Filter Design Toolbox Release Notes

The "Filter Design Toolbox 3.2 Release Notes" on page 1-1 describe the changes introduced in the Filter Design Toolbox 3.2. The following topics are discussed in these release notes:

The Filter Design Toolbox Release Notes also provide information about Version 3.1, in case you are upgrading from a version that was released prior to Release 14SP2. Refer to the "Filter Design Toolbox 3.1 Release Notes" on page 2-1.

Printing the Release Notes

If you would like to print the release notes, you can link to a PDF version.

Contents

1

Filter Design Toolbox 3.2 Release Notes

New Features

This section introduces the new features and enhancements added to the Filter Design Toolbox 3.2 since Version 3.1.

If you are upgrading from a release earlier than Release 14SP1, refer to "New Features" on page 2-2 in the Filter Design Toolbox 3.1 Release Notes.

Improved Fixed-Point Support for FIR Filters

Four FIR filters now support fixed-point processing using the same properties or attributes and methods (mostly) that the fixed-point multirate filters use.

- `dfilt.dfasymfir`
- `dfilt.dffir`
- `dfilt.dffirt`
- `dfilt.dfsymfir`

With the improved filter objects, the properties for your discrete-time filters now look the same as your multirate filters. Unifying the look and feel makes working with the full range of filters in the toolbox easier and more clear.

Additionally, making the switch from floating-point to fixed-point by setting `Arithmetic` to `fixed` creates a fixed-point version of your floating-point filter that uses full precision arithmetic internally. The result—a fixed-point filter that most closesly matches to your floating-point prototype. If your design cannot support the resources for this fixed-point implementation, you can start to adjust the fixed-point properties as you need.

To continue to use your existing fixed-point FIR filters, you have to upgrade them to the new format. The toolbox includes a new utility for making the transition—`legacyfixptfir`. Note that this utility is not covered in the Filter Design Toolbox documentation. You can get help by entering

```
help legacyfixptfir
```

at the MATLAB prompt.

For information about converting your existing fixed-point FIR filters to the new objects, refer to "Upgrading Your Existing Fixed-Point FIR Filters to the New Properties" on page 1-7.

Fixed-Point Linear and Hold Interpolators

Both `mfilt.holdinterp` and `mfilt.linearinterp` now let you use fixed-point arithmetic. After you create the interpolator object, you can switch the setting for the `Arithmetic` property to `fixed` to use fixed-point interpolation.

Both also support single-precision floating-point arithmetic.

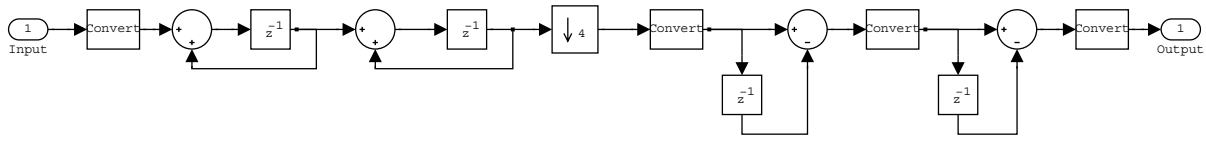
realizemdl Creates CIC Filters

You can use `realizemdl` to construct CIC filters from basic blocks for processing signals. If you construct a CIC decimator filter, as shown here, `realizemdl` can make an atomic subsystem CIC filter block in Simulink for you.

```
hm=mfilt.cicdecim(4);

realizemdl(hm)
```

A new Simulink model window opens and you see a filter block. Double-clicking on the new block shows you the CIC filter subsystem.



Note Note—you must have the Signal Processing Blockset to use realize model to implement CIC filters.

Context-Sensitive Help for FDATool Returns

FDATool now provides help for options on the quantization, multirate filter design, and frequency transformation panels. Access the new help feature either by right-clicking on an option and selecting **What's This** from the context menu, or clicking the **What's This** help icon on the tool bar.

Second-Order Section Filter View Options Available from the Command Line

In Filter Visualization Tool (FVTool), you can view second-order section filters as “individual sections,” “cumulative sections,” or as sections that you define. Now this functionality is available from the MATLAB command line, by using the `sosViewSettings` property of the FVTool object. In previous releases these view options were available only as options in the SOS View Settings dialog in FVTool.

Access the FVTool object properties by launching FVTool with a filter object and including a left-hand side output argument:

```
handle = fvtool(hd)
```

`handle` now contains the FVTool properties, similar to the following listing—you use `set` and `get` to manipulate the property values.

```
handle=fvtool(hd)

handle =

     1

set(handle.sosviewsettings,'view')

ans =

    'Complete'
    'Individual'
    'Cumulative'
    'UserDefined'

set(handle.sosviewsettings,'view','individual')
```

In `SOSViewSettings`, the options are the same, with the same meaning, that you find in **View->SOS View Settings** in FDATool.

