Signal Processing Blockset Release Notes

The Signal Processing Blockset 6.2 Release Notes describe the changes introduced in the latest version of the Signal Processing Blockset. The following topics are discussed in these Release Notes.

The Signal Processing Blockset Release Notes also provide information about recent versions of the product, in case you are upgrading from a version that was released prior to Release 14 with Service Pack 2.

Printing the Release Notes. If you would like to print the Release Notes, you can link to a PDF version.

Contents

1 Signal Processing Blockset 6.2 Release Notes

2 Signal Processing Blockset 6.1 Release Notes

Signal Processing Blockset 6.0.1 Release Notes

3

Signal Processing Blockset 6.0 Release Notes

4

5 DSP Blockset 5.1 Release Notes

6 DSP Blockset 5.0 Release Notes

1

Signal Processing Blockset 6.2 Release Notes

New Features

This section summarizes the new features and enhancements introduced in the Signal Processing Blockset 6.2:

If you are upgrading from a release earlier than Release 14SP2, then you should also see New Features in the Signal Processing Blockset 6.1 Release Notes.

New Numerically Controlled Oscillator (NCO) Block

The NCO block in the Signal Operations library is new for this release.

Digital Filter Block Enhancements

Significant enhancements were made to the Digital Filter block for this release:

- Digital Filter can now operate in two different modes, which you select in the **Filter source** group box. If you select **Specify filter characteristics in dialog**, you enter information about the filter in the block mask as in previous releases. If you select **Specify discrete-time filter object (DFILT)**, you can now specify the filter using a `dfilt` object from the Signal Processing Toolbox.
- You can now open the Signal Processing Toolbox `FVTool` from the Digital Filter block mask to view the filter response.

Fixed-Point Support Added to the Matrix Multiply Block

The Matrix Multiply block now has functionality identical to the Simulink Product block. The block now supports Boolean, integer, and fixed-point data types.

Simulink Virtual Bus Support Added to Key Blocks

Simulink virtual bus support has been added to the following blocks:

- DCT
- Delay
- Flip
- Overwrite Values
- Submatrix
- Transpose

For more information on virtual buses, refer to “Signal Buses” in the Using Simulink documentation.

New Audio Sample Rate Conversion Demo

The new Audio Sample Rate Conversion demo illustrates audio sample rate conversion of a 48 kHz (DAT sampling rate) input audio signal to a 44.1 kHz (CD sampling rate) output audio signal using a multistage multirate FIR rate conversion approach. You can access this demo from the **Demos** pane of the Help browser under **Blocksets** > **Signal Processing** > **Audio Processing**.

Major Bug Fixes

To view major bug fixes made in R14SP3 for the Signal Processing Blockset, use the Bug Reports interface on the MathWorks Web site.

Note that if you are not already logged in to Access Login, when you link to the Bug Reports interface (see below), you will be prompted to log in or create an Access Login account.

After you are logged in, use this Bug Fixes link. You will see the bug report for the Signal Processing Blockset. The report is sorted with fixed bugs listed first, and then open bugs.

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

For bug fixes added prior to R14SP2, see “Major Bug Fixes” on page 3-5 in the Signal Processing Blockset 6.0.1 release notes.

Known Software Problems

To view important open bugs in R14SP3 for the Signal Processing Blockset, use the Bug Reports interface on the MathWorks Web site.

Note that if you are not already logged in to Access Login, when you link to the Bug Reports interface (see below), you will be prompted to log in or create an Access Login account.

After you are logged in, use this Open Bugs link. You will see the bug report for the Signal Processing Blockset. The report is sorted with fixed bugs listed first, and then open bugs. You can select the Status column to list the open bugs first.

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

2

Signal Processing Blockset 6.1 Release Notes

New Features

This section introduces the new features and enhancements added to the Signal Processing Blockset 6.1 (Release 14 with Service Pack 2) since Signal Processing Blockset 6.0.1 (Release 14 with Service Pack 1):

Broader Support for the Logging of Simulation Minimums and Maximums and Fixed-Point Autoscaling

An increased number of fixed-point capable blocks in the Signal Processing Blockset now support the logging of simulation minimums and maximums and autoscaling via the Fixed-Point Settings interface. Refer to Fixed-Point Settings Interface in the Signal Processing Blockset User's Guide for more information.

Fixed-Point Support for the DCT and IDCT Blocks

The DCT and IDCT blocks now support fixed-point data types.

New Audio File Source and Sink Blocks

The From Multimedia File and To Multimedia File blocks in the Platform Specific I/O > Windows (WIN32) library are new in this release.

Multirate Support for CIC Filter Blocks

The CIC Decimation and CIC Interpolation blocks now support multirate sample-based processing.

Major Bug Fixes

The Signal Processing Blockset Version 6.1 includes important bug fixes made since Version 6.0.1. This section describes the important Version 6.1 bug fixes. You can see a list of "Major Bug Fixes" on page 3-5 in Version 6.0.1 on the MathWorks Web site.

Segmentation Violation with Pad and Zero Pad Blocks Fixed

In the previous release, using the Pad or Zero Pad blocks under certain conditions could cause a segmentation violation. This has been fixed.

Waterfall Scope Crash Problem Fixed

In the previous release, running a model with an open Waterfall Scope could cause MATLAB to crash. This has been fixed.

dsp_links and liblinks Functions Fixed

In previous releases, the `dsp_links` and `liblinks` functions could behave erratically when used with large models with a significant number of nested links. This behavior has been fixed.

Upgrading from an Earlier Release

This section describes issues you might encounter when upgrading to the Signal Processing Blockset 6.1.

Obsolete Blocks

You can run the Signal Processing Blockset function `dsp_links` to see if you are using any obsolete blocks in your models. If your models are using obsolete blocks, we strongly recommend that you exchange them for blocks that are currently supported.

