uses the **stopband attenuation** input to estimate filter order. If **order** is an odd number, this VI returns an error.
- **roll off** determines the relative transition bandwidth, which equals (transition band)/(2*passband+transition band). The default is 0.2. **roll off** must be in the range (0, 1). If the passband is fixed, a

smaller **roll off** value results in a narrower transition bandwidth.

- **stopband attenuation** specifies the stopband attenuation in decibels. The value must be greater than zero. The default is 40. If **order** is greater than zero, this VI ignores **stopband attenuation** and uses **order** to create the filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filtering mode** specifies the processing mode of the filter that this VI creates.

0 **No Rate Change**—Does not change the sampling frequency of a signal.

1 **Interpolation**—Increases the sampling frequency of a signal to a higher sampling frequency that differs from the original frequency by an integer value. <u>Interpolation</u> also is known as up-sampling.

2 **Decimation** (default)—Reduces the sampling frequency of a

signal to a lower sampling frequency that differs from the original frequency by an integer value. <u>Decimation</u> also is known as down-sampling.

- **multirate filter out** returns a new multirate filter.
- **order out** returns the actual order of the new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Halfband Filter VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Multirate directory for an example of using the DFD Halfband Design VI.

■ Open example ■ Browse related examples
DFD MRate Filter Design VI

Owning Palette: <u>Multirate Filter Design VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a single-stage multirate filter. You must <u>manually select the</u> <u>polymorphic instance</u> you want to use.

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

•

■ Place on the block diagram ■ Find on the **Functions** palette

DFD MRate Filter Design (Integer)



method specifies the method this VI uses to create the filter.



- **factor** specifies the sampling frequency conversion factor of the multirate filter. The value of **factor** must be an integer greater than zero. The default is 8.
- **order** specifies the filter order. The value of **order** must be an even integer that is greater than zero. If **order** is not greater than zero, this VI uses the **ripple specs** input to estimate filter order. If **order** is an odd number, this VI returns an error. The default is -1.
- **freq specs** specifies the passband edge frequency and stopband edge frequency of the multirate filter.
 - **fpass** specifies the <u>passband edge frequency</u> of the multirate filter.
 - **fstop** specifies the <u>stopband edge frequency</u> of the multirate filter.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The default is 60.
 - **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE,

this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs in** specifies the input sampling frequency of the multirate filter in hertz. The default is 800.
- **filtering mode** specifies the processing mode of the filter that this VI creates.
 - 0 **No Rate Change**—Does not change the sampling frequency of a signal.
 - 1 **Interpolation**—Increases the sampling frequency of a signal to a higher sampling frequency that differs from the original frequency by an integer value. <u>Interpolation</u> also is known as up-sampling.

2 **Decimation** (default)—Reduces the sampling frequency of a

signal to a lower sampling frequency that differs from the original frequency by an integer value. <u>Decimation</u> also is known as down-sampling.

- multirate filter out returns a new multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD MRate Filter Design (Rational)



method specifies the method this VI uses to create the filter.

0 Kaiser Window

1 Dolph-Chebyshev Window

2 **Equi-Ripple** (default)

- **factor** contains the sampling frequency conversion factor of the multirate filter.
 - L contains the numerator factor of the rational resampling frequency conversion factor. The value of L must be an integer greater than zero. The value of L must not equal the value of M. The default is 8.
 - M contains the denominator factor of the rational resampling frequency conversion factor. The value of **M** must be an integer greater than zero. The value of **M** must not equal the value of **L**. The default is 3.
- **order** specifies the filter order. The value of **order** must be an even integer that is greater than zero. If **order** is not greater than zero, this VI uses the **ripple specs** input to estimate filter order. If **order** is an odd number, this VI returns an error. The default is –1.
- **freq specs** specifies the passband edge frequency and stopband edge frequency of the multirate filter.
 - **fpass** specifies the <u>passband edge frequency</u> of the multirate filter.
 - **fstop** specifies the <u>stopband edge frequency</u> of the multirate filter.
- **ripple specs** specifies the ripple level in the passband and stopband of the filter.

- **passband** specifies the ripple level in the passband. The default is 0.1.
- **stopband** specifies the ripple level in the stopband. The default is 60.
- **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs in** specifies the input sampling frequency of the multirate filter in hertz. The default is 100.
- **multirate filter out** returns a new multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Single Stage Multirate Filter Design VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Multirate directory for an example of using the DFD MRate Filter Design VI.

DFD NStage MRate Filter Design VI

Owning Palette: <u>Multirate Filter Design VIs</u>

Installed With: Digital Filter Design Toolkit

Creates multistage multirate filters, which meet the requirements you specify by cascading filters in **multirate filters out**.

Wire the **multirate filters out** output to the **multirate filters in** input of the DFD NStage MRate Filtering VI or the DFD NStage MRate Filtering for Single Block VI if you want to process data with the new filter.

Details Examples



■ Place on the block diagram ■ Find on the **Functions** palette

- **manual factorization** specifies the factors for each stage. The product of all the factors in the **manual factorization** input must equal the **factor** input value.
- **factor** specifies the sampling frequency conversion factor of the multirate filter. **factor** must be greater than 1. The default is 8. If you set **using CIC?** to TRUE, **factor** must be greater than 4 and divisible by 4.

using CIC? specifies if you want to use the <u>cascaded integrator</u> <u>comb (CIC) filter</u> design method to design one stage of the multistage multirate filters. The default is FALSE, which means this VI does not use the CIC filter design method. If the value is TRUE, this VI uses the CIC filter design method to design the first stage of the filter when you set **filtering mode** to Decimation, or the last stage of the filter when you set **filtering mode** to Interpolation.

- **freq specs** specifies the passband edge frequency and stopband edge frequency of the multistage multirate filters.
 - **fpass** specifies the <u>passband edge frequency</u> of the multistage multirate filters. The default is 45.

fstop specifies the <u>stopband edge frequency</u> of the multistage multirate filters. The default is 52.

- **ripple specs** specifies the ripple level in the passband and stopband of the filter.
 - **passband** specifies the ripple level in the passband. The default is 0.1.
 - **stopband** specifies the ripple level in the stopband. The default is 60.
 - **dB/linear?** specifies whether this VI applies a decibel scale or a linear scale to the ripple levels. If the value is TRUE, this VI applies a decibel scale to the ripple level. If the value is FALSE, this VI applies a linear scale to the ripple level. The default is TRUE.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **fs in** specifies the input sampling frequency of the multistage multirate filters in hertz. The default is 800.

- **filtering mode** specifies the processing mode of the filter that this VI creates.
 - 0 **No Rate Change**—Does not change the sampling frequency of a signal.
 - 1 **Interpolation**—Increases the sampling frequency of a signal to a higher sampling frequency that differs from the original frequency by an integer value. <u>Interpolation</u> also is known as up-sampling.
 - 2 **Decimation** (default)—Reduces the sampling frequency of a signal to a lower sampling frequency that differs from the original frequency by an integer value. <u>Decimation</u> also is known as down-sampling.
- **multirate filters out** returns the new multistage multirate filters.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD NStage MRate Filter Design Details

Use the DFD NStage MRate Filter Design VI to create multirate filters with large multirate factors. This VI distributes a large factor into each stage of the multirate filter, as shown in the following example. A multistage design requires less computation and storage than a single-stage design.

For example, if the sampling frequency conversion factor *M* contains the factors M_1 , M_2 , and M_3 , such that $M = M_1 * M_2 * M_3$, and $M_1 > M_2 > M_3 > 1$, you can design a three-stage multirate filter in which all three stages use the same multirate processing mode that **filtering mode** specifies.

If you set **using CIC?** to TRUE, the factor *M* must be divisible by 4, for example, $M = 2*2*M_1$. In this case, the filter with the largest factor M_1 is a CIC filter.

The overall response of cascading the multirate filters is a lowpass response.

Examples

Refer to the following VIs for examples of using the DFD NStage MRate Filter Design VI:

• Analyze Coefficients-Quantized Multistage Multirate Filter VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate

Open example Browse related examples

• Multistage Decimation Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Multirate

■ Open example ■ Browse related examples

• Multistage Multirate Filter Design (with CIC) VI: labview\examples\Digital Filter Design\Floating-Point Filters\Multirate

■ Open example ■ Browse related examples

• Multistage Multirate Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Multirate

DFD Nyquist Design VI

Owning Palette: <u>Multirate Filter Design VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a <u>Nyquist filter</u> using the window or equi-ripple methods with an automatic order estimation.

You can use the Nyquist filter as a single-rate filter, an interpolation filter, or a decimation filter.

Example



Place on the block diagram Find on the Functions palette

method specifies the <u>method this VI uses to create the filter</u>.

0 Kaiser Window

1 Dolph-Chebyshev Window

2 Equi-Ripple

3 **Positive Equi-Ripple** (default)

- **factor** specifies the sampling frequency conversion factor of the multirate filter. The default is 4. The passband of the filter depends on **factor** by Nyquist sampling theorem.
- **order** specifies the filter order. The value of **order** must be an even integer that is greater than zero. The default is -1. If **order** is not greater than zero, this VI uses the **stopband attenuation** input to estimate filter order. If **order** is an odd number, this VI returns an error.
- **roll off** determines the relative transition bandwidth, which equals (transition band)/(2*passband + transition band). The default is 0.2. **roll off** must be in the range of (0,1). If the value of **factor** is fixed, a smaller **roll off** value results in a narrower transition bandwidth.

- **stopband attenuation** specifies the stopband attenuation in decibels. The value must be greater than zero. The default is 40. If **order** is greater than zero, this VI ignores **stopband attenuation** and uses **order** to create the filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filtering mode** specifies the processing mode of the filter that this VI creates.
 - 0 **No Rate Change**—Does not change the sampling frequency of a signal.
 - 1 **Interpolation**—Increases the sampling frequency of a signal to a higher sampling frequency that differs from the original frequency by an integer value. <u>Interpolation</u> also is known as up-sampling.
 - 2 **Decimation** (default)—Reduces the sampling frequency of a signal to a lower sampling frequency that differs from the original frequency by an integer value. <u>Decimation</u> also is

known as down-sampling.

- multirate filter out returns a new multirate filter.
- **order out** returns the actual order of the new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Nyquist Filter VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Multirate directory for an example of using the DFD Nyquist Design VI.

DFD Raised Cosine Design VI

Owning Palette: <u>Multirate Filter Design VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a <u>raised cosine or a root-raised cosine</u> finite impulse response (FIR) filter.

You can use the filter as a single-rate filter, an interpolation filter, or a decimation filter.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

window specifies the time-domain window this VI uses.

0	None (default)
1	Hanning
2	Hamming
3	Blackman-Harris
4	Exact Blackman
5	Blackman
6	Flat Top
7	4 Term B-Harris
8	7 Term B-Harris
9	Low Sidelobe
30	Triangular

- **factor** specifies the sampling frequency conversion factor of the multirate filter. The default is 4. The passband of the filter depends on **factor** by Nyquist sampling theorem.
- **order** specifies the filter order. The value must be an even integer

that is greater than zero. The default is 10. If **order** is an odd number, this VI returns an error. Increasing the value of **order** can increase the stopband attenuation.

- **roll off** determines the relative transition bandwidth, which equals (transition band)/(2*passband + transition band). The default is 0.2. **roll off** must be in the range of [0,1]. If the value of **factor** is fixed, a smaller **roll off** value results in a narrower transition bandwidth.
- **type** specifies the type of filter that this VI creates.

0 Raised Cosine (default)1 Root Raised Cosine

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filtering mode** specifies the processing mode of the filter that this VI creates.

0 **No Rate Change**—Does not change the sampling frequency of

a signal.

- 1 **Interpolation**—Increases the sampling frequency of a signal to a higher sampling frequency that differs from the original frequency by an integer value. <u>Interpolation</u> also is known as up-sampling.
- 2 **Decimation** (default)—Reduces the sampling frequency of a signal to a lower sampling frequency that differs from the original frequency by an integer value. <u>Decimation</u> also is known as down-sampling.
- **multirate filter out** returns a new multirate filter.
- **order out** returns the actual order of the new filter.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Raised Cosine Filter VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Multirate directory for an example of using the DFD Raised Cosine Design VI.

Multirate Fixed-Point Tools VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Multirate Fixed-Point Tools VIs to quantize filter coefficients, model the behavior of fixed-point multirate filters, simulate multirate filtering processes, and generate fixed-point target code.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
<u>DFD FXP Get</u> <u>MRate Coef</u> Quantizer	Retrieves settings for the filter coefficients quantizer of a multirate filter.
DFD FXP Get MRate Output Quantizer	Retrieves the settings for the output quantizer of a multirate filter.
DFD FXP Moving Average Code Generator	Generates LabVIEW field-programmable gate array (FPGA) code from a fixed-point moving average (MA) multirate filter.
<u>DFD FXP</u> <u>MRate Code</u> <u>Generator</u>	Generates LabVIEW field-programmable gate array (FPGA) code from a fixed-point multirate filter.
<u>DFD FXP</u> <u>MRate</u> Modeling	Creates a fixed-point multirate filter model according to the input and output word length settings.
DFD FXP MRate Postprocessing	Converts the output signal of a fixed-point multirate filter from an integer, fixed-point representation to a floating- point representation. You must <u>manually select the</u> <u>polymorphic instance</u> you want to use.
DFD FXP MRate Quantization	Quantizes the coefficients of a floating-point multirate filter and generates a fixed-point multirate filter.

DFD FXP MRate Simulation	Simulates the filtering process of a fixed-point multirate filter continuously. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually</u> <u>select</u> the instance.
DFD FXP NStage MRate Code Generator	Generates LabVIEW field-programmable gate array (FPGA) code from multistage multirate filters.
DFD FXP NStage MRate Modeling	Creates a fixed-point multistage multirate filter model according to the input and output word length settings.
DFD FXP NStage MRate Postprocessing	Converts the output signal of a fixed-point multistage multirate filter from a fixed-point representation to a floating-point representation. You must <u>manually select</u> <u>the polymorphic instance</u> you want to use.
DFD FXP NStage MRate Quantization	Quantizes the coefficients of a floating-point multistage multirate filter and generates a fixed-point multistage multirate filter. This VI has the same internal settings as the <u>DFD FXP MRate Quantization</u> VI.
DFD FXP NStage MRate Simulation	Simulates the filtering process of a fixed-point multistage multirate filter continuously. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.