For more information about the `fvtool` properties, refer to `fvtool` in the Signal Processing Toolbox documentation or in the online Help system.

Function fdesign Designs Filters with Specified Structure

You can use `fdesign.type` to design a filter and specify the filter structure to use during construction.

Major Bug Fixes

The Filter Design Toolbox 3.2 includes important bug fixes made since Version 3.1. You can see a list of major Version 3.2 bug fixes on the MathWorks Web site.

If you are viewing these release notes in PDF form on the MathWorks Web site, please refer to the HTML form of the release notes on the MathWorks Web site and use the link provided.

If you are upgrading from a version earlier than Version 3.2, you should also see Version 3.1 major bug fixes in the Filter Design Toolbox 3.1 Release Notes.

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving to the Filter Design Toolbox 3.2 from Version 3.1.

If you are upgrading from a version earlier than Version 3.1 you should also see the Version 3.1 upgrade issues described in the Filter Design Toolbox 3.1 Release Notes.

Upgrading Your Existing Fixed-Point FIR Filters to the New Properties

There is a utility named `legacyfixptfir` to ensure backward compatibility of your existing scripts and a function `update` to help you migrate to the new FIR filters. `legacyfixptfir` switches the preferences for your session between pre- and post-Filter Design Toolbox 3.2 FIR filters.

Here is an example of the process of converting your old FIR filters to the new form in this version of the toolbox.

Begin with an existing direct-form FIR filter `h` that you constructed with

```
h = dfilt.dffir
```

in an earlier version of the toolbox. First, use `legacyfixptfir` to retrieve `h` in the old format. Then convert `h` to the new form.

```
legacyfixptfir(true) % To get the old form of h.
h.Arithmetic='fixed'

h =

     FilterStructure: 'Direct-Form FIR'
          Arithmetic: 'fixed'
           Numerator: 1
    PersistentMemory: false

     CoeffWordLength: 16
      CoeffAutoScale: true
              Signed: true

     InputWordLength: 16
```

```
      InputFracLength: 15

     OutputWordLength: 16
           OutputMode: 'AvoidOverflow'

          ProductMode: 'FullPrecision'

            AccumMode: 'KeepMSB'
      AccumWordLength: 40
        CastBeforeSum: true

            RoundMode: 'convergent'
         OverflowMode: 'wrap'

update(h) % Convert h to the new properties.
h

h =

      FilterStructure: 'Direct-Form FIR'
          Arithmetic: 'fixed'
            Numerator: 1
     PersistentMemory: false

      CoeffWordLength: 16
       CoeffAutoScale: true
               Signed: true

      InputWordLength: 16
      InputFracLength: 15

      FilterInternals: 'SpecifyPrecision'

     OutputWordLength: 16
     OutputFracLength: 13

    ProductWordLength: 32
    ProductFracLength: 29
```

```
AccumWordLength: 40
AccumFracLength: 29

      RoundMode: 'convergent'
   OverflowMode: 'wrap'
```

Note the changes in properties. The filter performs the same way but the attributes are now updated to the newest form.

Filter Weights Have Been Removed from the Specifications in fdesign

The weights applied to the filter magnitude response are now design options. They are no longer properties of the `fdesign.typeobject`. To set them, pass them as property name/property value (PV) pairs to the appropriate filter design method, as shown in this example.

```
h = fdesign.lowpass('N,Fp,Fst',30) % Was 'N,Fp,Fst,Wp,Wst'.
                                    % Removed Wp and Wst.
hd = equiripple(h, 'Wpass', 3, 'Wstop', 25); % Specify the
                                              % weights here.
hd(2) = equiripple(h, 'Wpass', 3, 'Wstop', 1);
fvtool(hd)
```

Known Software and Documentation Problems

The Filter Design Toolbox 3.2 includes several bugs identified since Version 3.1. You can see a list of important Version 3.2 bugs on the MathWorks Web site.

If you are viewing these Release Notes in PDF form, please refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the link provided.

2

Filter Design Toolbox 3.1 Release Notes

New Features

This section introduces the new features and enhancements added to the Filter Design Toolbox 3.1 since Version 3.0.

CIC Filters Now Support Word Lengths Larger Than 32 Bits

In addition to adding fixed-point capability to CIC filters, you can also design filters that use more than 32 bits for the word length.

CIC Filters Now Use Fixed-Point Filtering

The CIC filters in the toolbox — `mfilt.cicdecim` and `mfilt.cicinterp`—now use fixed-point filtering. Tools in the toolbox let you determine the number of bits required for your CIC filter.