The blocks in the table below are obsolete, although they are currently still shipped with the product, and may be removed in a future version of the Signal Processing Blockset. We recommend that you use the replacement blocks listed in the third column.

To access each replacement block, type the library name listed in the **Replacement Block(s) Library** column at the MATLAB command line.

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
Analog Filter Design	`dspddes2`	Analog Filter Design	`dsparch4`
Analytic Signal	`dspbdsp2`	Analytic Signal	`dspxfrm3`
Autocorrelation	`dspvect2`	Autocorrelation	`dspstat3`
Backward Substitution	`dsplinalg`	Backward Substitution	`dspsolvers`
Biquadratic Filter	`dsparch2`	Digital Filter	`dsparch4`
Buffer	`dspbuff2`	Buffer	`dspbuff3`
Buffered FFT Frame Scope	`dspsnks2`	Spectrum Scope	`dspsnks4`
Burg AR Estimator	`dsppparest2`	Burg AR Estimator	`dsppparest3`
Burg Method	`dspspect2`	Burg Method	`dspspect3`
Chirp	`dspsrcs2`	Chirp	`dspsrcs4`

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
Cholesky Factorization	dsplinalg	Cholesky Factorization	dspfactors
Cholesky Solver	dsplinalg	Cholesky Solver	dspsolvers
Commutator	dspswit2	Reshape > Frame Conversion > Unbuffer	Simulink block, dspsigattribs, dspbuff3
Complex Cepstrum	dspxfrm2	Complex Cepstrum	dspxfrm3
Complex Exponential	dspelem2	Complex Exponential	dspmathops
Constant Diagonal Matrix	dspmtrx2	Constant Diagonal Matrix	dspmtrx3
Contiguous Copy	dspelem2	Contiguous Copy	dspobslib
Convert Complex DSP to Simulink	dspelem2	No Direct Replacement	N/A
Convert Complex Simulink to DSP	dspelem2	No Direct Replacement	N/A
Convolution	dspvect2	Convolution	dspsigops
Correlation	dspvect2	Correlation	dspstat3
Covariance AR Estimator	dsppparest2	Covariance AR Estimator	dapparest3
Covariance Method	dspspect2	Covariance Method	dspspect3
Create Diagonal Matrix	dspmtrx2	Create Diagonal Matrix	dspmtrx3
Cumulative Sum	dspvect2	Cumulative Sum	dspmathops
Counter	dspswit2	Counter	dspswit3
dB	dspelem2	dB Conversion	dspmathops
dB Gain	dspelem2	dB Gain	dspmathops
DCT	dspxfrm2	DCT	dspxfrm3
Detrend	dspbdsp2	Detrend	dspstat3

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
Difference	dspvect2	Difference	dspmathops
Digital FIR Filter Design	dspddes2	Digital Filter Design	dsparch4
Digital FIR Raised Cosine Filter Design	dspddes2	Digital Filter Design	dsparch4
Digital IIR Filter Design	dspddes2	Digital Filter Design	dsparch4
Direct-Form II Transpose Filter	dsparch2	Digital Filter	dsparch4
Discrete Constant	dspsrcs2	DSP Constant	dspsrcs4
Discrete Impulse	dspsrcs2	Discrete Impulse	dspsrcs4
Distributor	dspswit2	Buffer	dspbuff3
Downsample	dspbdsp2	Downsample	dspsigops
Dyadic Analysis Filter Bank	dspmlti2	Dyadic Analysis Filter Bank	dspmlti4
Dyadic Synthesis Filter Bank	dspmlti2	Dyadic Synthesis Filter Bank	dspmlti4
Edge Detector	dspswit2	Edge Detector	dspswit3
Event-Count Comparator	dspswit2	Event-Count Comparator	dspswit3
Extract Diagonal	dspmtrx2	Extract Diagonal	dspmtrx3
Extract Triangular Matrix	dspmtrx2	Extract Triangular Matrix	dspmtrx3
FFT	dspxfrm2	FFT	dspxfrm3
FFT Frame Scope	dspsnks2	Spectrum Scope	dspsnks4
Filter Realization Wizard	dsparch2	Filter Realization Wizard	daparch4
FIR Decimation	dspmlti2	FIR Decimation	dspmlti4

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
FIR Interpolation	dspmlti2	FIR Interpolation	dspmlti4
FIR Rate Conversion	dspmlti2	FIR Rate Conversion	dspmlti4
Flip	dspvect2	Flip	dspindex
Forward Substitution	dsplinalg	Forward Substitution	dspsolvers
Frequency Frame Scope	dspsnks2	Vector Scope	dspsnks4
From Wave Device	dspsrcs2	From Wave Device	dspwin32
From Wave File	dspsrcs2	From Wave File	dspwin32
Histogram	dspstat2	Histogram	dspstat3
IDCT	dspxfrm2	IDCT	dspxfrm3
IFFT	dspxfrm2	IFFT	dspxfrm3
Inherit Complexity	dspelem2	Inherit Complexity	dspsigattribs
Integer Delay	dspbdsp2	Delay	dspsigops
Kalman Adaptive Filter	dspadpt2	Kalman Adaptive Filter	dspadpt3
LDL Factorization	dsplinalg	LDL Factorization	dspfactors
LDL Solver	dsplinalg	LDL Solver	dspsolvers
Least Squares FIR Filter Design	dspddes2	Digital Filter Design	dsparch4
Levinson Solver	dsplinalg	Levinson-Durbin	dspsolvers
LMS Adaptive Filter	dspadpt2	LMS Filter	dspadpt3
LPC	dspbdsp2	Autocorrelation LPC	dsplp
LU Factorization	dsplinalg	LU Factorization	dspfactors
LU Solver	dsplinalg	LU Solver	dspsolvers
Magnitude FFT	dspspect2	Magnitude FFT	dspspect3
Matrix 1-Norm	dspmtrx2	Matrix 1-Norm	dspmtrx3
Matrix Constant	dspmtrx2	Constant	Simulink block