DFD FXP Get MRate Coef Quantizer VI

Owning Palette: <u>Multirate Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Retrieves settings for the filter coefficients quantizer of a multirate filter.



■ Place on the block diagram ■ Find on the **Functions** palette

- **multirate filter in** specifies the input fixed-point multirate filter. You must specify a finite impulse response (FIR) multirate filter for this input.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- multirate filter out returns the multirate filter in unchanged.
- **coefficients quantizer** returns the settings of the filter coefficients quantizer.

- **source** returns the quantizer source.
- wl returns the word length, in number of bits, that the quantizer uses to represent a fixed-point number.
- **iwl** returns the integer word length, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number.
- **overflow mode** returns the operation mode for overflow and underflow in the quantizer.
- **rounding mode** returns the mode for rounding numbers in the quantizer.
- **signed?** is TRUE if the fixed-point number is a signed number. **signed?** is FALSE if the fixed-point number is an unsigned number.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Get MRate Output Quantizer VI

Owning Palette: Multirate Fixed-Point Tools VIs

Installed With: Digital Filter Design Toolkit

Retrieves the settings for the output quantizer of a multirate filter.



■ Place on the block diagram ■ Find on the **Functions** palette

multirate filter in specifies the input fixed-point multirate filter.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- multirate filter out returns the multirate filter in unchanged.
- **output quantizer** returns the settings of the output quantizer.
 - **source** returns the quantizer source.
 - wl returns the word length, in number of bits, that the

quantizer uses to represent a fixed-point number.

- **iwl** returns the integer word length, in number of bits, within **wl** that the quantizer uses to represent the integer part of a fixed-point number.
- **overflow mode** returns the operation mode for overflow and underflow in the quantizer.
- **rounding mode** returns the mode for rounding numbers in the quantizer.
- **signed?** is TRUE if the fixed-point number is a signed number. **signed?** is FALSE if the fixed-point number is an unsigned number.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Moving Average Code Generator VI

Owning Palette: Multirate Fixed-Point Tools VIs

Installed With: Digital Filter Design Toolkit

Generates LabVIEW field-programmable gate array (FPGA) code from a fixed-point moving average (MA) multirate filter.

Details Example



■ Place on the block diagram ■ Find on the **Functions** palette

- **open project?** specifies if this VI opens the project file after generating the code. The default is FALSE, which means that you must open the project file manually after this VI generates the code.
- # channels specifies the number of channels that you want the generated code to process. The default is 1.
- moving average filter specifies the input moving average filter.
- destination folder specifies the path to the folder in which you want to save the generated code. This VI returns an error if you do not specify a valid path to the folder.
- **Filter name** specifies a name for the multirate filter code that this VI generates. This VI also uses this value as the filename of the project file that contains the generated filter code. You can use only letters and digits in the **filter name** input. This VI ignores other characters. If you specify an invalid name, this VI creates a string that starts with Unknown.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while

this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **confirm?** specifies if you want this VI to ask you for confirmation before replacing an existing file. If the value is TRUE, this VI displays a dialog box asking for confirmation to replace the existing file. If the value is FALSE, this VI replaces the existing file automatically. The default is TRUE.
- **Ivproj path** returns the path to the generated project file.
- **sampling frequency/FPGA clock** returns a ratio. You can multiply this ratio with a specific FPGA clock rate to calculate the maximum input sampling frequency per channel that the generated FPGA code can process at the FPGA clock rate. For example, if the ratio is 0.05 and the FPGA clock rate is 40 MHz, then the maximum input sampling frequency per channel that the generated FPGA code can process is 2 MHz.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP Moving Average Code Generator Details

An MA filter is a lowpass, fixed-point <u>cascaded integrator comb (CIC)</u> <u>filter</u> that meets the following criteria:

- The filtering mode of the CIC filter is no-rate change or decimation.
- The number of stages of the CIC filter is 1.
- The differential delay of the CIC filter is 1.
- The values of the sampling frequency conversion factor and input word length must satisfy the following equation:

 $\log_2(a) + b \le 32$

where *a* is the sampling frequency conversion factor of the CIC filter and *b* is the input word length that you specified when creating a fixed-point CIC filter model.

• The internal precision setting must be Full.

Example

Refer to the Generate LabVIEW FPGA Code for Moving Average Filter VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate directory for an example of using the DFD FXP Moving Average Code Generator VI.

DFD FXP MRate Code Generator VI

Owning Palette: Multirate Fixed-Point Tools VIs

Installed With: Digital Filter Design Toolkit

Generates LabVIEW field-programmable gate array (FPGA) code from a fixed-point multirate filter.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

- **open project?** specifies if this VI opens the project file after generating the code. The default is FALSE, which means that you must open the project file manually after this VI generates the code.
- # channels specifies the number of channels that you want the generated code to process. The default is 1.
- **multirate filter** specifies the input multirate filter.
- destination folder specifies the path to the folder in which you want to save the generated code. This VI returns an error if you do not specify a valid path to the folder.
- **filter name** specifies a name for the multirate filter code that this VI generates. This VI also uses this value as the filename of the project file that contains the generated filter code. You can use only letters and digits in the **filter name** input. This VI ignores other characters. If you specify an invalid name, this VI creates a string that starts with Unknown.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while

this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **confirm?** specifies if you want this VI to ask you for confirmation before replacing an existing file. If the value is TRUE, this VI displays a dialog box asking for confirmation to replace the existing file. If the value is FALSE, this VI replaces the existing file automatically. The default is TRUE.
- **Ivproj path** returns the path to the generated project file.
- **sampling frequency/FPGA clock** returns a ratio. You can multiply this ratio with a specific FPGA clock rate to calculate the maximum input sampling frequency per channel that the generated FPGA code can process at the FPGA clock rate. For example, if the ratio is 0.05 and the FPGA clock rate is 40 MHz, then the maximum input sampling frequency per channel that the generated FPGA code can process is 2 MHz.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Generate LabVIEW FPGA Code for Multirate Filter VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate directory for an example of using the DFD FXP MRate Code Generator VI.
DFD FXP MRate Modeling VI

Owning Palette: <u>Multirate Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a fixed-point multirate filter model according to the input and output word length settings.

Example



 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

- **multirate filter in** specifies the input multirate filter.
- input word length specifies the word length, in number of bits, that this VI uses to represent the input signal. The valid range is [1, 32]. The default is 16.
- **output word length** specifies the word length, in number of bits, that this VI uses to represent the output signal. The valid range is [1, 32]. The default is 16.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**

is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.



Г	4	۲	1
-			_

output rounding mode specifies the <u>rounding mode</u> this VI uses in the output quantizer.

0 **Nearest**—Rounds to the closest representable number.

1 **Truncation** (default)—Rounds to the closest representable number less than the original value.

internal precision specifies the precision of the internal signals.

- Truncated (default)—Approximates a full-precision result and helps spare logical resources of the FPGA hardware target. This option is valid for all finite impulse response (FIR) multirate filters and <u>cascaded integrator comb (CIC) decimation filters</u>.
- 1 **Full**—Provides a full-precision result. You must use this option if the filter is a moving average (MA) multirate filter. Otherwise the simulation cannot return the same results as the code generation results.
- **multirate filter out** returns a fixed-point multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Fixed-Point Multirate Filter Modeling and Simulation VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate directory for an example of using the DFD FXP MRate Modeling VI.

■ Open example ■ Browse related examples

DFD FXP MRate Postprocessing VI

Owning Palette: Multirate Fixed-Point Tools VIs

Installed With: Digital Filter Design Toolkit

Converts the output signal of a fixed-point multirate filter from an integer, fixed-point representation to a floating-point representation. You must manually select the polymorphic instance you want to use.

Ŧ

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD FXP MRate Postprocessing (I32, nCh)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- # channels specifies the number of channels that signal in contains. The default is 1.
- **signal in** specifies the input signal that you want to process.
- multirate filter specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **history in** specifies the data from the last iteration of the

postprocessing process.

- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **history out** returns the remaining data for the next iteration of postprocessing. You can wire this output to the **history in** input of the next call to this VI if you want to process the data continuously.

DFD FXP MRate Postprocessing (I16, nCh)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- # channels specifies the number of channels that signal in contains. The default is 1.
- **signal in** specifies the input signal that you want to process.
- multirate filter specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **history in** specifies the history data from the last iteration of

postprocessing.

- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **history out** returns the remaining data for the next iteration of postprocessing. You can wire this output to the **history in** input of the next call to this VI if you want to process the data continuously.

DFD FXP MRate Postprocessing (I32, 1Ch)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP MRate Postprocessing (I16, 1Ch)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP MRate Quantization VI

Owning Palette: Multirate Fixed-Point Tools VIs

Installed With: Digital Filter Design Toolkit

Quantizes the coefficients of a floating-point multirate filter and generates a fixed-point multirate filter.

Details Examples



■ Place on the block diagram ■ Find on the **Functions** palette

- **scale by power of 2?** specifies whether this VI scales the multirate filter with the original norm value or the smallest power of 2 value that is greater than the norm value. If **scale by power of 2?** is TRUE, this VI scales the multirate filter with the smallest power of 2 value. If **scale by power of 2?** is FALSE, this VI scales the multirate filter with the original norm value. The default is FALSE.
- **scale type** specifies how to scale the multirate filter coefficients. Refer to the <u>Details</u> section of the DFD Scale Filter VI for more information about each scale type.

0 No Norm

1 **Time Domain 1-Norm** (default)

2 **Time Domain 2-Norm**

3 Time Domain Inf-Norm

- **multirate filter in** specifies the input multirate filter.
- **coefficients word length** specifies the <u>word length</u>, in number of bits, that this VI uses to represent the multirate filter coefficients. The valid range is [1, 32]. The default is 16.
- **error in** describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **gain settings** specifies the settings for the multirate filter gain.
 - **gain processing** specifies whether you want to process the multirate filter gain on a host machine or an NI Reconfigurable I/O (RIO) target.
 - 0 **On Target**—Specifies to process the multirate filter gain on an NI-RIO target.
 - 1 **On Host** (default)—Specifies to process the multirate filter gain on a host machine.
 - gain word length specifies the word length, in number of bits, that this VI uses to represent the multirate filter gain if you set gain processing to On Target. The valid range is [1, 32]. The default is 16.
- multirate filter out returns a fixed-point multirate filter.
- **postprocessing gain** returns the filter gain that this VI scaled.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP MRate Quantization Details

The DFD FXP MRate Quantization VI automatically determines a quantizer for the filter coefficients according to the **scale type** you specify. The input coefficients word length determines the word length of the quantizer, and the maximum absolute value among the filter coefficients determines the integer word length of the quantizer. This VI sets the overflow mode and rounding mode to saturation and nearest, respectively.



Tip You can use the <u>DFD FXP Get MRate Coef Quantizer</u> VI to retrieve the quantizer information.

Examples

Refer to the following VIs for examples of using the DFD FXP MRate Quantization VI:

• How to Build Coefficients Quantizer VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

■ Open example ■ Browse related examples

- Analyze Coefficients-Quantized Multirate Filter VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate
 - Open example Browse related examples

DFD FXP MRate Simulation VI

Owning Palette: <u>Multirate Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Simulates the filtering process of a fixed-point multirate filter continuously. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

-

■ Place on the block diagram ■ Find on the **Functions** palette

DFD FXP MRate Simulation (DBL)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **init?** specifies how you want to initialize the internal states. The default is TRUE, which specifies that this VI initializes the internal states to zero. If **init?** is FALSE, this VI initializes the internal states from the final states of the previous call to the current VI instance. To process a large data sequence, split the sequence into smaller blocks, set **init?** to TRUE for the first block, and set **init?** to FALSE for the remaining blocks.
- **signal in** specifies the input signal you want to process. The **input word length** value you set on the <u>DVD FXP MRate Modeling</u> VI determines the range of **signal in**. The range equals [-2^(**input word length**-1), 2^(**input word length**-1)-1]. For example, if you specify 16 as the **input word length** value, the corresponding range is [-32768, 32767].
- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DBL signal out** returns the output floating-point signal after postprocessing.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP MRate Simulation (I32)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **init?** specifies how you want to initialize the internal states. The default is TRUE, which specifies that this VI initializes the internal states to zero. If **init?** is FALSE, this VI initializes the internal states from the final states of the previous call to the current VI instance. To process a large data sequence, split the sequence into smaller blocks, set **init?** to TRUE for the first block, and set **init?** to FALSE for the remaining blocks.
- **signal in** specifies the input signal that you want to process.
- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DBL signal out** returns the output floating-point signal after postprocessing.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP MRate Simulation (I16)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **init?** specifies how you want to initialize the internal states. The default is TRUE, which specifies that this VI initializes the internal states to zero. If **init?** is FALSE, this VI initializes the internal states from the final states of the previous call to the current VI instance. To process a large data sequence, split the sequence into smaller blocks, set **init?** to TRUE for the first block, and set **init?** to FALSE for the remaining blocks.
- **signal in** specifies the input signal that you want to process.
- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DBL signal out** returns the output floating-point signal after postprocessing.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Fixed-Point Multirate Filter Modeling and Simulation VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate directory for an example of using the DFD FXP MRate Simulation VI.