Fixed-Point and Single-Precision Multirate Filters Now Available

Four of the multirate filters in the toolbox now provide single-precision floating-point and fixed-point filtering. They are:

- `mfilt.firdecim` — Constructs a direct-form FIR polyphase decimator that can use fixed-point filtering
- `mfilt.firinterp` — Constructs a direct-form FIR polyphase interpolator that can use fixed-point filtering
- `mfilt.firsrc` — Constructs a direct-form FIR polyphase sample rate changer that can use fixed-point filtering
- `mfilt.firtdecim` — Constructs a direct-form transposed FIR polyphase decimator that can use fixed-point filtering

Each now has the `Arithmetic` property, similar to discrete-time filter objects, that lets you select fixed-point arithmetic after you construct your multirate filter. FDATool supports the new single-precision and fixed-point arithmetic options as well.

Multirate Filters Support Processing by Samples

It is now possible to filter a signal one data sample at a time with any multirate filter. Previously, you could only filter vectors of data with minimum length equal to the decimation factor.

This change implies that the length of the output signal may change in some cases. In particular, for `mfilt.firdecim`, `mfilt.firtdecim`, `mfilt.cicdecim`, `mfilt.firsrc`, `mfilt.firfracinterp`, and `mfilt.firfracdecim`, the length of the output signal will be different when the length of the input signal is not a multiple of the decimation factor. Care must be taken if you have existing code that depends on the length of the output signal for these cases.

New Introduction to Multirate Filters in the User's Guide

For this release we added a chapter to the Filter Design Toolbox User's Guide to introduce you to developing and analyzing multirate filters in the toolbox. The new material covers constructing multirate filters, both CIC and FIR based, and analyzing the results of your filter design.

New Demos for Multirate Filters

We added some new demos in the Filter Design Toolbox that show you how to work with multirate filters. They demonstrate general information about multirate filters, as well as providing a few details about CIC filters, FIR multirate filters, and analyzing multirate filters.

Major Bug Fixes

The Filter Design Toolbox 3.1 includes several bug fixes made since Version 3.0. This section describes the particularly important Version 3.1 bug fixes on the MathWorks Web site.

If you are viewing these Release Notes in PDF form, please refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the link provided.

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving to the Filter Design Toolbox 3.1 from Version 3.0.

mfilt.cicdecimzerolat and mfilt.cicinterpzerolat Removed from the Toolbox

With this release, the zero-latency versions for CIC decimators and interpolators—`mfilt.cicdecimzerolat` and `mfilt.cicinterpzerolat`—are no longer provided by the toolbox. If you use these filters, they no longer work. Use the new `mfilt.cicdecim` and `mfilt.cicinterp` to replace the older zero-latency versions.

Multirate Filters Support Processing by Samples

It is now possible to filter a signal one data sample at a time with any multirate filter. Previously, you could only filter vectors of data with minimum length equal to the decimation factor. This change implies that there no longer are nonprocessed samples. Therefore, the `NonProcessedSamples` property has been removed.

This change implies that the length of the output signal may change in some cases. In particular, for `mfilt.firdecim`, `mfilt.firtdecim`, `mfilt.cicdecim`, `mfilt.firsrc`, `mfilt.firfracinterp`, and `mfilt.firfracdecim`, the length of the output signal will be different when the length of the input signal is not a multiple of the decimation factor. Care must be taken if you have existing code that depends on the length of the output signal for these cases.

Property NonProcessedSamples Has Been Removed

It is now possible to filter a single sample at a time with any multirate filter. Previously, you could only filter vectors of data with minimum length equal to the decimation factor. This change implies that there no longer are any samples that are not processed in filtering. Therefore, the `NonProcessedSamples` property has been removed.

In Discrete-Time, Multirate, and Adaptive Filters, Property PersistentMemory Replaces ResetBeforeFiltering

Although using `ResetBeforeFiltering` still works, the property is no longer displayed. Moreover, the property will be removed in the future and therefore should no longer be used. `PersistentMemory` should be used instead. The mapping from one property to the other is as follows:

`ResetBeforeFiltering`: `on`—> `PersistentMemory`: `false`
`ResetBeforeFiltering`: `off`—> `PersistentMemory`: `true`

Loading Multirate Filters Saved Prior to Version 3.0 of the Filter Design Toolbox

If you have multirate filters that you created and stored in toolbox versions prior to 3.0, we recommend that you load these filters in Version 3.1 and use the settings to create new filters. Then save the newly reconstructed filters in place of your previously saved ones. This is necessary because, depending on your multirate filter settings, loading your multirate filters saved in earlier versions of the toolbox may not result in working filters.