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
Matrix From Workspace	dspmtrx2	Signal From Workspace	dspsrcs4
Matrix Multiplication	dspmtrx2	Matrix Multiply	dspmtrx3
Matrix Product	dspmtrx2	Matrix Product	dspmtrx3
Matrix Scaling	dspmtrx2	Matrix Scaling	dspmtrx3
Matrix Square	dspmtrx2	Matrix Square	dspmtrx3
Matrix Sum	dspmtrx2	Matrix Sum	dspmtrx3
Matrix To Workspace	dspmtrx2	To Workspace	Simulink block
Matrix Viewer	dspsnks2	Matrix Viewer	dspsnks4
Maximum	dspstat2	Maximum	dspstat3
Mean	dspstat2	Mean	dspstat3
Median	dspstat2	Median	dspstat3
Minimum	dspstat2	Minimum	dspstat3
Modified Covariance AR Estimator	dsppparest2	Modified Covariance AR Estimator	dsppparest3
Modified Covariance Method	dspspect2	Modified Covariance Method	dspspect3
Multiphase Clock	dspswit2	Multiphase Clock	dspswit3
Normalization	dspvect2	Normalization	dspmathops
N-Sample Enable	dspswit2	N-Sample Enable	dspswit3
N-Sample Switch	dspswit2	N-Sample Switch	dspswit3
Overlap-Add FFT Filter	dsparch2	Overlap-Add FFT Filter	dsparch4
Overlap-Save FFT Filter	dsparch2	Overlap-Save FFT Filter	dsparch4
Partial Unbuffer	dspbuff2	Submatrix > Unbuffer	dspmtrx3, dspbuff3
Permute Matrix	dspmtrx2	Permute Matrix	dspmtrx3

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
Polynomial Evaluation	dspelem2	Polynomial Evaluation	dsppolyfun
Queue	dspbuff2	Queue	dspbuff3
QR Factorization	dsplinalg	QR Factorization	dspfactors
QR Solver	dsplinalg	QR Solver	dspsolvers
Random Source	dspsrcs2	Random Source	dspsrcs4
Repeat	dspbdsp2	Repeat	dspsigops
Real Cepstrum	dspxfrm2	Real Cepstrum	dspxfrm3
Rebuffer	dspbuff2	Buffer	dspbuff3
Reciprocal Condition	dsplinalg	Reciprocal Condition	dspmtrx3
Remez FIR Filter Design	dspddes2	Digital Filter Design	dsparch4
Reshape	dspmtrx2	Reshape	Simulink block
RLS Adaptive Filter	dspadpt2	RLS Filter	dspadpt3
RMS	dspstat2	RMS	dspstat3
Shift Register	dspbuff2	Delay Line	dspbuff3
Sample and Hold	dspswit2	Sample and Hold	dspsigops
Short-Time FFT	dspspect2	Periodogram	dspspect3
Signal From Workspace	dspsrcs2	Signal From Workspace	dspsrcs4
Signal To Workspace	dspsnks2	Signal To Workspace	dspsnks4
Sine Wave	dspsrcs2	Sine Wave	dspsrcs4
Sort	dspstat2	Sort	dspstat3
Stack	dspbuff2	Stack	dspbuff3
Standard Deviation	dspstat2	Standard Deviation	dspstat3
Submatrix	dspmtrx2	Submatrix	dspmtrx3
Time Frame Scope	dspsnks2	Vector Scope	dspsnks4

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
Time-Varying Direct-Form II Transpose Filter	dsparch2	Digital Filter	dsparch4
Time-Varying Lattice Filter	dsparch2	Digital Filter	dsparch4
Toeplitz	dspmtrx2	Toeplitz	dspmtrx3
To Wave Device	dspsnks2	To Wave Device	dspwin32
To Wave File	dspsnks2	To Wave File	dspwin32
Transpose	dspmtrx2	Transpose	dspmtrx3
Triggered Matrix To Workspace	dspsnks2	Triggered To Workspace	dspsnks4
Triggered Shift Register	dspbuff2	Triggered Delay Line	dspbuff3
Triggered Signal From Workspace	dspbdsp2	Triggered Signal From Workspace	dspsigops
Triggered Signal To Workspace	dspsnks2	Triggered To Workspace	dspsnks4
Unbuffer	dspbuff2	Unbuffer	dspbuff3
Uniform Decoder	dspquant	Uniform Decoder	dspquant2
Uniform Encoder	dspquant	Uniform Encoder	dspquant2
Unwrap	dspvect2	Unwrap	dspsigops
Upsample	dspbdsp2	Upsample	dspsigops
User-defined Frame Scope	dspsnks2	Vector Scope	dspsnks4
Variable Fractional Delay	dspbdsp2	Variable Fractional Delay	dspsigops
Variable Integer Delay	dspbdsp2	Variable Integer Delay	dspsigops
Variable Selector	dspelem2	Variable Selector	dspindex

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
Variance	dspstat2	Variance	dapstat3
Wavelet Analysis	dspmlti2	Wavelet Analysis	dspobslib
Wavelet Synthesis	dspmlti2	Wavelet Synthesis	dspobslib
Window Function	dspbdsp2	Window Function	dspsigops
Yule-Walker AR Estimator	dspparest2	Yule-Walker AR Estimator	dspparest3
Yule-Walker IIR Filter Design	dspddes2	Digital Filter Design	dsparch4
Yule-Walker Method	dspspect2	Yule-Walker Method	dspspect3
Zero Pad	dspbdsp2	Zero Pad	dspsigops

Known Software Problems

This section lists the major known software problems in Version 6.1 of the Signal Processing Blockset.