■ Open example ■ Browse related examples

DFD FXP NStage MRate Code Generator VI

Owning Palette: Multirate Fixed-Point Tools VIs

Installed With: Digital Filter Design Toolkit

Generates LabVIEW field-programmable gate array (FPGA) code from multistage multirate filters.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

- **open project?** specifies if this VI opens the project file after generating the code. The default is FALSE, which means that you must open the project file manually after this VI generates the code.
- # channels specifies the number of channels that you want the generated code to process. The default is 1.
- **multirate filters** specifies the input multistage multirate filters.
- destination folder specifies the path to the folder in which you want to save the generated code. This VI returns an error if you do not specify a valid path to the folder.
- **filter name** specifies a name for the multistage multirate filter code that this VI generates. This VI also uses this value as the filename of the project file that contains the generated filter code. You can use only letters and digits in the **filter name** input. This VI ignores other characters. If you specify an invalid name, this VI creates a string that starts with Unknown.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while

this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **confirm?** specifies if you want this VI to ask you for confirmation before replacing an existing file. If the value is TRUE, this VI displays a dialog box asking for confirmation to replace the existing file. If the value is FALSE, this VI replaces the existing file automatically. The default is TRUE.
- **Ivproj path** returns the path to the generated project file.
- **sampling frequency/FPGA clock** returns a ratio. You can multiply this ratio with a specific FPGA clock rate to calculate the maximum input sampling frequency per channel that the generated FPGA code can process at the FPGA clock rate. For example, if the ratio is 0.05 and the FPGA clock rate is 40 MHz, then the maximum input sampling frequency per channel that the generated FPGA code can process is 2 MHz.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Generate LabVIEW FPGA Code for Multistage Multirate Filter VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate directory for an example of using the DFD FXP NStage MRate Code Generator VI.

■ Open example ■ Browse related examples

DFD FXP NStage MRate Modeling VI

Owning Palette: <u>Multirate Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a fixed-point multistage multirate filter model according to the input and output word length settings.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

- **interstage word lengths** specifies the word lengths, in number of bits, that this VI uses to represent the signal between every two consecutive stages of the multirate filter.
- **multirate filters in** specifies the input multistage multirate filter.
- input word length specifies the word length, in number of bits, that this VI uses to represent the input signal. The valid range is [1, 32]. The default is 16.
- **output word length** specifies the word length, in number of bits, that this VI uses to represent the output signal. The valid range is [1, 32]. The default is 16.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **output rounding mode** specifies the <u>rounding mode</u> this VI uses in the output quantizer.

0 **Nearest**—Rounds to the closest representable number.

1 **Truncation** (default)—Rounds to the closest representable number less than the original value.

internal precision specifies the precision of the internal signals.

- 0 **Truncated** (default)—Approximates a full-precision result and helps spare logical resources of the FPGA hardware target. This option is valid for all finite impulse response (FIR) multirate filters and <u>cascaded integrator comb (CIC) decimation filters</u>.
- 1 **Full**—Provides a full-precision result. You must use this option if the filter is a moving average (MA) multirate filter. Otherwise the simulation cannot return the same results as the code generation results.
- **multirate filters out** returns the fixed-point multistage multirate filters.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.

- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Fixed-Point Multistage Multirate Filter Modeling and Simulation VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate directory for an example of using the DFD FXP NStage MRate Modeling VI.

■ Open example ■ Browse related examples

DFD FXP NStage MRate Postprocessing VI

Owning Palette: <u>Multirate Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Converts the output signal of a fixed-point multistage multirate filter from a fixed-point representation to a floating-point representation. You must manually select the polymorphic instance you want to use.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD FXP NStage MRate Postprocessing (I32, nCh)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- # channels specifies the number of channels that signal in contains. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **multirate filters** specifies the input multistage multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **history in** specifies the data from the last iteration of the
postprocessing process.

- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **history out** returns the remaining data for the next iteration of postprocessing. You can wire this output to the **history in** input of the next call to this VI if you want to process the data continuously.

DFD FXP NStage MRate Postprocessing (I16, nCh)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- # channels specifies the number of channels that signal in contains. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **multirate filters** specifies the input multistage multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **history in** specifies the history data from the last iteration of

postprocessing.

- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **history out** returns the remaining data for the next iteration of postprocessing. You can wire this output to the **history in** input of the next call to this VI if you want to process the data continuously.

DFD FXP NStage MRate Postprocessing (I32, 1Ch)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **multirate filters** specifies the input multistage multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP NStage MRate Postprocessing (I16, 1Ch)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **signal in** specifies the input signal that you want to process.
- **multirate filters** specifies the input multistage multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns a floating-point signal after postprocessing signal in.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP NStage MRate Quantization VI

Owning Palette: Multirate Fixed-Point Tools VIs

Installed With: Digital Filter Design Toolkit

Quantizes the coefficients of a floating-point multistage multirate filter and generates a fixed-point multistage multirate filter. This VI has the same internal settings as the <u>DFD FXP MRate Quantization</u> VI.

Examples



■ Place on the block diagram ■ Find on the **Functions** palette

- **Scale by power of 2?** specifies whether this VI scales the multirate filter with the original norm value or the smallest power of 2 value that is greater than the norm value. If **scale by power of 2?** is TRUE, this VI scales the multirate filter with the smallest power of 2 value. If **scale by power of 2?** is FALSE, this VI scales the multirate filter with the original norm value. The default is FALSE.
- **scale type** specifies how to scale the multirate filter coefficients. Refer to the <u>Details</u> section of the DFD Scale Filter VI for more information about each scale type.

1 **Time Domain 1-Norm** (default)

2 **Time Domain 2-Norm**

3 **Time Domain Inf-Norm**

- **multirate filters in** specifies the input multistage multirate filter.
- **coefficients word lengths** specifies the <u>word length</u>, in number of bits, that this VI uses to represent the coefficients of each stage in the input multirate filter. The valid range is [1, 32]. The default is 16.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **gain settings** specifies the settings for the multirate filter gain.
 - **gain processing** specifies whether you want to process the multirate filter gain on a host machine or an NI Reconfigurable I/O (RIO) target.
 - 0 **On Target**—Specifies to process the multirate filter gain on an NI-RIO target.
 - 1 **On Host** (default)—Specifies to process the multirate filter gain on a host machine.
 - **gain word length** specifies the word length, in number of bits, that this VI uses to represent the multirate filter gain if you set **gain processing** to On Target. The valid range is [1, 32]. The default is 16.
- **multirate filters out** returns the fixed-point multistage multirate filters.

- **postprocessing gain** returns the filter gain that this VI scaled.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD FXP NStage MRate Quantization VI:

• How to Build Coefficients Quantizer VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

■ Open example ■ Browse related examples

• Analyze Coefficients-Quantized Multistage Multirate Filter VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate

Open example Browse related examples

DFD FXP NStage MRate Simulation VI

Owning Palette: <u>Multirate Fixed-Point Tools VIs</u>

Installed With: Digital Filter Design Toolkit

Simulates the filtering process of a fixed-point multistage multirate filter continuously. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

•

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD FXP NStage MRate Simulation (DBL)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **init?** specifies how you want to initialize the internal states. The default is TRUE, which specifies that this VI initializes the internal states to zero. If **init?** is FALSE, this VI initializes the internal states from the final states of the previous call to the current VI instance. To process a large data sequence, split the sequence into smaller blocks, set **init?** to TRUE for the first block, and set **init?** to FALSE for the remaining blocks.
- **signal in** specifies the input signal you want to process. The **input word length** value you set on the <u>DVD FXP NStage MRate</u> <u>Modeling</u> VI determines the range of **signal in**. The range equals [-2^(**input word length**-1), 2^(**input word length**-1)-1]. For example, if you specify 16 as the **input word length** value, the corresponding range is [-32768, 32767].
- **multirate filters** specifies the input multistage multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DBL signal out** returns the output floating-point signal after postprocessing.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP NStage MRate Simulation (I32)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **init?** specifies how you want to initialize the internal states. The default is TRUE, which specifies that this VI initializes the internal states to zero. If **init?** is FALSE, this VI initializes the internal states from the final states of the previous call to the current VI instance. To process a large data sequence, split the sequence into smaller blocks, set **init?** to TRUE for the first block, and set **init?** to FALSE for the remaining blocks.
- **signal in** specifies the input signal you want to process. The **input word length** value you set on the <u>DVD FXP NStage MRate</u> <u>Modeling</u> VI determines the range of **signal in**. The range equals [-2^(**input word length**-1), 2^(**input word length**-1)-1]. For example, if you specify 16 as the **input word length** value, the corresponding range is [-32768, 32767].
- **multirate filters** specifies the input multistage multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DBL signal out** returns the output floating-point signal after postprocessing.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD FXP NStage MRate Simulation (I16)



- **input range** specifies the maximum absolute value of the input signal that the fixed-point integer can represent. For example, the input ranges both are 10 for DAQ devices with ranges of [0, 10V] and [–10, 10V]. The default is 1.
- **init?** specifies how you want to initialize the internal states. The default is TRUE, which specifies that this VI initializes the internal states to zero. If **init?** is FALSE, this VI initializes the internal states from the final states of the previous call to the current VI instance. To process a large data sequence, split the sequence into smaller blocks, set **init?** to TRUE for the first block, and set **init?** to FALSE for the remaining blocks.
- **signal in** specifies the input signal you want to process. The **input word length** value you set on the <u>DVD FXP NStage MRate</u> <u>Modeling</u> VI determines the range of **signal in**. The range equals [-2^(**input word length**-1), 2^(**input word length**-1)-1]. For example, if you specify 16 as the **input word length** value, the corresponding range is [-32768, 32767].
- **multirate filters** specifies the input multistage multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **DBL signal out** returns the output floating-point signal after postprocessing.
- **integer signal out** returns the output fixed-point integer signal, which is the same as the output signal from a fixed-point target.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Fixed-Point Multistage Multirate Filter Modeling and Simulation VI in the labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate directory for an example of using the DFD FXP NStage MRate Simulation VI.

■ Open example ■ Browse related examples

Multirate Processing VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Multirate Processing VIs to filter signals with multirate digital filters.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
DFD MRate Filtering for Single Block	Filters a single-block signal with a multirate filter. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.
DFD MRate Filtering with State	Filters a signal with a multirate filter. You must specify the initial internal states in state in to filter samples accurately. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.
<u>DFD</u> <u>MRate</u> Filtering	Filters a signal continuously with a multirate filter. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.
DFD NStage MRate Filtering for Single Block	Filters a single-block signal with a multistage multirate filter. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.
DFD NStage <u>MRate</u> Filtering with State	Filters a signal with a multistage multirate filter. You must specify the initial internal states in state in to filter the samples accurately. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually</u> <u>select</u> the instance.

DFD	Filters a signal continuously with a multistage multirate filter.
<u>NStage</u>	Wire data to the signal in input to determine the
<u>MRate</u>	polymorphic instance to use or <u>manually select</u> the instance.
Filtering	

DFD MRate Filtering VI

Owning Palette: <u>Multirate Processing VIs</u>

Installed With: Digital Filter Design Toolkit

Filters a signal continuously with a multirate filter. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

•

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD MRate Filtering (Array)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** specifies the input array of single-channel samples you want to process.
- multirate filter specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether this VI uses the multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. This input is valid only if you set **init?** to TRUE or if you

run this VI for the first time. The default is TRUE. You can use a multirate filter as a zero-phase filter only if it is an even-order filter.

signal out returns an array of filtered single-channel samples.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD MRate Filtering (Wfm)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** specifies the input waveform you want to process.
- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether this VI uses the multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. This input is valid only if you set **init?** to TRUE or if you run this VI for the first time. The default is TRUE. You can use a

multirate filter as a zero-phase filter only if it is an even-order filter.

- **signal out** returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD MRate Filtering (Array NChan)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** is the input array of multiple-channel samples you want to process. Each element in the first dimension of the array corresponds to a channel. Each element in the second dimension of the array corresponds to a sample from each channel.
- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - code is the error or warning code. The default is 0. If status is TRUE, code is a nonzero error code. If status is FALSE, code is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- zero phase? specifies whether this VI uses the multirate filter as a

zero-phase filter, in which no delay occurs between **signal in** and **signal out**. This input is valid only if you set **init?** to TRUE or if you run this VI for the first time. The default is TRUE. You can use a multirate filter as a zero-phase filter only if it is an even-order filter.

- **signal out** returns an array of filtered multiple-channel samples. Each element in the first dimension of the array corresponds to a channel. Each element in the second dimension of the array corresponds to a sample from each channel.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD MRate Filtering (Wfm NChan)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- signal in is the input array of multiple-channel waveforms you want to process.
- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether this VI uses the multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. This input is valid only if you set **init?** to TRUE or if you

run this VI for the first time. The default is TRUE. You can use a multirate filter as a zero-phase filter only if it is an even-order filter.

signal out returns an array of filtered multiple-channel waveforms.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Single Stage Multirate Filter Design VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Multirate directory for an example of using the DFD MRate Filtering VI.