CIC Filter Blocks Always Use Zero-Valued Initial Conditions

When the Filter Design Toolbox is installed, the CIC Decimation block or the CIC Interpolation block enable pop-up menus that allow you to specify the filter using an `mfilt` object defined in the MATLAB workspace. The `mfilt` object has a `States` property that holds the initial filter states. These initial state values are currently not being used by the blocks. Instead, the blocks always use zero-valued initial conditions.

FFT, IFFT, Magnitude FFT, and Short-Time FFT Blocks Ignore Hardware Implementation Parameters for Fixed-Point Internal Rules

The FFT and IFFT blocks ignore the settings on the **Hardware Implementation** pane of the Configuration Parameters dialog box if you choose `Inherit via internal rule` for the **Product output**, **Accumulator**, and/or **Output** data type mode parameters. Instead, these blocks work as if the **Device type** parameter is set to ASIC/FPGA, that is, as if fixed-point word lengths are unconstrained. The Magnitude FFT and Short-Time FFT blocks, which are implemented using the FFT block, work the same way.

The `Inherit via internal rule` choice affects the internal calculation of fixed-point word lengths for the product output, accumulator, and/or output data types. These blocks will simulate and generate code; however, they may generate extra code to accurately model fixed-point word lengths that are not natively supported by the compiler for the hardware target.

See the Simulink documentation for more information about the **Hardware Implementation** pane of the Configuration Parameters dialog box.

3

Signal Processing Blockset 6.0.1 Release Notes

Changes from the Previous Release

In this release, the following blocks have been affected by changes in the behavior of source block dialog boxes and the Model Explorer. See the Changes from the Previous Release section in the Simulink Release Notes.

- Chirp
- Constant Diagonal Matrix
- DSP Constant
- Multiphase Clock
- N-Sample Enable
- Random Source
- Sine Wave

New Features

This section introduces the new features and enhancements added to the Signal Processing Blockset 6.0.1 (Release 14 with Service Pack 1) since Signal Processing Blockset 6.0 (Release 14):

New Demos

Demo Name	Signal Processing Demo Library Location	Launch Command
Cochlear implant speech processor	Audio Processing	`dspcochlear_all` (Platform independent) `dspcochlear_all_fixpt` (Platform independent, fixed-point version)
Creating sample-based signals	Working with Signals	`dspcreatesbsigs`
Creating frame-based signals	Working with Signals	`dspcreatefbsigs`
Creating multichannel signals	Working with Signals	`dspcreatemltichansigs`
Splitting and reordering multichannel signals	Working with Signals	`dspsplitreordmltichansigs`
Importing signals	Working with Signals	`dspimportsigs`
Exporting signals	Working with Signals	`dspexportsigs`

Enhanced Blocks

The following blocks have been enhanced for Release 14SP1:

- Sample and Hold
- Spectrum Scope

The Sample and Hold block has a new parameter, the **Latch (buffer) input** check box. If you select this check box, the block outputs the value of the input from the previous time step until the next triggering event occurs. This parameter enables this block to be used in a feedback loop.

The Spectrum Scope block has two new parameters, **Window type** and **Window sampling**. Use the **Window type** parameter to specify which window to apply to the input. Use the **Window sampling** parameter to specify whether the window samples are computed in a periodic or a symmetric manner.

Major Bug Fixes

The Signal Processing Blockset 6.0.1 includes several bug fixes made since Version 6.0. This section describes the important Version 6.0.1 bug fixes.

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

There are no issues for upgrading from Signal Processing Blockset 6.0 (Release 14) to Signal Processing Blockset 6.0.1 (Release 14 with Service Pack 1).

For information about upgrading from an earlier version than Signal Processing Blockset 6.0 (Release 14), see “Upgrading from an Earlier Release” on page 4-14

4

Signal Processing Blockset 6.0 Release Notes

New Features

This section introduces the new features and enhancements added to the Signal Processing Blockset 6.0 (Release 14) since DSP Blockset 5.1 (Release 13 SP1):

If you are upgrading from a release earlier than Release 13 SP1, then you should see of the .

Product Name Change

The DSP Blockset has been renamed. The new name is the Signal Processing Blockset.

Additional Fixed-Point Support

For this release, significant support for fixed-point development has been added to the Signal Processing Blockset.

For a complete list of all blocks in the Signal Processing Blockset that support fixed-point data types, refer to Blocks with Fixed-Point Support in the Signal Processing Blockset User's Guide.

New Fixed-Point Blocks

The following new blocks support fixed-point data types:

- CIC Decimation
- CIC Interpolation
- Offset
- Peak Finder
- Scalar Quantizer Decoder
- Scalar Quantizer Encoder
- Vector Quantizer Decoder
- Vector Quantizer Encoder
- Zero Crossing

Blocks with Added Fixed-Point Support

The following blocks now support fixed-point data types:

- Constant Ramp
- Cumulative Product
- Cumulative Sum
- Difference
- Digital Filter — more structures now support fixed-point data types
- FIR Rate Conversion
- Histogram
- Levinson-Durbin
- LMS Filter
- Matrix 1-Norm

- Matrix Scaling
- Mean
- Median
- Normalization
- Short-Time FFT
- Signal From Workspace
- Signal To Workspace
- Sort
- Triggered Signal From Workspace
- Triggered To Workspace
- Toeplitz
- Two-Channel Analysis Subband Filter
- Two-Channel Synthesis Subband Filter

Fixed-Point Blocks with New Complex Support

The following blocks supported real fixed-point data types in the last major release. They now also support complex fixed-point data types:

- Autocorrelation
- Convolution
- Correlation
- FIR Decimation
- FIR Interpolation
- Sort

Fixed-Point Blocks with a New Interface

Many of the Signal Processing Blockset blocks that support fixed-point data types have a new, easier-to-use interface. For more information, see Setting Block Parameters in the Signal Processing Blockset User's Guide.