■ Open example ■ Browse related examples

DFD MRate Filtering for Single Block VI

Owning Palette: <u>Multirate Processing VIs</u>

Installed With: Digital Filter Design Toolkit

Filters a single-block signal with a multirate filter. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

•

Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD MRate Filtering for Single Block (Array)



- **signal in** specifies the input array of single-channel samples you want to process.
- **multirate filter** specifies the input multirate filter.
- extension type specifies the method this VI uses to pad data at the beginning and end of **signal in** to lessen artificial jumps.
 - 0 **Zero padding** (default)—Uses zeroes to pad the input data. Watch for abrupt transitions between the padded zeroes and the input data, which causes large artifacts near the transition.
 - 1 **Symmetric**—Uses replications of the input data to pad the data, except that this VI flips the replications so that there are mirroring symmetries at the beginning and at the end of the input data.
 - 2 **Periodic**—Adds a replication of the input data block at the beginning of the input data block and adds another replication at the end to pad the data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether this VI uses the multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. The default is TRUE. You can use a multirate filter as a zero-phase filter only if it is an even-order filter.
- **signal out** returns an array of filtered single-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD MRate Filtering for Single Block (Wfm)



- **signal in** specifies the input waveform you want to process.
- **multirate filter** specifies the input multirate filter.
- extension type specifies the method this VI uses to pad data at the beginning and end of **signal in** to lessen artificial jumps.
 - 0 **Zero padding** (default)—Uses zeroes to pad the input data. Watch for abrupt transitions between the padded zeroes and the input data, which causes large artifacts near the transition.
 - 1 **Symmetric**—Uses replications of the input data to pad the data, except that this VI flips the replications so that there are mirroring symmetries at the beginning and at the end of the input data.
 - 2 **Periodic**—Adds a replication of the input data block at the beginning of the input data block and adds another replication at the end to pad the data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**

is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether this VI uses the multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. The default is TRUE. You can use a multirate filter as a zero-phase filter only if it is an even-order filter.
- **signal out** returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
DFD MRate Filtering for Single Block (Array NChan)



- **signal in** is the input array of multiple-channel samples you want to process. Each element in the first dimension of the array corresponds to a channel. Each element in the second dimension of the array corresponds to a sample from each channel.
- **multirate filter** specifies the input multirate filter.
- extension type specifies the method this VI uses to pad data at the beginning and end of **signal in** to lessen artificial jumps.
 - 0 **Zero padding** (default)—Uses zeroes to pad the input data. Watch for abrupt transitions between the padded zeroes and the input data, which causes large artifacts near the transition.
 - 1 **Symmetric**—Uses replications of the input data to pad the data, except that this VI flips the replications so that there are mirroring symmetries at the beginning and at the end of the input data.
 - 2 **Periodic**—Adds a replication of the input data block at the beginning of the input data block and adds another replication at the end to pad the data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether this VI uses the multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. The default is TRUE. You can use a multirate filter as a zero-phase filter only if it is an even-order filter.
- **signal out** returns an array of filtered multiple-channel samples. Each element in the first dimension of the array corresponds to a channel. Each element in the second dimension of the array corresponds to a sample from each channel.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD MRate Filtering for Single Block (Wfm NChan)



- **signal in** is the input array of multiple-channel waveforms you want to process.
- **multirate filter** specifies the input multirate filter.

extension type specifies the method this VI uses to pad data at the beginning and end of **signal in** to lessen artificial jumps.

- 0 **Zero padding** (default)—Uses zeroes to pad the input data. Watch for abrupt transitions between the padded zeroes and the input data, which causes large artifacts near the transition.
- 1 **Symmetric**—Uses replications of the input data to pad the data, except that this VI flips the replications so that there are mirroring symmetries at the beginning and at the end of the input data.
- 2 **Periodic**—Adds a replication of the input data block at the beginning of the input data block and adds another replication at the end to pad the data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether this VI uses the multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. The default is TRUE. You can use a multirate filter as a zero-phase filter only if it is an even-order filter.
- **signal out** returns an array of filtered multiple-channel waveforms.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - code is the error or warning code. If status is TRUE, code is a nonzero error code. If status is FALSE, code is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD MRate Filtering for Single Block VI:

• Interpolation Filtering for Single Block VI: labview\examples\Digital Filter Design\Floating-Point Filters\Multirate

Open example Browse related examples

• Multirate Filtering VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

Open example Browse related examples

• Decimation Filtering for Single Block VI: labview\examples\Digital Filter Design\Floating-Point Filters\Multirate

■ Open example ■ Browse related examples

DFD MRate Filtering with State VI

Owning Palette: <u>Multirate Processing VIs</u>

Installed With: Digital Filter Design Toolkit

Filters a signal with a multirate filter. You must specify the initial internal states in **state in** to filter samples accurately. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD MRate Filtering with State (Array)



signal in specifies the input array of single-channel samples you want to process.

- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **signal out** returns an array of filtered single-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD MRate Filtering with State (Wfm)



- **signal in** specifies the input waveform you want to process.
- **multirate filter** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **signal out** returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD NStage MRate Filtering VI

Owning Palette: <u>Multirate Processing VIs</u>

Installed With: Digital Filter Design Toolkit

Filters a signal continuously with a multistage multirate filter. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

▼

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD NStage MRate Filtering (Array)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** specifies the input array of single-channel samples you want to process.
- **multirate filters** specifies the input multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether this VI uses the multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. This input is valid only if you set **init?** to TRUE or if you

run this VI for the first time. The default is TRUE. You can use a multistage multirate filter as a zero-phase filter only if every stage is an even-order filter.

- **signal out** returns an array of filtered single-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD NStage MRate Filtering (Wfm)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** specifies the input waveform you want to process.
- **multirate filters** specifies the input multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether this VI uses the multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. This input is valid only if you set **init?** to TRUE or if you run this VI for the first time. The default is TRUE. You can use a

multistage multirate filter as a zero-phase filter only if every stage is an even-order filter.

- **signal out** returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Multistage Multirate Filter Continuous Processing VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Multirate directory for an example of using the DFD NStage MRate Filtering VI.

■ Open example ■ Browse related examples

DFD NStage MRate Filtering for Single Block VI

Owning Palette: <u>Multirate Processing VIs</u>

Installed With: Digital Filter Design Toolkit

Filters a single-block signal with a multistage multirate filter. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

▼

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD NStage MRate Filtering for Single Block (Array)



- **signal in** specifies the input array of single-channel samples you want to process.
- **multirate filters** specifies the input multirate filters.
- extension type specifies the method this VI uses to pad data at the beginning and end of **signal in** to lessen artificial jumps.
 - 0 **Zero padding** (default)—Uses zeroes to pad the input data. Watch for abrupt transitions between the padded zeroes and the input data, which causes large artifacts near the transition.
 - 1 **Symmetric**—Uses replications of the input data to pad the data, except that this VI flips the replications so that there are mirroring symmetries at the beginning and at the end of the input data.
 - 2 **Periodic**—Adds a replication of the input data block at the beginning of the input data block and adds another replication at the end to pad the data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether the VI uses the multistage multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. The default is TRUE. You can use a multistage multirate filter as a zero-phase filter only if every stage is an even-order filter.
- **signal out** returns an array of filtered single-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD NStage MRate Filtering for Single Block (Wfm)



- **signal in** specifies the input waveform you want to process.
- **multirate filters** specifies the input multirate filters.
- extension type specifies the method this VI uses to pad data at the beginning and end of **signal in** to lessen artificial jumps.
 - 0 **Zero padding** (default)—Uses zeroes to pad the input data. Watch for abrupt transitions between the padded zeroes and the input data, which causes large artifacts near the transition.
 - 1 **Symmetric**—Uses replications of the input data to pad the data, except that this VI flips the replications so that there are mirroring symmetries at the beginning and at the end of the input data.
 - 2 **Periodic**—Adds a replication of the input data block at the beginning of the input data block and adds another replication at the end to pad the data.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**

is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **zero phase?** specifies whether the VI uses the multistage multirate filter as a zero-phase filter, in which no delay occurs between **signal in** and **signal out**. The default is TRUE. You can use a multistage multirate filter as a zero-phase filter only if every stage is an even-order filter.
- **signal out** returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD NStage MRate Filtering with State VI

Owning Palette: <u>Multirate Processing VIs</u>

Installed With: Digital Filter Design Toolkit

Filters a signal with a multistage multirate filter. You must specify the initial internal states in **state in** to filter the samples accurately. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD NStage MRate Filtering with State (Array)



signal in specifies the input array of single-channel samples you want to process.

multirate filters specifies the input multirate filters.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

state in specifies the initial internal states before processing.
 state specifies the internal states of one stage of the multirate filters.

- **signal out** returns an array of filtered single-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **Status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.
 - **state** returns the internal states of one stage of the **multirate filters** after filtering.

DFD NStage MRate Filtering with State (Wfm)



signal in specifies the input waveform you want to process.

- multirate filters specifies the input multirate filters.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- state in specifies the initial internal states before processing.
 state specifies the internal states of one stage of the multirate filters.
- signal out returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.
 - **state** returns the internal states of one stage of the **multirate filters** after filtering.

Multirate Utilities VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Multirate Utilities VIs to retrieve the multirate filter coefficients and parameters and to create multirate filters from the filter coefficients.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
<u>DFD Build</u> <u>CIC Filter</u>	Creates a <u>cascaded integrator comb (CIC) filter</u> .
<u>DFD Build</u> <u>MRate</u> <u>Filter</u>	Creates a multirate filter. You must <u>manually select the</u> polymorphic instance you want to use.
<u>DFD Get</u> <u>MRate CIC</u> Parameters	Retrieves the transfer function coefficients, sampling frequency conversion factor, filtering mode, and filter parameters of a <u>cascaded integrator comb (CIC) filter</u> .
<u>DFD Get</u> <u>MRate</u> <u>Filter</u> Parameters	Retrieves the transfer function, sampling frequency conversion factor, and filtering mode of a multirate filter.
<u>DFD Get</u> <u>MRate</u> <u>Filter</u> <u>Structure</u>	Retrieves the structure of a multirate filter. A multirate filter can have either a <u>finite impulse response (FIR)</u> or <u>cascaded</u> <u>integrator comb (CIC)</u> structure.
<u>DFD Load</u> from File	Retrieves a filter from a file. You must <u>manually select the</u> polymorphic instance you want to use.
DFD Save MRate to Text File	Saves a multirate filter to a text file in XML format.
<u>DFD Save</u> <u>to File</u>	Saves a filter into a file. Wire data to the filter in input to determine the polymorphic instance to use or <u>manually</u>

select the instance. Use the DFD Load from File VI to load the filter from the file.

DFD Build CIC Filter VI

Owning Palette: <u>Multirate Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a cascaded integrator comb (CIC) filter.

<u>Examples</u>



■ Place on the block diagram ■ Find on the **Functions** palette

filtering mode specifies the processing mode of the filter that this VI creates.

0 **No Rate Change**—Does not change the sampling frequency of a signal.

1 **Interpolation**—Increases the sampling frequency of a signal to a higher sampling frequency that differs from the original frequency by an integer value. <u>Interpolation</u> also is known as up-sampling.

2 **Decimation** (default)—Reduces the sampling frequency of a signal to a lower sampling frequency that differs from the original frequency by an integer value. <u>Decimation</u> also is known as down-sampling.

filter type specifies the <u>type of filter</u> that this VI creates.



- # stages specifies the number of stages in the CIC filter.
- **factor** specifies the sampling frequency conversion factor of the CIC filter.
- **differential delay** specifies the differential delay of the CIC filter in samples. The valid values include 1 and 2. The default is 1.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **CIC filter out** returns a new CIC multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD Build CIC Filter VI:

• CIC Filter Design VI: labview\examples\Digital Filter Design\Floating-Point Filters\Multirate

Open example Browse related examples

• Generate LabVIEW FPGA Code for Moving Average Filter VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate

Open example Browse related examples

• Create Multirate Filter VI: labview\examples\Digital Filter Design\Getting Started\Design Filters

Open example Browse related examples

DFD Build MRate Filter VI

Owning Palette: <u>Multirate Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a multirate filter. You must <u>manually select the polymorphic</u> <u>instance</u> you want to use.

•

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD Build MRate Filter from TF



- **h(n)** specifies the input transfer function this VI uses to create the filter.
- **factor** specifies the sampling frequency conversion factor this VI uses to create the filter.
- **filtering mode** specifies the processing mode of the filter that this VI creates.
 - 0 **No Rate Change**—Does not change the sampling frequency of a signal.
 - 1 **Interpolation**—Increases the sampling frequency of a signal to a higher sampling frequency that differs from the original frequency by an integer value. <u>Interpolation</u> also is known as up-sampling.
 - 2 **Decimation** (default)—Reduces the sampling frequency of a signal to a lower sampling frequency that differs from the original frequency by an integer value. <u>Decimation</u> also is known as down-sampling.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- multirate filter out returns a new multirate filter.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Build MRate Filter from FIR Filter



- **filter in** specifies the input finite impulse response (FIR) filter that this VI uses to create the multirate filter.
- **factor** specifies the sampling frequency conversion factor this VI uses to create the filter.
- **filtering mode** specifies the processing mode of the filter that this VI creates.
 - 0 **No Rate Change**—Does not change the sampling frequency of a signal.
 - 1 **Interpolation**—Increases the sampling frequency of a signal to a higher sampling frequency that differs from the original frequency by an integer value. <u>Interpolation</u> also is known as up-sampling.
 - 2 **Decimation** (default)—Reduces the sampling frequency of a signal to a lower sampling frequency that differs from the original frequency by an integer value. <u>Decimation</u> also is known as down-sampling.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- multirate filter out returns a new multirate filter.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
DFD Build MRate Filter from MRate Filter



- **multirate filter in** specifies the input multirate filter that includes the transfer function this VI uses to create the new multirate filter.
- **factor** specifies the sampling frequency conversion factor this VI uses to create the filter.
- **filtering mode** specifies the processing mode of the filter that this VI creates.
 - 0 **No Rate Change**—Does not change the sampling frequency of a signal.
 - 1 **Interpolation**—Increases the sampling frequency of a signal to a higher sampling frequency that differs from the original frequency by an integer value. <u>Interpolation</u> also is known as up-sampling.
 - 2 **Decimation** (default)—Reduces the sampling frequency of a signal to a lower sampling frequency that differs from the original frequency by an integer value. <u>Decimation</u> also is known as down-sampling.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- multirate filter out returns a new multirate filter.
- **error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Build MRate Filter from TF (Rational)



- **h(n)** specifies the input transfer function this VI uses to create the filter.
- **factor** contains the sampling frequency conversion factor of the multirate filter.
 - L contains the numerator factor of the rational resampling frequency conversion factor. The value of L must be an integer greater than zero. The value of L must not equal the value of M. The default is 8.
 - M contains the denominator factor of the rational resampling frequency conversion factor. The value of **M** must be an integer greater than zero. The value of **M** must not equal the value of **L**. The default is 3.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **multirate filter out** returns a new multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Build MRate Filter from FIR Filter (Rational)



- **filter in** specifies the input finite impulse response (FIR) filter that this VI uses to create the multirate filter.
- **factor** contains the sampling frequency conversion factor of the multirate filter.
 - L contains the numerator factor of the rational resampling frequency conversion factor. The value of L must be an integer greater than zero. The value of L must not equal the value of M. The default is 8.
 - M contains the denominator factor of the rational resampling frequency conversion factor. The value of **M** must be an integer greater than zero. The value of **M** must not equal the value of **L**. The default is 3.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **multirate filter out** returns a new multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Build MRate Filter from MRate Filter (Rational)



- **multirate filter in** specifies the input multirate filter that includes the transfer function this VI uses to create the new multirate filter.
- **factor** contains the sampling frequency conversion factor of the multirate filter.
 - L contains the numerator factor of the rational resampling frequency conversion factor. The value of L must be an integer greater than zero. The value of L must not equal the value of M. The default is 8.
 - M contains the denominator factor of the rational resampling frequency conversion factor. The value of **M** must be an integer greater than zero. The value of **M** must not equal the value of **L**. The default is 3.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced

the error or warning. The default is an empty string.

- **multirate filter out** returns a new multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Create Multirate Filter VI in the labview\examples\Digital Filter Design\Getting Started\Design Filters directory for an example of using the DFD Build MRate Filter VI.

■ Open example ■ Browse related examples

DFD Get MRate CIC Parameters VI

Owning Palette: <u>Multirate Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Retrieves the transfer function coefficients, sampling frequency conversion factor, filtering mode, and filter parameters of a <u>cascaded</u> <u>integrator comb (CIC) filter</u>.



■ Place on the block diagram ■ Find on the **Functions** palette

- multirate filter in specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **h(n)** returns the equivalent finite impulse response (FIR) coefficients of the CIC filter.

- **CIC parameters** returns the CIC filter parameters.
 - # stages returns the number of stages in the CIC filter.
 - **differential delay** returns the differential delay of the CIC filter in sample times.
 - **filter type** returns the passband of the filter.
- **factor** returns the sampling frequency conversion factor of the multirate filter.
- **filtering mode** returns the filtering mode of the multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Get MRate Filter Parameters VI

Owning Palette: Multirate Utilities VIs

Installed With: Digital Filter Design Toolkit

Retrieves the transfer function, sampling frequency conversion factor, and filtering mode of a multirate filter.



 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

type specifies the transfer function that this VI retrieves.

- 0 **FLP** (default)—Specifies to retrieve the transfer function of a floating-point multirate filter. If **multirate filter in** is a fixed-point multirate filter, this VI retrieves the transfer function of the reference floating-point multirate filter.
- 1 **FXP**—Specifies to retrieve the transfer function of a fixed-point multirate filter. If **multirate filter in** is a floating-point multirate filter, this VI returns an error.
- multirate filter in specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **h(n)** returns the transfer function coefficients of the multirate filter.
- **factor** contains the sampling frequency conversion factor of the multirate filter.
 - L contains the numerator factor of the rational resampling frequency conversion factor. L contains 1 for no-ratechange or decimation filters.
 - M contains the denominator factor of the rational resampling frequency conversion factor. **M** contains 1 for no-rate-change or interpolation filters.
- **filtering mode** returns the filtering mode of the multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Get MRate Filter Structure VI

Owning Palette: Multirate Utilities VIs

Installed With: Digital Filter Design Toolkit

Retrieves the structure of a multirate filter. A multirate filter can have either a <u>finite impulse response (FIR)</u> or <u>cascaded integrator comb (CIC)</u> structure.



 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

- **multirate filter in** specifies the input multirate filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- multirate filter out returns the multirate filter in unchanged.
- **structure** returns the structure of the multirate filter.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Load from File VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Retrieves a filter from a file. You must <u>manually select the polymorphic</u> <u>instance</u> you want to use.

▼

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD Load Filter from File



- **file path in** specifies the path to the file. If **file path in** is empty (default) or <Not A Path>, this VI displays the **Select File Path** dialog box from which you can select a file. This VI returns an error if you specify a file path that does not exist or if you click the **Cancel** button on the dialog box.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the filter you loaded from the text file.
- **file path out** returns the path to the file to which this VI saved the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Load MRate Filter from File



- **file path in** specifies the path to the file. If **file path in** is empty (default) or <Not A Path>, this VI displays the **Select File Path** dialog box from which you can select a file. This VI returns an error if you specify a file path that does not exist or if you click the **Cancel** button on the dialog box.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- multirate filter out is the newly loaded multirate filter.
- **file path out** returns the path to the file to which this VI saved the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Save and Load Filter VI in the labview\examples\Digital Filter Design\Getting Started\Apply Filters directory for an example of using the DFD Load from File VI.

■ Open example ■ Browse related examples

DFD Save MRate to Text File VI

Owning Palette: <u>Multirate Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Saves a multirate filter to a text file in XML format.



Note This VI uses periods for decimal points in the coefficients values.

Details Examples



■ Place on the block diagram ■ Find on the **Functions** palette

- **multirate filter in** specifies the input multirate filter. You cannot specify a <u>multistage multirate filter</u> for this input.
- **File path in** specifies the path to the file in which you want to save the multirate filter. The default file extension is .xml. You can use any other file extension for the file. If **file path in** is empty (default) or <Not A Path>, this VI displays a dialog box from which you can select a file. This VI returns an error if you specify a file path that does not exist or if you click the **Cancel** button on the dialog box.
- **comments** specifies the comments that you want to add to the text file.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- multirate filter out returns the multirate filter in unchanged.
- **file path out** returns the path to the file to which this VI saved the multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Save MRate to Text File Details

This VI saves a multirate filter as an .xml file by default. The following table lists the tags that the .xml file might contain.

Tag Name	Description
DFDTData	Contains the entire information about a multirate filter.
CustomerComments	Contains the information you specified in the comments input of this VI.
Structure	Contains the multirate filter structure.
FilteringMode	Contains the filtering mode setting of the multirate filter.
Factor	Contains the sampling frequency conversion factor of the multirate filter. This tag contains the following sub-tags:
	 L—Contains the value of the interpolation factor.
	 M—Contains the value of the decimation factor.
CICParameters	Contains the filter parameters of the cascaded integrator comb (CIC) filter. If the filter is not a CIC filter, this XML file does not contain this tag. If the filter is a CIC filter, this tag contains the following sub-tags:
	 NumberOfStages—Contains the number of stages of the CIC filter.
	 DifferentialDelay—Contains the differential delay value of the CIC filter. FilterType—Contains the filter type of the CIC filter.
Order	Contains the multirate filter order.
Coefficients	Contains the floating-point multirate filter coefficients. If the filter is a fixed-point multirate filter, this tag contains the coefficients of the

	reference floating-point multirate filter.
FXPInformation	Contains the entire information about the fixed- point multirate filter. If the filter is a floating-point multirate filter, this XML file does not contain this tag. If the filter is a fixed-point multirate filter, this tag can contain the following sub-tags:
	 FXPCoefficients—Contains the fixed- point multirate filter coefficients. FXPGain—Contains the fixed-point
	multirate filter gain.
	 IntegerCoefficients—Contains the integer multirate filter coefficients.
	 IntegerGain—Contains the integer multirate filter gain.
	• Quantizers —Contains the multirate filter coefficients quantizer settings and multirate filter gain quantizer settings. This tag contains the following sub-tags:
	 CoefficientsQuantizer— Contains the multirate filter coefficients quantizer settings.
	- ScaleByPowerOf2—Indicates the scaling setting of the multirate filter. This tag appears only if you set the scale by power of 2? input of the DFD FXP MRate Quantization VI to TRUE.
	 ScaleType—Contains the scale type setting of the multirate filter. GainQuantizer—Contains the multirate filter gain quantizer settings.
	The CoefficientsQuantizer and GainQuantizer tags can contain the following sub-tags:
	- WordLength—Contains the word

 length of the quantizer. IntegerWordLength—Contains the integer word length of the quantizer.
 OverflowMode—Contains the overflow mode setting of the quantizer.
 RoundingMode—Contains the rounding mode setting of the quantizer.
 Signed—Indicates the quantizer uses a signed fixed-point format. This tag is always empty.
 Unsigned—Indicates the quantizer uses an unsigned fixed- point format. This tag is always empty.
Note If the gain processing occurs on a host machine, or if the gain processing occurs on a target but the filter gain is a power-of-2 value, the multirate filter gain quantizer contains the following message only: No integer gain is processed on the target .

Examples

Refer to the following VIs for examples of using the DFD Save MRate to Text File VI:

• Export Multirate FIR Coef to Xilinx COE File VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Multirate

Open example Browse related examples

• Save Multirate Filter to Text File VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

■ Open example ■ Browse related examples

DFD Save to File VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Saves a filter into a file. Wire data to the **filter in** input to determine the polymorphic instance to use or <u>manually select</u> the instance. Use the <u>DFD Load from File</u> VI to load the filter from the file.

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

▼

DFD Save Filter to File



filter in specifies the input filter.

- **File path in** specifies the path to the file in which you want to save the filter. The default file extension is .fds. You can use any other file extension for the file. If **file path in** is empty (default) or <Not A Path>, this VI displays a dialog box from which you can select a file. This VI returns an error if you specify a file path that does not exist or if you click the **Cancel** button on the dialog box. Use the DFD Load from File VI to load the filter from the file.
- **start path** is the path name to the initially displayed directory or folder in the file dialog. The default value is **Not A Path**, which is the path to the last directory or folder shown in the file dialog.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **filter out** returns the **filter in** unchanged.
- **file path out** returns the path to the file to which this VI saved the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Save MRate Filter to File

multirate filter in multirate filter out file path in file path out start path (Not A Path)

multirate filter in specifies the input multirate filter. You cannot specify a <u>multistage multirate filter</u> for this input.

- **File path in** specifies the path to the file in which you want to save the multirate filter. The default file extension is .mfs. You can use any other file extension for the multirate file. If **file path in** is empty (default) or <Not A Path>, this VI displays a dialog box from which you can select a file. This VI returns an error if you specify a file path that does not exist or if you click the **Cancel** button on the dialog box. Use the <u>DFD Load from File</u> VI to load the multirate filter from the file.
- **start path** is the path name to the initially displayed directory or folder in the file dialog. The default value is **Not A Path**, which is the path to the last directory or folder shown in the file dialog.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in

most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- multirate filter out returns the multirate filter in unchanged.
- **file path out** returns the path to the file to which this VI saved the multirate filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Save and Load Filter VI in the labview\examples\Digital Filter Design\Getting Started\Apply Filters directory for an example of using the DFD Save to File VI.

■ Open example ■ Browse related examples

Processing VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Processing VIs to filter signals with digital filters.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
<u>DFD</u> Filtering with State	Filters an input signal. You must specify the initial internal states in the state in input to generate an accurate output. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.
<u>DFD</u> Filtering	Filters an input signal continuously. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.
<u>DFD Integer</u> <u>Delay with</u> <u>State</u>	Delays a signal by a specified integer number of sample intervals. You must specify the initial internal states in the state in input to generate an accurate output. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.
<u>DFD Integer</u> Delay	Delays a signal continuously by a specified integer number of sample intervals. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually</u> <u>select</u> the instance.
<u>DFD</u> Narrowband Filtering with State	Filters a signal with a narrowband filter. You must specify the initial internal states in state in to generate an accurate output. Wire data to the signal in input to determine the polymorphic instance to use or <u>manually select</u> the instance.
DFD Narrowband	Filters a signal continuously with a narrowband filter. Wire data to the signal in input to determine the polymorphic

Filtering instance to use or manually select the instance.

DFD Filtering VI

Owning Palette: Processing VIs

Installed With: Digital Filter Design Toolkit

Filters an input signal continuously. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

Examples

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

◄
DFD Filtering (PtByPt)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** is the input sample you want to process.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns the filtered sample.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Filtering (Array)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** specifies the input array of single-channel samples you want to process.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns an array of filtered single-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Filtering (Wfm)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** specifies the input waveform you want to process.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Filtering (PtByPt NChan)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** is the input array of multiple-channel samples you want to process.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns an array of filtered multiple-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Filtering (Array NChan)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** is the input array of multiple-channel samples you want to process. Each element in the first dimension of the array corresponds to a channel. Each element in the second dimension of the array corresponds to a sample from each channel.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns an array of filtered multiple-channel samples.

Each element in the first dimension of the array corresponds to a channel. Each element in the second dimension of the array corresponds to a sample from each channel.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Filtering (Wfm NChan)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** is the input array of multiple-channel waveforms you want to process.
- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns an array of filtered multiple-channel waveforms.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the

same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **Source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Examples

Refer to the following VIs for examples of using the DFD Filtering VI:

 Online Filtering - DFD VI: labview\examples\Digital Filter Design\AALXMPL

■ Open example ■ Browse related examples

Train Wheel PtByPt - DFD VI: labview\examples\Digital Filter
Design\AALXMPL

Open example Browse related examples

• Filtering VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

■ Open example ■ Browse related examples

DFD Filtering with State VI

Owning Palette: Processing VIs

Installed With: Digital Filter Design Toolkit

Filters an input signal. You must specify the initial internal states in the **state in** input to generate an accurate output. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD Filtering with State (PtByPt)



signal in is the input sample you want to process.

filter specifies the input filter.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **signal out** returns the filtered sample.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD Filtering with State (Array)



signal in specifies the input array of single-channel samples you want to process.

- **filter** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **signal out** returns an array of filtered single-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu

for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD Filtering with State (Wfm)



signal in specifies the input waveform you want to process.

filter specifies the input filter.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **signal out** returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD Integer Delay VI

Owning Palette: Processing VIs

Installed With: Digital Filter Design Toolkit

Delays a signal continuously by a specified integer number of sample intervals. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

•

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD Integer Delay (PtByPt)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** is the input sample you want to process.
- **delay** specifies the number of samples by which you want to delay the input signal. The default is 1. The value of **delay** must be greater than or equal to zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns the delayed sample.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Integer Delay (Array)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** specifies the input array of single-channel samples you want to process.
- **delay** specifies the number of samples by which you want to delay the input signal. The default is 1. The value of **delay** must be greater than or equal to zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns an array of delayed single-channel samples.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Integer Delay (Wfm)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** specifies the input waveform you want to process.
- **delay** specifies the number of samples by which you want to delay the input signal. The default is 1. The value of **delay** must be greater than or equal to zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** contains the delayed waveform.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Integer Delay (PtByPt NChan)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** is the input array of multiple-channel samples you want to process.
- **delay** specifies the number of samples by which you want to delay the input signal. The default is 1. The value of **delay** must be greater than or equal to zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns an array of delayed multiple-channel samples.