New Automatic Selection of Fixed-Point Word and Fraction Lengths

Many blocks with fixed-point support in the Signal Processing Blockset allow you to set intermediate data types via block mask parameters. The **Accumulator**, **Product output**, and **Output** parameters on many such blocks have a new `Inherit via internal rule` setting. When you select `Inherit via internal rule`, the accumulator, product output, or block output word and fraction lengths will be automatically calculated for you. In general, all the bits are preserved in the internal block algorithm for quantities using this selection. That is, the accumulator, product output, or block output word and fraction lengths are selected such that

- No overflow occurs.
- No precision loss occurs.

Internal rule equations specific to each block are given in the block reference pages.

New Logging of Simulation Minimums and Maximums and Autoscaling

A number of fixed-point blocks in the Signal Processing Blockset now support the logging of simulation minimums and maximums and autoscaling via the Fixed-Point Settings interface. Refer to Fixed-Point Settings Interface in the Signal Processing Blockset User's Guide for more information.

New Blocks

A summary of the new blocks is presented below:

- CIC Decimation and CIC Interpolation
- G711 Codec
- Inverse Short-Time FFT
- LPC to/from Cepstral Coefficients
- Offset
- Peak Finder

- Scalar Quantizer Decoder
- Scalar Quantizer Encoder
- Short-Time FFT
- Vector Quantizer Decoder
- Vector Quantizer Design
- Vector Quantizer Encoder
- Waterfall
- Zero Crossing

CIC Decimation and CIC Interpolation

The CIC Decimation and CIC Interpolation blocks are in the Filtering/ Multirate Filters library. These blocks decimate or interpolate a signal using a Cascaded Integrator-Comb filter.

G711 Codec

The G711 Codec block is in the Quantizers library. This block encodes a linear, pulse code modulation (PCM) narrowband speech signal using an A-law or mu-law encoder. The block decodes index values into quantized output values using an A-law or mu-law decoder. The block converts between A-law and mu-law index values.

Inverse Short-Time FFT

The Inverse Short-Time FFT block is in the Transforms library. This block recovers the time-domain signal by performing an inverse short-time, fast Fourier transform operation.

LPC to/from Cepstral Coefficients

The LPC to/from Cepstral Coefficients block is in the Linear Prediction library. This block converts linear prediction coefficients (LPCs) to cepstral coefficients (CCs) or cepstral coefficients to linear prediction coefficients.

Offset

The Offset block is in the Signal Operations library. This block truncates vectors by removing or keeping beginning or ending values.

Peak Finder

The Peak Finder block is in the Signal Operations library. This block finds the local maxima and/or minima of an input signal.

Scalar Quantizer Decoder

The Scalar Quantizer Decoder block is in the Quantizers library. This block converts each index value into a quantized output value.

Scalar Quantizer Encoder

The Scalar Quantizer Encoder block is in the Quantizers library. This block encodes each input value by associating it with the index value of a quantization region.

Short-Time FFT

The Short-Time FFT block is in the Transforms library. This block computes a nonparametric estimate of the spectrum using the short-time, fast Fourier transform method. The Short-Time FFT block that was located in the Power Spectrum Estimation library has been renamed the Periodogram block.

Vector Quantizer Decoder

The Vector Quantizer Decoder block is in the Quantizers library. This block finds the vector quantizer codeword that corresponds to a given, zero-based index value.

Vector Quantizer Design

The Vector Quantizer Design block is in the Quantizers library. This block designs a vector quantizer using the Vector Quantizer Design Tool (VQDTool).

Vector Quantizer Encoder

The Vector Quantizer Encoder block is in the Quantizers library. This block finds the index of the nearest codeword based on a Euclidean or weighted Euclidean distance measure.

Waterfall

The Waterfall block is in the DSP Sinks library. This block enables you to view vectors of data over time.

Zero Crossing

The Zero Crossing block is in the Signal Operations library. This block counts the number of times a signal crosses zero.

Enhanced Blocks

For descriptions of the enhanced blocks, see the following sections:

Counter

The **Count data type** parameter of the Counter block now supports signed and unsigned integers.

Digital Filter

The Digital Filter block now supports these additional filter structures:

- FIR
  - Direct form symmetric
  - Direct form antisymmetric

- IIR Biquad (SOS)
  - Direct form I
  - Direct form I transposed
  - Direct form II

Every filter structure now supports fixed-point data types.

Biquad (SOS) filter structures support interstage floating-point and fixed-point scale values.

Matrix Viewer

The Matrix Viewer block parameters dialog box has been upgraded.

Scalar Quantizer Design

You can now use the Scalar Quantizer Design Tool to create Scalar Quantizer Encoder and Scalar Quantizer Decoder blocks inside your models.

Sort

The Sort block now supports an additional sorting algorithm. Now, for the **Sort algorithm** parameter, you can choose either `Quick sort` or `Insertion sort`. Previously, only the quick sort algorithm was supported.

Renamed Blocks

Periodogram

The Short-Time FFT block that was located in the Power Spectrum Estimation library has been renamed the Periodogram block. This block computes a nonparametric estimate of the spectrum. All instances of the old Short-Time FFT block have been replaced by the Periodogram block.

New Demos

Demo Name	Signal Processing Demo Library Location	Launch Command
Adaptive filter convergence	Adaptive Processing	`lmsxyplot`
CELP speech coder	Audio Processing	`dspcelpcoder`
G711 A-law and A-Mu-A conversion	Audio Processing	`dspg711amua`
G711 Mu-law and Mu-A-Mu conversion	Audio Processing	`dspg711muamu`
G711 and PCM encoding	Audio Processing	`dspg711cmp`
Phase vocoder	Audio Processing	`dsppitchtime`
Plucked string	Audio Processing	`dsppluck`
Radar tracking demonstration	Aerospace	`aero_radmod_dsp`
Short-Time Spectral Attenuation	Spectral Analysis	`dspstsa`
Vector quantizer design	Miscellaneous	`dspvqtwodim`

The Short-Time FFT demo in Spectral Analysis demo library is now the Periodgram demo.