Each element in the first dimension of the array corresponds to a channel. Each element in the second dimension of the array corresponds to a sample from each channel.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Integer Delay (Array NChan)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- **signal in** is the input array of multiple-channel samples you want to process. Each element in the first dimension of the array corresponds to a channel. Each element in the second dimension of the array corresponds to a sample from each channel.
- **delay** specifies the number of samples by which you want to delay the input signal. The default is 1. The value of **delay** must be greater than or equal to zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

- **signal out** returns an array of delayed multiple-channel samples. Each element in the first dimension of the array corresponds to a channel. Each element in the second dimension of the array corresponds to a sample from each channel.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Integer Delay (Wfm NChan)



- **init?** controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.
- signal in is the input array of multiple-channel waveforms you want to process.
- **delay** specifies the number of samples by which you want to delay the input signal. The default is 1. The value of **delay** must be greater than or equal to zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns an array of delayed multiple-channel

waveforms.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Delay Signal VI in the labview\examples\Digital Filter Design\Getting Started\Apply Filters directory for an example of using the DFD Integer Delay VI.

Open example Browse related examples

DFD Integer Delay with State VI

Owning Palette: Processing VIs

Installed With: Digital Filter Design Toolkit

Delays a signal by a specified integer number of sample intervals. You must specify the initial internal states in the **state in** input to generate an accurate output. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

Use the pull-down menu to select an instance of this VI.

Select an instance

•

 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

DFD Integer Delay with State (PtByPt)



signal in is the input sample you want to process.

- **delay** specifies the number of samples by which you want to delay the input signal. The default is 1. The value of **delay** must be greater than or equal to zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **signal out** returns the delayed sample.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front
panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD Integer Delay with State (Array)



- **signal in** specifies the input array of single-channel samples you want to process.
- **delay** specifies the number of samples by which you want to delay the input signal. The default is 1. The value of **delay** must be greater than or equal to zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **signal out** returns an array of delayed single-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status

that this VI or function produces. Right-click the **error out** front panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD Integer Delay with State (Wfm)



- **signal in** specifies the input waveform you want to process.
- **delay** specifies the number of samples by which you want to delay the input signal. The default is 1. The value of **delay** must be greater than or equal to zero.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **state in** specifies the initial internal states before processing.
- **signal out** contains the delayed waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after processing. You can wire this output to the **state in** input of the next call to this VI if you want to process data continuously.

DFD Narrowband Filtering VI

Owning Palette: Processing VIs

Installed With: Digital Filter Design Toolkit

Filters a signal continuously with a narrowband filter. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually</u> <u>select</u> the instance.

•

Example

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD Narrowband Filtering (Array)



TF

init? controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.

- **signal in** specifies the input array of single-channel samples you want to process.
- **narrowband filter** contains the narrowband filter.
 - **multirate filters** contains the multirate filters this VI uses to construct the narrowband filter.
 - **filter type** contains the type of filter.

0	Lowpass	(default)
---	---------	-----------

1 Highpass

- 2 Bandpass
- 3 Bandstop
- 4 Wideband-Lowpass
- 5 Wideband-Highpass
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or

that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns an array of filtered single-channel samples.

error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Narrowband Filtering (Wfm)



TF

init? controls the initialization of internal states. The default is FALSE, in which this VI initializes internal states from the final states of the previous call to the current VI instance. If you select TRUE, this VI initializes internal states to zero.

- **signal in** specifies the input waveform you want to process.
- **narrowband filter** contains the narrowband filter.
 - **multirate filters** contains the multirate filters this VI uses to construct the narrowband filter.
 - filter type contains the type of filter.

0	Lowpass (default)
---	-------------------

- 1 Highpass
- 2 Bandpass
- 3 Bandstop
- 4 Wideband-Lowpass
- 5 Wideband-Highpass
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **signal out** returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Narrowband Filter Design and Processing VI in the labview\examples\Digital Filter Design\Floating-Point Filters\Multirate directory for an example of using the DFD Narrowband Filtering VI.

■ Open example ■ Browse related examples

DFD Narrowband Filtering with State VI

Owning Palette: Processing VIs

Installed With: Digital Filter Design Toolkit

Filters a signal with a narrowband filter. You must specify the initial internal states in **state in** to generate an accurate output. Wire data to the **signal in** input to determine the polymorphic instance to use or <u>manually select</u> the instance.

•

Use the pull-down menu to select an instance of this VI.

Select an instance

■ Place on the block diagram ■ Find on the **Functions** palette

DFD Narrowband Filtering with State (Array)



signal in specifies the input array of single-channel samples you want to process.

narrowband filter contains the narrowband filter.

- **multirate filters** contains the multirate filters this VI uses to construct the narrowband filter.
- filter type contains the type of filter.

0	Lowpass (default)	
1	Highpass	
2	Bandpass	
3	Bandstop	
4	Wideband-Lowpass	
5	Wideband-Highpass	

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE,

code is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- state in specifies the initial internal states before filtering.
 state specifies the internal states of one filter within narrowband filter.
- **signal out** returns an array of filtered single-channel samples.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after filtering. You can wire this output to the **state in** input of another call to this VI if you want to process data continuously.
 - **state** returns the internal states of one filter in **narrowband filter** after filtering.

DFD Narrowband Filtering with State (Wfm)



signal in specifies the input waveform you want to process.

narrowband filter contains the narrowband filter.

multirate filters contains the multirate filters this VI uses to construct the narrowband filter.

- **filter type** contains the type of filter.
 - 0 Lowpass (default)
 - 1 Highpass
 - 2 Bandpass
 - 3 Bandstop
 - 4 Wideband-Lowpass
 - 5 Wideband-Highpass
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- state in specifies the initial internal states before filtering.
 state specifies the internal states of one filter within narrowband filter.
- **signal out** returns the filtered waveform.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.
- **state out** returns the internal states after filtering. You can wire this output to the **state in** input of another call to this VI if you want to process data continuously.
 - **state** returns the internal states of one filter in **narrowband filter** after filtering.

Utilities VIs

Owning Palette: Digital Filter Design VIs and Functions

Installed With: Digital Filter Design Toolkit. This topic might not match its corresponding palette in LabVIEW depending on your operating system, licensed product(s), and target.

Use the Utilities VIs to retrieve the filter coefficients and and to create filters from the coefficients.

The VIs on this palette can return <u>general LabVIEW error codes</u> or <u>specific digital filter design error codes</u>.

Palette Object	Description
DFD Build Filter from Cascaded Coef	Converts the IIR Filter Cluster output of the <u>Advanced IIR</u> <u>Filtering</u> VIs into a filter.
DFD Build Filter from Lattice Coef	Creates a filter from lattice form coefficients.
DFD Build Filter from Transfer Function	Creates a filter from a transfer function.
<u>DFD Build</u> Filter from Zero-Pole- Gain	Builds a filter using poles, zeroes, and gain values.
<u>DFD Get</u> <u>Cascaded</u> <u>Coef</u>	Converts a filter to an infinite impulse response (IIR) filter cluster that is compatible with the IIR Filter Cluster output in the Advanced IIR Filtering VIs.
<u>DFD Get</u> Lattice Coef	Retrieves the lattice form coefficients of a filter. The filter you specify must have a lattice form filter structure.
DFD Get Order	Retrieves the order of a filter.
1	

<u>DFD Get</u> <u>Transfer</u> Function	Retrieves the transfer function of a filter.
<u>DFD Get</u> <u>Zero-Pole-</u> <u>Gain</u>	Retrieves the zeroes, poles, and gain (ZPK) of a filter.
<u>DFD Load</u> from File	Retrieves a filter from a file. You must <u>manually select the</u> polymorphic instance you want to use.
<u>DFD Load</u> from Text File	Retrieves a filter from a text file.
<u>DFD</u> <u>Render</u> <u>Difference</u> Equation	Draws the difference equation of a filter in a <u>picture</u> indicator.
DFD Render Transfer Function Equation	Draws the transfer function of a filter in a <u>picture indicator</u> .
<u>DFD</u> <u>Render</u> <u>Zero-Pole-</u> <u>Gain</u> Equation	Draws the zero-pole-gain equation of a filter in a <u>picture</u> indicator.
<u>DFD Save</u> <u>to File</u>	Saves a filter into a file. Wire data to the filter in input to determine the polymorphic instance to use or <u>manually</u> <u>select</u> the instance. Use the <u>DFD Load from File</u> VI to load the filter from the file.
<u>DFD Save</u> to Text File	Saves a filter to a text file in XML format.

DFD Build Filter from Cascaded Coef VI

Owning Palette: Utilities VIs

Installed With: Digital Filter Design Toolkit

Converts the **IIR Filter Cluster** output of the <u>Advanced IIR Filtering</u> VIs into a filter.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

- IIR Filter Cluster specifies the cascaded form of the IIR filter coefficients. This cluster is the output from one of the IIR coefficient design VIs: <u>Butterworth Coefficients</u>, <u>Bessel</u> <u>Coefficients</u>, <u>Chebyshev Coefficients</u>, <u>Elliptic Coefficients</u>, or <u>Inv</u> <u>Chebyshev Coefficients</u>.
 - **filter structure** selects IIR second-order or fourth-order filter stages.
 - **Reverse Coefficients** is the reverse coefficients of the IIR cascade filter.
 - **Forward Coefficients** is the forward coefficients of the IIR cascade filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The

default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- filter out returns a new filter.
- error out contains error information corresponding to the error code. Right-click the error out indicator on the front panel and select Explain Error or Explain Warning from the shortcut menu for more information about the error or warning.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Create IIR Filter from Cascaded Coefficients VI in the labview\examples\Digital Filter Design\Getting Started\Design Filters directory for an example of using the DFD Build Filter from Cascaded Coef VI.

■ Open example ■ Browse related examples

DFD Build Filter from Lattice Coef VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Creates a filter from lattice form coefficients.

<u>Example</u>



- \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette
- **Iattice coef** specifies the lattice form coefficients this VI uses to create the filter.
 - **structure type** specifies the <u>structure of the filter</u>.
 - 13 Lattice Allpass (basic sections)
 - **14** Lattice Allpass (one multiplier sections)
 - **15** Lattice Allpass (normalized sections)
 - 16 Lattice AR (basic sections)
 - **17** Lattice AR (one multiplier sections)
 - 18 Lattice AR (normalized sections)
 - 19 Lattice MA (minimum phase)
 - 20 Lattice MA (maximum phase)
 - 21 Lattice ARMA (basic sections) (default)
 - 22 Lattice ARMA (one multiplier sections)
 - 23 Lattice ARMA (normalized sections)
 - **reflection coefficients** specifies the lattice reflection coefficients of the filter.
 - **ladder coefficients** specifies the lattice ladder coefficients of the filter.
 - **gain** specifies the lattice gain.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before

this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns a new filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Create Filter from Lattice Coefficients VI in the labview\examples\Digital Filter Design\Getting Started\Design Filters directory for an example of using the DFD Build Filter from Lattice Coef VI.

■ Open example ■ Browse related examples

DFD Build Filter from Transfer Function VI

Owning Palette: Utilities VIs

Installed With: Digital Filter Design Toolkit

Creates a filter from a transfer function.

Details Examples



Place on the block diagram Find on the Functions palette

- **numerator** specifies the numerator polynomial of the transfer function in ascending order of z^{-1} .
- **denominator** specifies the denominator polynomial of the transfer function in ascending order of z^-1. The default is 1, which means this VI builds an FIR filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

filter out returns a new filter.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Build Filter from Transfer Function Details

The transfer function of a filter is defined by the following equation:

$$H(z) = \frac{\sum_{k=0}^{N} b[k] z^{-k}}{\sum_{k=0}^{M} a[k] z^{-k}} = \frac{b[0] + b[1] z^{-1} + \dots + b[N] z^{-N}}{a[0] + a[1] z^{-1} + \dots + a[N] z^{-N}}$$

where z is a complex variable

b is the set of coefficients of the **numerator** polynomial, also known as the forward coefficients

a is the set of coefficients of the **denominator** polynomial, also known as the reverse coefficients

N is the numerator order

M is the denominator order

The default value of **denominator** is 1.00, which means this VI builds an FIR filter.

Examples

Refer to the following VIs for examples of using the DFD Build Filter from Transfer Function VI:

• Build an Exponentially Weighted Moving Average Filter VI: labview\examples\Digital Filter Design\Getting Started\Design Filters

■ Open example ■ Browse related examples

• Create Filter from Transfer Function VI: labview\examples\Digital Filter Design\Getting Started\Design Filters

Open example Browse related examples

DFD Build Filter from Zero-Pole-Gain VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Builds a filter using poles, zeroes, and gain values.

Details Example



Place on the block diagram Find on the Functions palette

- **gain** specifies the gain this VI uses to create the filter in linear units.
- **zeroes** specifies the zeroes this VI uses to create the filter.
- **poles** specifies the poles this VI uses to create the filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.

filter out returns a new filter.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Build Filter from Zero-Pole-Gain Details

The filter that the **filter out** returns represents the transfer function H(z), expressed by

$$H(z) = Gain \cdot \frac{\prod_{k=1}^{N} (1 - z_k z^{-1})}{\prod_{k=1}^{M} (1 - \rho_k z^{-1})}$$

where z is a complex variable, z_k is a zero, and p_k is a pole. N is the numerator order and M is the denominator order.

The poles of a stable filter must remain inside the unit circle.

Example

Refer to the Create Filter from Zero-Pole-Gain VI in the labview\examples\Digital Filter Design\Getting Started\Design Filters directory for an example of using the DFD Build Filter from Zero-Pole-Gain VI.

■ Open example ■ Browse related examples

DFD Get Cascaded Coef VI

Owning Palette: Utilities VIs

Installed With: Digital Filter Design Toolkit

Converts a filter to an infinite impulse response (IIR) filter cluster that is compatible with the **IIR Filter Cluster** output in the <u>Advanced IIR Filtering</u> VIs.

You can wire the **IIR Filter Cluster** output of this VI to the **IIR Filter Cluster** input of the Advanced IIR Filtering VIs.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

type specifies the type of coefficients to retrieve.

0 **FLP** (default)—Specifies to retrieve the coefficients of a floating-point filter. If **filter in** is a fixed-point filter, this VI retrieves the coefficients of the reference floating-point filter.

1 **FXP**—Specifies to retrieve the coefficients of a fixed-point filter. If **filter in** is a floating-point filter, this VI returns an error.

filter in specifies the input filter.

error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI or

function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the **filter in** unchanged.
- IIR Filter Cluster returns the cascaded form of the IIR filter coefficients.
 - **filter structure** indicates either IIR second-order or IIR fourth-order filter stages.
 - **Reverse Coefficients** is the reverse coefficients of the IIR cascade filter.
 - **Forward Coefficients** is the forward coefficients of the IIR cascade filter.
- **gain** returns the gain of the filter in linear units.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Example

Refer to the Get Cascaded Coefficients of IIR Filter VI in the labview\examples\Digital Filter Design\Getting Started\Analyze Filters directory for an example of using the DFD Get Cascaded Coef VI.

■ Open example ■ Browse related examples

DFD Get Lattice Coef VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Retrieves the lattice form coefficients of a filter. The filter you specify must have a lattice form filter structure.

Example



 \blacksquare Place on the block diagram \blacksquare Find on the **Functions** palette

type specifies the type of coefficients to retrieve.

0 **FLP** (default)—Specifies to retrieve the coefficients of a floating-point filter. If **filter in** is a fixed-point filter, this VI retrieves the coefficients of the reference floating-point filter.

1 **FXP**—Specifies to retrieve the coefficients of a fixed-point filter. If **filter in** is a floating-point filter, this VI returns an error.

- **filter in** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status**
is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- filter out returns the filter in unchanged.
- **Iattice coef** returns the lattice form coefficients from the filter.
 - **structure type** returns the structure of the filter.
 - **reflection coefficients** returns the lattice reflection coefficients of the filter.
 - **ladder coefficients** returns the lattice ladder coefficients of the filter.
 - **gain** returns the lattice gain of the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Refer to the Get Lattice Coefficients of Filter VI in the labview\examples\Digital Filter Design\Getting Started\Analyze Filters directory for an example of using the DFD Get Lattice Coef VI.

DFD Get Order VI

Owning Palette: Utilities VIs

Installed With: Digital Filter Design Toolkit

Retrieves the order of a filter.



■ Place on the block diagram ■ Find on the **Functions** palette

- **filter in** specifies the input filter.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
 - **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the **filter in** unchanged.
- **order** returns the filter order. For infinite impulse response (IIR) filters, **order** is the larger of the numerator and denominator order values.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Get Transfer Function VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Retrieves the transfer function of a filter.

Examples



■ Place on the block diagram ■ Find on the **Functions** palette

type specifies the type of transfer function to retrieve.

0 **FLP** (default)—Specifies to retrieve the transfer function of a floating-point filter. If **filter in** is a fixed-point filter, this VI retrieves the transfer function of the reference floating-point filter.

1 **FXP**—Specifies to retrieve the transfer function of a fixed-point filter. If **filter in** is a floating-point filter, this VI returns an error.

filter in specifies the input filter.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the **filter in** unchanged.
- **numerator** returns the numerator polynomial of the transfer function in ascending order of z^{-1} .
- **denominator** returns the denominator polynomial of the transfer function in ascending order of z^-1.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Refer to the following VIs for examples of using the DFD Get Transfer Function VI:

• Get Integer Coef of Fixed-Point FIR Filter VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

■ Open example ■ Browse related examples

• Get Transfer Function of Filter VI: labview\examples\Digital Filter Design\Getting Started\Analyze Filters

DFD Get Zero-Pole-Gain VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Retrieves the zeroes, poles, and gain (ZPK) of a filter.

Example



■ Place on the block diagram ■ Find on the **Functions** palette

type specifies the type of zero-pole-gain (ZPK) coefficients to retrieve.

0 **FLP** (default)—Specifies to retrieve the coefficients of a floating-point filter. If **filter in** is a fixed-point filter, this VI retrieves the coefficients of the reference floating-point filter.

1 **FXP**—Specifies to retrieve the coefficients of a fixed-point filter. If **filter in** is a floating-point filter, this VI returns an error.

filter in specifies the input filter.

- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the Simple Error Handler or General Error Handler VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.
 - **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.

- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **gain** returns the gain of the filter in linear units.
- **filter out** returns the **filter in** unchanged.
- **zeroes** returns the zeroes of the filter.
- **poles** returns the poles of the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Refer to the Get Zero-Pole-Gain of Filter VI in the labview\examples\Digital Filter Design\Getting Started\Analyze Filters directory for an example of using the DFD Get Zero-Pole-Gain VI.

DFD Load from Text File VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Retrieves a filter from a text file.



Note This VI uses periods as the decimal points in the coefficients values.

<u>Example</u>

Place on the block diagram **I** Find on the **Functions** palette

load type specifies whether to load a filter from the coefficients or from the zeroes, poles, and gain.

0 **From Coefficients** (default)—Loads the filter from the filter coefficients.

- 1 **From Zero-Pole-Gain**—Loads the filter from the zeroes, poles, and gain. For a filter with a <u>finite impulse response (FIR)</u> symmetric or antisymmetric structure, if you saved the filter to a text file by using the DFD Save to Text File VI and then load the filter from the text file by using the DFD Load from Text File VI, the new filter coefficients might not be exactly the same as the original filter coefficients because of the numeric errors occurred during the conversion between the filter coefficients and zero-pole-gain values.
- **file path in** specifies the path to the file. If **file path in** is empty (default) or <Not A Path>, this VI displays the **Select File Path** dialog box from which you can select a file. This VI returns an error if you specify a file path that does not exist or if you click the **Cancel** button on the dialog box.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while

this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the filter you loaded from the text file.
- **file path out** returns the path to the file to which this VI saved the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Refer to the Save and Load Filter to and from Text File VI in the labview\examples\Digital Filter Design\Getting Started\Apply Filters directory for an example of using the DFD Load from Text File VI.

DFD Render Difference Equation VI

Owning Palette: Utilities VIs

Installed With: Digital Filter Design Toolkit

Draws the difference equation of a filter in a picture indicator.

<u>Example</u>



■ Place on the block diagram ■ Find on the **Functions** palette

coefficients format specifies how this VI <u>converts</u> the filter coefficients into a string.

type specifies the type of difference equation to draw.

0 **FLP** (default)—Specifies to draw the difference equation of a floating-point filter. If **filter in** is a fixed-point filter, this VI draws the difference equation of the reference floating-point filter.

1 **FXP**—Specifies to draw the difference equation of a fixed-point filter. If **filter in** is a floating-point filter, this VI returns an error.

- **filter in** specifies the input filter.
- **origin** specifies the upper-left-hand position of the difference equation this VI draws.
 - **left** specifies the horizontal coordinate that increases to the right. The default is 10.
 - **top** specifies the vertical coordinate that increases to the bottom. The default is 10.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error

status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **display format** specifies the format in which this VI displays the difference equation.
 - 0 **Feedback on left and feedforward on right** (default)— Displays the difference equation with the feedback on the left side of the equation and the feedforward on the right side of the equation.
 - 1 **Feedback and feedforward on right**—Displays the difference equation with both feedback and feedforward on the right side of the equation.
 - 2 **Feedback and feedforward on right in one line**—Displays the difference equation with both feedback and feedforward on the right side of the equation in one line.
- **filter out** returns the **filter in** unchanged.
- **difference equation** returns the difference equation of **filter in** in a picture indicator.
- **draw area size** returns the size of the area in the picture indicator this VI uses to draw the difference equation.
 - width returns the width of the draw area.
 - **height** returns the height of the draw area.

- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Refer to the Filter Rendering VI in the labview\examples\Digital Filter Design\Getting Started\Analyze Filters directory for an example of using the DFD Render Difference Equation VI.

DFD Render Transfer Function Equation VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Draws the transfer function of a filter in a picture indicator.

<u>Example</u>



■ Place on the block diagram ■ Find on the **Functions** palette

coefficients format specifies how this VI <u>converts</u> the filter coefficients into a string.

type specifies the type of transfer function to draw.

0 **FLP** (default)—Specifies to draw the transfer function of a floating-point filter. If **filter in** is a fixed-point filter, this VI draws the transfer function of the reference floating-point filter.

1 **FXP**—Specifies to draw the transfer function of a fixed-point filter. If **filter in** is a floating-point filter, this VI returns an error.

- **filter in** specifies the input filter.
- **origin** specifies the upper-left-hand position of the transfer function this VI draws.
 - **left** specifies the horizontal coordinate that increases to the right. The default is 10.
 - **top** specifies the vertical coordinate that increases to the bottom. The default is 10.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error

status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **display format** specifies the format in which this VI displays the transfer function.

0 **Expanded Form** (default)—Displays the transfer function in ascending order of power.

1 **Second-Order Sections**—Displays the transfer function in strict order with two as the highest order.

- **filter out** returns the **filter in** unchanged.
- transfer function returns the transfer function of filter in in a picture indicator.
- **draw area size** returns the size of the area in the picture indicator this VI uses to draw the transfer function.
 - width returns the width of the draw area.
 - **height** returns the height of the draw area.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Refer to the Filter Rendering VI in the labview\examples\Digital Filter Design\Getting Started\Analyze Filters directory for an example of using the DFD Render Transfer Function Equation VI.

DFD Render Zero-Pole-Gain Equation VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Draws the zero-pole-gain equation of a filter in a picture indicator.

<u>Example</u>



■ Place on the block diagram ■ Find on the **Functions** palette

coefficients format specifies how this VI <u>converts</u> the filter coefficients into a string.

type specifies the type of zero-pole-gain equation to draw.

0 **FLP** (default)—Specifies to draw the zero-pole-gain equation of a floating-point filter. If **filter in** is a fixed-point filter, this VI draws the zero-pole-gain equation of the reference floatingpoint filter.

1 **FXP**—Specifies to draw the zero-pole-gain equation of a fixedpoint filter. If **filter in** is a floating-point filter, this VI returns an error.

- **filter in** specifies the input filter.
- **origin** specifies the upper-left-hand position of the zero-pole-gain equation this VI draws.
 - **left** specifies the horizontal coordinate that increases to the right. The default is 10.
 - **top** specifies the vertical coordinate that increases to the bottom. The default is 10.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error

occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **display format** specifies the format in which this VI displays the zero-pole-gain equation.

0 **Separated Roots** (default)—Displays the zero-pole-gain equation in the first-order form.

1 **Combined Roots**—Displays the zero-pole-gain equation in the second-order form.

- **filter out** returns the **filter in** unchanged.
- **zero-pole-gain equation** returns the zero-pole-gain equation of **filter in** in a picture indicator.
- **draw area size** returns the size of the area in the picture indicator this VI uses to draw the zero-pole-gain equation.
 - **width** returns the width of the draw area.
 - **height** returns the height of the draw area.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front

panel indicator and select **Explain Error** from the shortcut menu for more information about the error.

- **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
- **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

Refer to the Filter Rendering VI in the labview\examples\Digital Filter Design\Getting Started\Analyze Filters directory for an example of using the DFD Render Zero-Pole-Gain Equation VI.

DFD Save to Text File VI

Owning Palette: <u>Utilities VIs</u>

Installed With: Digital Filter Design Toolkit

Saves a filter to a text file in XML format.

Use the DFD Load from Text File VI to load the filter from the text file.

Note This VI uses periods for decimal points in the coefficients values.

Details Examples

M



Place on the block diagram Find on the Functions palette

filter in specifies the input filter.

File path in specifies the path to the file in which you want to save the filter. The default file extension is .xml. You can use any other file extension for the file. If **file path in** is empty (default) or <Not A Path>, this VI displays a dialog box from which you can select a file. This VI returns an error if you specify a file path that does not exist or if you click the **Cancel** button on the dialog box. Use the DFD Load from Text File VI to load the filter from the text file.

- **comments** specifies the comments that you want to add to the text file.
- error in describes error conditions that occur before this VI or function runs. The default is no error. If an error occurred before this VI or function runs, the VI or function passes the error in value to error out. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in error out. Use the <u>Simple Error Handler</u> or <u>General Error</u> <u>Handler</u> VIs to display the description of the error code. Use error in and error out to check errors and to specify execution order by wiring error out from one node to error in of the next node.

status is TRUE (X) if an error occurred before this VI or

function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
- **source** specifies the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning. The default is an empty string.
- **filter out** returns the **filter in** unchanged.
- **file path out** returns the path to the file to which this VI saved the filter.
- error out contains error information. If error in indicates that an error occurred before this VI or function ran, error out contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the error out front panel indicator and select Explain Error from the shortcut menu for more information about the error.
 - **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
 - **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero <u>error code</u>. If **status** is FALSE, **code** is 0 or a warning code.
 - **source** describes the origin of the error or warning and is, in most cases, the name of the VI or function that produced the error or warning.

DFD Save to Text File Details

This VI saves a filter as an .xml file by default. The following table lists the tags that the .xml file might contain.