The Acoustic Noise Canceler demo (`dspanc`) is now supported on all platforms. It also has a fixed-point version (`dspanc_fixpt`).

The Signal Processing Blockset has a new demo library called Fixed-Point. This library contains demo models that support fixed-point data types.

Triggered Subsystem Support

Signal Processing Blockset blocks now support triggered subsystems. The exceptions are

- Chirp block
- Multiphase Clock
- Sine Wave block

- Blocks with multiple sample times

Constant Sample Time Support

The Signal Processing Blockset has extended support of constant sample times to its blocks. The output of blocks with constant sample times does not change during the simulation. You can remove all blocks having constant sample times from the simulation "loop" by setting the **Inline parameters** option. If you select the **Inline parameters** check box on the **Optimization** pane of the Configuration Parameters dialog box, the parameters of these blocks cannot be changed during a simulation, and simulation speed is improved.

Source Blocks Obey New Simulink Inherited Sample Time Parameter

Signal Processing Blockset source blocks capable of inheriting their sample time obey a new Simulink inherited sample time parameter. To view this parameter, open the Configuration Parameters dialog box. In the **Select** pane, expand **Diagnostics** and click **Sample Time**. The new parameter, **Source block specifies -1 sample time** appears in the left pane. This parameter can be set to `none`, `warning` (default), or `error`.

The Random Source block is the only block that does not obey this parameter. If its **Sample time** parameter is set to `-1`, the Random Source block inherits its sample time from its output port and never produces warnings or errors.

Signal & Scope Manager Support

You can use the Signal & Scope Manager to create and view signals without using blocks. The Signal Processing Blockset provides signal generators and viewers that you can associate with your model using the Signal & Scope Manager. To view these generators and viewers, right-click in your model, and select **Signal & Scope Manager**. From the **Generators** and **Viewers** lists, expand **Signal Processing**.

For information on how to use the Signal & Scope Manager, see The Signal & Scope Manager in the Simulink documentation.

Multitasking Support

If you have a multirate model that you want to run in `MultiTasking` mode and your model contains any of the blocks listed below, your reset event can be delayed as much as one reset time interval so your model behaves deterministically:

- Minimum
- Maximum
- Mean
- Standard Deviation
- Variance
- RMS
- Cumulative Sum
- Cumulative Product
- Delay

To minimize delay in multirate models, run them in `SingleTasking` mode.

Multirate Models

The following blocks no longer support different sample rates at their input ports:

- Permute Matrix
- Variable Selector
- Variable Integer Delay

Scalar Quantizer Block Obsoleted

The Scalar Quantizer block has been replaced by the Scalar Quantizer Encoder and Scalar Quantizer Decoder blocks.

Major Bug Fixes

The Signal Processing Blockset 6.0 includes several bug fixes made since Version 5.1. This section describes the important Version 6.0 bug fixes.

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 issues involved in migrating DSP Blockset Version 2.2 and earlier models to the Signal Processing Blockset Version 6.0 environment.

As of Version 6.0 (Release 14) of the Signal Processing Blockset, DSP Blockset Versions 2.2 (Release 10) and earlier are obsolete and no longer supported. Models that contain blocks from Versions 2.2 and earlier will have broken links when loaded into Simulink 6.0 (Release 14).

If you have models that contain blocks from DSP Blockset Versions 2.2 or earlier, replace the older blocks by blocks from DSP Blockset Versions 4.0 (Release 12) or later before upgrading to Signal Processing Blockset 6.0 (Release 14). Use the command `dsp_links` to facilitate this process.

DSP Blockset Version 3.x (Release 11) might also be obsoleted in a future release.

5

DSP Blockset 5.1 Release Notes

New Features

This section introduces the new features and enhancements added to the DSP Blockset 5.1 since DSP Blockset 5.0 (Release 13).

The new features discussed in this section are:

If you are upgrading from a release earlier than Release 13, then you should see "New Features" on page 6-2 of the Release Notes for DSP Blockset 5.0.

Additional Fixed-Point Support

For this release, significant support for fixed-point development has been added to the DSP Blockset.

Blocks

Fixed-point support has been added to key blocks essential for the modeling of fixed-point DSP systems. These blocks support the processing of signed fixed-point quantities in both simulation and code generation. Supported word sizes are from 2 to 128 bits in simulation, and from 2 to the size of a `long` on the intended target for code generation. The following table lists all DSP Blockset blocks that currently support fixed-point signals. These blocks are colored orange in the DSP Blockset library.

Autocorrelation	Buffer	Check Signal Attributes	Constant Diagonal Matrix
Convert 1-D to 2-D	Convert 2-D to 1-D	Convolution	Correlation

Counter	Create Diagonal Matrix	Data Type Conversion (Simulink block)	Delay Line
Digital Filter	Discrete Impulse	Display (Simulink block)	Downsample
DSP Constant	DSP Fixed-Point Attributes	DSP Gain	DSP Product
DSP Sum	Edge Detector	Event-Count Comparator	Extract Diagonal
Extract Triangular Matrix	FFT	Filter Realization Wizard	FIR Decimation
FIR Interpolation	Flip	Frame Status Conversion	Identity Matrix
IFFT	Integer Delay	Matrix Concatenation (Simulink block)	Matrix Product
Matrix Scaling	Matrix Sum	Matrix Viewer	Maximum
Minimum	Multiphase Clock	Multiport Selector	N-Sample Enable
N-Sample Switch	Overwrite values	Pad	Permute Matrix
Queue	Repeat	Sample and Hold	Selector (Simulink block)
Signal To Workspace	Sine Wave	Spectrum Scope	Stack
Submatrix	Time Scope (Simulink block)	Toeplitz	Transpose
Triggered Delay Line	Triggered to Workspace	Unbuffer	Upsample
Variable Integer Delay	Variable Selector	Vector Scope	Window Function
Zero Pad			

Many of the above blocks that perform arithmetic calculations, such as Digital Filter and FFT, now allow you to specify intermediate fixed-point data types on their block mask. The ability to configure these parameters, such as accumulator and product output data type parameters, allows you to more

closely simulate your target hardware. Refer to Setting Block Parameters in the DSP Blockset User's Guide documentation for more information.