Tag Name	Description
DFDTData	Contains the entire information about a filter.
CustomerComments	Contains the information you specified in the comments input of this VI.
Order	Contains the filter order.
Structure	Contains the filter structure.
Coefficients	Contains the floating-point filter coefficients. If the filter is a fixed-point filter, this tag contains the coefficients of the reference floating-point filter. Depending on the type of filter you specified in the filter in input, this tag can contain the following sub-tags:
	 FIRCoefficients—Contains the coefficients of the filter if the filter has a finite impulse response (FIR) structure. ForwardCoefficients—Contains the forward coefficients of the filter if the filter has an infinite impulse response (IIR) Direct Form structure.
	 ReverseCoefficients—Contains the reverse coefficients of the filter if the filter has an IIR Direct Form structure.
	 NumberOfSections—Contains the number of sections in the filter if the filter has an <u>IIR Cascaded Second-Order</u> <u>Sections Form structure</u>.
	• SecondOrderSectionsCoefficients— Contains the second-order section representation of the filter if the filter has an IIR Cascaded Second-Order Sections Form structure. This tag contains an <i>L</i> - by-6 matrix, where <i>L</i> is the number of

	 rows of the matrix. Each row of the matrix contains the coefficients of one filter section in the form [b0 b1 b2 1 a1 a2]. ReflectionCoefficients—Contains the reflection coefficients of the filter if the filter has a lattice allpass, lattice auto-regressive (AR), lattice moving average (MA), or lattice ARMA structure. LadderCoefficients—Contains the ladder coefficients of the filter if the filter has a lattice ARMA structure. Gain—Contains the filter gain that corresponds to the filter coefficients.
ZeroPoleGain	 Contains the floating-point zeroes, poles, and filter gain. If the filter is a fixed-point filter, this tag contains the zeroes, poles, and filter gain of the reference floating-point filter. This tag contains the following sub-tags: Zeroes—Contains the zeroes of the filter. Poles—Contains the poles of the filter. Gain—Contains the filter gain that corresponds to the zeroes and poles.
FXPInformation	 Contains the entire information about the fixed-point filter. If the filter is a floating-point filter, this XML file does not contain this tag. If the filter is a fixed-point filter, this tag can contain the following sub-tags: FXPCoefficients—Contains the fixed-point filter coefficients. This tag contains the same sub-tags as the Coefficients tag. IntegerCoefficients—Contains the integer filter coefficients. This tag contains the same sub-tags as the Coefficients tag. FXPCoefficients—Contains the integer filter coefficients. This tag contains the same sub-tags as the Coefficients tag. FXPZeroPoleGain—Contains the fixed-

point zeroes, poles, and filter gain. This tag contains the same sub-tags as the **ZeroPoleGain** tag.

- Quantizers—Contains information about the filter coefficients quantizer and filter gain quantizer. Each quantizer corresponds to a sub-tag, except the NumberOfSections sub-tag, in the Coefficients tag. Each quantizer contains the following sub-tags:
 - **WordLength**—Contains the word length of the quantizer.
 - IntegerWordLength—Contains the integer word length of the quantizer.
 - OverflowMode—Contains the overflow mode setting of the quantizer.
 - RoundingMode—Contains the rounding mode setting of the quantizer.
 - Signed—Indicates the quantizer uses a signed fixed-point format. This tag is always empty.
 - **Unsigned**—Indicates the quantizer uses an unsigned fixed-point format. This tag is always empty.

Note If the gain processing occurs on a host machine, or if the gain processing occurs on a target but the filter gain is a power-of-2 value, the filter gain quantizer contains the following message only: No integer gain is processed on the target.

N

Refer to the following VIs for examples of using the DFD Save to Text File VI:

• Export FIR Coef to Xilinx COE File VI: labview\examples\Digital Filter Design\Fixed-Point Filters\Single-Rate

Open example Browse related examples

• Export SOS Coef to MathScript VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

Open example Browse related examples

• Export SOS Zero-Pole-Gain to MathScript VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

■ Open example ■ Browse related examples

• Save and Load Filter to and from Text File VI: labview\examples\Digital Filter Design\Getting Started\Apply Filters

Error Codes (Digital Filter Design Toolkit)

The <u>Digital Filter Design</u> VIs can return the following error codes. Refer to the <u>KnowledgeBase</u> for more information about correcting errors in LabVIEW.

Code	Description
-24116	The band edge frequencies must correspond to the frequency points.
-24115	The length of the input string must be compatible with the length of frequency points.
-24114	The length of the input array must be compatible with the length of frequency points.
-24113	The frequency points must be in ascending order.
-24112	The filter must have a nonnegative zero-phase response.
-24111	The input filter is not a valid fixed-point moving average (MA) filter. Refer to the LabVIEW Help for more information about fixed-point MA filters.
-24110	The input filter must be a fixed-point filter with the IIR Cascaded Second-Order Sections Form structure.
-24109	The code generation was cancelled.
-24108	To generate C code from the filter, you must specify whether to process the gain on the target or the host.
-24107	The input signal must be in the range specified in the input word length value. Refer to the LabVIEW Help for more information about the valid range for the input signal.
-24106	The bandwidth must be greater than 0 and less than f0/2 for a Type I comb filter design. The bandwidth must be greater than 0 and less than f0 for a Type II comb filter design.
-24105	The bandwidth must be greater than 0 and less than the value of fn/N, where fn is the Nyquist frequency and N is the value you specify for the # notches/peaks input.
-24104	The Q factor must be greater than the value of f0/fn, where f0 is the center frequency of the notch or peak and fn is the Nyquist frequency.
1	

-24103	To design a filter by using the Q factor, you must specify the center frequency value f0 in the range (0, fn), where fn is the Nyquist frequency. If you want to design a filter with the center frequency at DC, choose the By Bandwidth instance.
-24102	The bandwidth must be greater than 0 and less than the Nyquist frequency.
-24101	The center frequency f0 must be greater than or equal to 0 and less than the Nyquist frequency.
-24100	This function is not applicable to cascaded integrator comb (CIC) filters.
-24099	The filter must be a linear phase finite impulse response (FIR) filter.
-24098	This function failed to calculate the spectral factor of the filter. Ensure that the filter is a linear phase filter with a nonnegative zero-phase response.
-24097	If you want to generate code from the resulting filter, the integer word length that you specify for the multiplicand quantizer must be greater than or equal to the integer word length of the sum quantizer.
-24096	If you want to generate code from the resulting filter, the integer word length that you specify for the multiplicand quantizer must be greater than or equal to the integer word length of the delay quantizer.
-24095	The number of channels must be greater than 0.
-24094	You must specify a valid value for the word length. Refer to the LabVIEW Help for more information about the valid range of word lengths.
-24093	If you want to process the filter gain on a fixed-point target, you must specify the word length of the filter gain in the range [1, 32].
-24092	You must specify the word length of at least one set of coefficients in the range [1, 32].
-24091	The input coefficients are invalid for the cascaded structure you specified.
-24088	The input multirate filter is not a valid cascaded integrator comb (CIC) filter.

-24087	The file path you specified is invalid. Specify a valid file path.
-24086	A memory overflow occurred on the field-programmable gate array (FPGA) target.
-24085	Fixed-point multirate finite impulse response (FIR) code generation supports factors only in the range [1, 255]. The taps per phase must be in the range [1, 32767].
-24084	Fixed-point cascaded integrator comb (CIC) filter code generation supports factors only in the range [1, 65535].
-24083	The code generation supports only one- to 255-channel processing.
-24082	Fixed-point cascaded integrator comb (CIC) filter code generation supports only one-channel processing.
-24081	A project with the same name already exists in memory. Close the existing project or specify another name for the new project.
-24080	The passband edge frequency must be greater than zero and less than the Nyquist frequency. Refer to the LabVIEW Help for more information about specifying a valid value for the passband edge frequency.
-24079	You must specify a valid filtering mode. Refer to the LabVIEW Help for more information about specifying the filtering mode.
-24078	The array of the input filter cannot be empty.
-24077	The array size for interstage word length values must equal # stages - 1, where # stages is the number of stages of the multistage multirate filter.
-24076	The DFD Plot NStage MRate Freq Response VI does not support multistage multirate filters that contain rational resampling filters.
-24075	The array size for the coefficient word length values must be the same as that of the input multirate filter.
-24074	The stopband edge frequency must be greater than the passband edge frequency.
-24073	To generate LabVIEW FPGA code, you must install the LabVIEW FPGA Module and NI-RIO driver software with R Series support. To execute the FPGA code, you also need an

	FPGA target on which to run the code.
-24071	The fixed-point cascaded integrator comb (CIC) filter does not support the Highpass mode. You must set the differential delay and # stages inputs to 1 if you want to design a fixed- point CIC filter that works as a moving-average (MA) filter.
-24070	The input range must be greater than 0.
-24069	The output phase of a decimation filter must be in the range [0, factor).
-24068	The fixed-point multirate filter object is invalid.
-24067	You must specify the input word length value in the range [1, 32].
-24066	The filtering mode of the multirate filter does not match the instance you chose in the polymorphic VI. Specify the appropriate instance of the polymorphic VI.
-24065	The pair of rational factors cannot be equal.
-24063	The value of fs/f0 must be an integer for a Type I comb filter design, where fs and f0 are the sampling frequency and central frequency, respectively. The value of fs/(f0*2) must be an integer for a Type II comb filter design.
-24062	You must specify a delay value that is greater than or equal to zero.
-24061	You must specify a roll off value in the range [0, 1].
-24060	You must specify a gain value that is not equal to zero.
-24059	The zeroes that you specified cannot contain NaN or zero values. The poles you specified cannot contain NaN or Inf values.
-24058	The stopband edge frequency value is too high and might introduce aliasing distortion in the passband.
-24057	The product of the factors in the manual factorization input must equal the factor input value.
-24056	You must specify a factor value that is greater than 4 and divisible by 4 if you use the cascaded integrator comb (CIC) filter in multistage multirate filters.
-24055	The overflow mode input is set to Saturation for a sum

	quantizer. Set the overflow mode input to Wrap.
-24053	The shift number in the states is invalid.
-24052	The diagonal size must match the matrix size.
-24051	LabVIEW failed to allocate space for data.
-24050	This VI failed to load the filter from the file. Specify a valid file path.
-24049	The multirate filter object is invalid.
-24048	You must specify a valid value for the differential delay. Valid values are 1 and 2.
-24047	The filter design failed with the specified numerator and denominator order values. Use smaller order values.
-24046	You must specify the frequencies of the exact gain within the frequency ranges of the band specs input.
-24045	The number of stages must be in the range [1, 8].
-24044	You must specify numerator order and denominator order values of less than 35.
-24043	You must specify a multirate factor that is greater than zero.
-24042	Bandpass and bandstop filters require an order value that is an even number.
-24041	The structure of the filter is invalid. Refer to the LabVIEW Help for more information about selecting a filter structure.
-24040	You must set the quantizer source input to Coefficients a/k or Coefficients b/v.
-24039	The input coefficients are invalid for the specified lattice filter structure.
-24038	The structure of the filter you specified is not a lattice filter structure.
-24037	The input filter is not an finite impulse response (FIR) filter.
-24036	The input filter is not a quantized fixed-point filter.
-24035	The filter design failed with the specifications you entered.
-24034	The value of the states in input must match the filter order.
-24032	You must specify a filter order value that is an even number.
-24031	You must specify a denominator order value that is greater than or equal to zero.
--------	--
-24030	You must specify a numerator order value that is greater than zero.
-24029	You must specify a multirate factor that is greater than one.
-24028	You must specify a roll off value in the range (0, 1).
-24027	Ripples in a linear scale must be in the range (0, 1).
-24026	You must specify a valid frequency sequence in the freq specs input.
-24025	The fixed-point filter model you specified is incompatible with the constraints of code generation. Refer to the LabVIEW Help for more information about generating code with the Digital Filter Design VIs.
-24024	The LabVIEW Digital Filter Design Toolkit does not support LabVIEW FPGA code generation for this filter structure.
-24023	You must specify a Q factor or Df value that is greater than zero.
-24022	You must specify an order value that is greater than zero.
-24021	You must specify frequency values that are greater than zero and less than the Nyquist frequency.
-24020	Each band must contain at least one point.
-24019	You must specify high frequency values that are greater than low frequency values.
-24018	You must specify a sampling frequency value that is greater than zero.
-24017	You must specify a stopband attenuation or ripple value that is greater than 0.
-24016	You must specify a nonnegative value or values for the magnitude input.
-24015	The filter object is invalid.
-24014	The transfer function is invalid.
-24013	You must specify a ripple ratio that is greater than zero for Dolph-Chebyshev windows.

-24012	The filter cannot be represented by the structure you specified.
-24011	The required order of the filter specifications is too large to work with the minimum order search option. Use the user defined option.
-24010	You must constrain all bands for minimum order designs.
-24009	You must specify a nonnegative frequency response for minimum or maximum phase designs.
-24008	The amplitude value at DC must be zero for odd-order, antisymmetric, finite impulse response (FIR) filter designs.
-24007	The amplitude values at the DC and Nyquist frequencies both must be zero for even-order, antisymmetric, finite impulse response (FIR) filter designs.
-24006	The amplitude value at the Nyquist frequency must be zero for odd-order, symmetric, finite impulse response (FIR) filter designs.
-24005	You must define at least one valid band.
-24004	All weighting values must be positive.
-24003	The band frequency must be in ascending order. Bands cannot overlap and must occur between the DC and Nyquist frequencies.
-24002	This VI failed to design a filter that meets all constraints. Specify a larger ripple constraint or order value.
-24001	The number of iterations has reached its maximum value, or a numeric error in the Remez design has occurred.
24001	The actual sampling frequency differs from the one used to create the filter.
24002	The timestamps of the two sequential data blocks are inconsistent.
24003	The input sequence is empty.
24004	The state in input is empty.
24005	The value of p must be in the range [2, 128].
24006	The pole radius must be in the range (0, 1].
0.4007	Zaraga at two ands of the transfer function have been removed

	The designed order is different from the specified order.
24009	Either the maximum iteration has been reached, or a numerical error has occurred during the least pth norm design.
24010	This VI will design a filter with a lower order than the order you specified.
24011	The pair of rational factors are not coprime.
24012	The filtering process will initialize because the sampling frequency of the signal to process has changed from that of the preceding block of signal data.
24013	The filtering process will initialize because the number of signal channels to process has changed.
24014	The filtering process will initialize because the delay value has changed.
24015	The filtering process will initialize because the number of multirate filters has changed.
24016	The input word length value of a stage must equal the output word length value of the preceding stage.
24017	The stopband edge frequency must be less than or equal to the Nyquist frequency if you do not allow aliasing.
24018	The design process might take a long time because the estimated filter order is greater than 1000.
24019	You first must click the Update Design button to design a floating-point filter before you can quantize the filter. You cannot quantize the filter if the filter is a multistage no-rate-change filter.