System-Level Control

The new DSP Fixed-Point Attributes (DFPA) block gives you access to a GUI that allows you to set fixed-point parameters for DSP blockset blocks on a system or subsystem level. This allows you to specify fixed-point attributes for large groups of blocks at one time, without having to individually configure the parameters on each block mask. The word and fraction lengths of fixed-point data types, as well as overflow handling and rounding methods, can be set using DFPA blocks. Refer to the DSP Fixed-Point Attributes block reference page in the DSP Blockset documentation for more information.

Extended Adaptive Filter Support

The LMS Filter and RLS Filter blocks have replaced the LMS Adaptive Filter and RLS Adaptive Filter blocks. The LMS Filter block is designed to support five different types of LMS algorithms. In addition, two other blocks, the Block LMS Filter and the Fast Block LMS Filter, have been added to the Adaptive Filters library. All of the new adaptive filter blocks support frame-based processing.

For more information, see the LMS Filter, RLS Filter, Block LMS Filter, and Fast Block LMS Filter block reference pages.

Triggered Subsystem Support

Almost all of the DSP Blockset blocks are now supported in triggered subsystems. Triggered subsystems do not support the following blocks:

- Overlap-Add FFT Filter
- Overlap-Save FFT Filter
- Unbuffer

New Blocks

For descriptions of the new blocks, see the following sections:

- "DSP Fixed-Point Attributes" on page 5-5

DSP Fixed-Point Attributes

The DSP Fixed-Point Attributes block is in the Signal Attributes library. This block sets fixed-point attributes of DSP Blockset blocks on the system or subsystem level.

DSP Gain

The DSP Gain block is in the Math Operations library. This block multiplies the input by a constant. The DSP Gain block has fixed-point support.

DSP Product

The DSP Product block is in the Math Operations library. This block performs element-wise multiplication of two inputs. The DSP Product block has fixed-point support.

DSP Sum

The DSP Sum block is in the Math Operations library. This block adds two inputs. The DSP Sum block has fixed-point support.

LPC to/from RC

The LPC to/from RC block is in the Linear Prediction library. This block converts linear prediction coefficients to reflection coefficients or reflection coefficients to linear prediction coefficients.

LPC/RC to Autocorrelation

The LPC/RC to Autocorrelation block is in the Linear Prediction library. This block converts linear prediction coefficients or reflection coefficients to autocorrelation coefficients.

Matrix Exponential

The Matrix Exponential block is in the Matrix Operations library. It computes the matrix exponential.

Scalar Quantizer

The Scalar Quantizer block is in the Quantizers library. It converts an input signal into a set of index values or a set of quantized output values. It can also convert a set of index values into a quantized output signal.

Scalar Quantizer Design

The Scalar Quantizer Design block is in the Quantizers library. You can use this block to start the Scalar Quantizer Design Tool (SQDTool) to design a scalar quantizer using the Lloyd algorithm.

Enhanced Blocks

The following blocks have improved functionality for DSP Blockset 5.1.

Delay Block

The Delay block has replaced the Integer Delay block in the Signal Operations library. The Delay block is optimized with an improved user-interface that makes it easier to specify initial conditions. In addition, the block has been optimized for computational complexity.

From Wave File Block

The From Wave File Block now supports the repeated output of audio files. You can now play your `.wav` file more than once.

Blocks with Added Fixed-Point Support

These DSP Blockset blocks now support fixed-point data types.

Autocorrelation	Convolution	Correlation
Digital Filter	Event-Count Comparator	FFT
FIR Decimation	FIR Interpolation	IFFT
Matrix Product	Matrix Scaling	Matrix Sum
Window Function		

New and Enhanced Demos

The Adaptive Noise Cancellation Demo was added to the DSP Demos. Run this demo by typing `dspanc` in the MATLAB Command Window. To learn more about this demo, see Creating an Acoustic Environment in the DSP Blockset documentation.

Any demos that contained the LMS Adaptive Filter or RLS Adaptive Filter blocks have been upgraded to use the new LMS Filter and RLS Filter blocks.

New Options for Delay Line Block

You now have the option of either feeding the input through the block directly, or imposing a one sample or one frame delay.

6

DSP Blockset 5.0 Release Notes

New Features

This section introduces the new features and enhancements added to the DSP Blockset 5.0 (Release 13) since DSP Blockset 4.1 (Release 12.1).

Two major new features of DSP Blockset 5.0 include the following:

- Full single-precision support in all blocks
- Full support of the embedded real-time (ERT) generated ANSI C code (requires the Real-Time Workshop Embedded Coder)

For details about these and other new features and enhancements, see the following topics:

The increased support for single-precision computation and ERT code generation, together with other major enhancements, provide an efficient solution for the design and implementation of complete, high-performance floating-point DSP systems.

Full Single-Precision Support

All DSP Blockset blocks now support single-precision floating-point computation in both simulation and Real-Time Workshop C code generation.

To learn more about data type support in the DSP Blockset, see Data Type Support in the DSP Blockset documentation.

Full Support of Embedded Real-Time (ERT) C Code Generation

All DSP Blockset blocks now support embedded real-time (ERT) ANSI C code generation (requires the Real-Time Workshop Embedded Coder).

Smaller, Faster Generated C Code That Requires Less RAM

Real-Time Workshop (RTW) ANSI C code generated from DSP Blockset blocks is now smaller, faster, and requires less RAM, largely due to the following enhancements:

- **Function reuse (run-time libraries)** — The code generated from most blocks now *reuses* common algorithmic functions from ANSI C run-time libraries, leading to smaller generated code. In addition, the functions in the run-time libraries are highly optimized, resulting in faster generated code that requires less RAM.
- **Parameter reuse (RTW run-time parameters)** — For most blocks, if there are multiple instances of a block that all have the same value for a specific parameter, each block instance points to the same variable in the generated code, thereby reducing memory requirements.

Full Boolean Data Type Support

All block input ports that accept logical signals now support the Boolean data type. In addition, for all outputs that are logical signals, the default data type is now Boolean.

For a list of DSP Blockset blocks that support Boolean data types, see Boolean Support in the DSP Blockset documentation.

In some cases, you may want to override the Simulink default and *disable* Boolean support, as described in “New Default Setting Enables Boolean Data Type Support” on page 6-18. For more information about disabling Boolean support, see the following sections in the DSP Blockset documentation:

- Steps to Disabling Boolean Support
- Effects of Enabling and Disabling Boolean Support

Expanding Fixed-Point Data Type Support

The DSP Blockset 5.0 includes more blocks that support fixed-point data types, including some source blocks. Support of fixed-point data types will continue to expand in future releases.

The following blocks are all the blocks in the DSP Blockset that support fixed-point data types. You can use all of these blocks with Fixed-Point Blockset blocks, and with Simulink blocks that support fixed-point data types. These blocks are colored orange in the DSP Blockset library. (Some of the blocks are Simulink blocks that are available in the DSP Blockset libraries as well as the Simulink libraries.)

To take full advantage of DSP Blockset fixed-point capabilities, you must install the Fixed-Point Blockset. For more information, see the topic on licensing information in the Fixed-Point Blockset documentation.

For more information on the DSP Blockset blocks that support fixed-point, see Working with Fixed-Point Data in the DSP Blockset documentation.

- Buffer
- Check Signal Attributes
- Constant Diagonal Matrix
- Convert 1-D to 2-D
- Convert 2-D to 1-D
- Create Diagonal Matrix
- Data Type Conversion (Simulink block)
- Delay Line
- Discrete Impulse
- Display (Simulink block)
- Multiport Selector
- N-Sample Enable
- N-Sample Switch
- Overwrite values
- Pad
- Permute Matrix
- Queue
- Repeat
- Sample and Hold
- Selector (Simulink block)
- Signal To Workspace

- Downsample
- DSP Constant
- Edge Detector
- Extract Diagonal
- Extract Triangular Matrix
- Filter Realization Wizard
- Flip
- Frame Status Conversion
- Identity Matrix
- Inherit Complexity
- Integer Delay
- Matrix Concatenation (Simulink block)
- Matrix Viewer
- Maximum
- Minimum
- Multiphase Clock
- Sine Wave
- Spectrum Scope
- Stack
- Submatrix
- Time Scope (Simulink block)
- Toeplitz
- Transpose
- Triggered Delay Line
- Triggered To Workspace
- Unbuffer
- Upsample
- Variable Integer Delay
- Variable Selector
- Vector Scope
- Zero Pad

New Blocks

The key new block for Release 13 is the Digital Filter block. For descriptions of this and other new blocks and links to their online reference pages, see the following sections:

Cumulative Product

The new Cumulative Product block is in the Math Operations library. This block computes the cumulative product of the row or column elements of the input matrix.

Digital Filter

The new Digital Filter block in the Filter Designs library is ideal for implementing digital filters that you have already designed. The block supports a variety of filter structures, and can implement *static filters* with fixed coefficients, as well as *time-varying filters* with coefficients that change over time. Using the filter you specify, the block individually filters each channel of the input signal and outputs the result. The block's output numerically matches the output of the filter function in the Filter Design Toolbox and the `filter` function in the Signal Processing Toolbox.

You can use the Digital Filter block to efficiently simulate the numerical behavior of your filter on a single- or double-precision floating-point system such as a personal computer or DSP chip. You can also use the block to generate highly optimized Real-Time Workshop ANSI C code for use in single- and double-precision floating-point embedded systems.

To implement a filter with the Digital Filter block, you must provide the following basic information about your filter:

- Whether the filter transfer function is FIR with all zeros, IIR with all poles, or IIR with poles and zeros
- The desired filter structure
- The filter coefficients (entered as a parameter, or from an additional port, which is useful for time-varying filters)

The block supports the following filter structures:

- Direct form
- Direct form I
- Direct form II
- Transposed direct form
- Direct form I transposed
- Direct form II transposed
- Biquadratic direct form II transposed (second order sections)
- Lattice AR
- Lattice MA

DWT and IDWT

The new DWT block in the Transforms library computes the discrete wavelet transform (DWT) of its input. The IDWT block computes the *inverse* DWT of its input. These blocks are identical to the Dyadic Analysis Filter Bank and Dyadic Synthesis Filter Bank blocks in the Multirate Filters library.

The DWT block uses a filter bank with specified highpass and lowpass FIR filters to decompose the input signal into subbands that have smaller bandwidths and slower sample rates. Similarly, the IDWT block uses a filter bank to reconstruct the original signal from the subbands output by the DWT block.

The filter bank filters in both blocks can be user-defined or wavelet-based. Use of wavelet-based filters requires the Wavelet Toolbox.

Interpolation

The new Interpolation block in the Signal Operations library interpolates real-valued input signals at points between sample times by using linear or FIR interpolation. You specify the times at which the block should interpolate points, and the block outputs the interpolated values.

LPC to LSF/LSP Conversion and LSF/LSP to LPC Conversion

The new LPC to LSF/LSP Conversion block and LSF/LSP to LPC Conversion block in the Linear Prediction library are ideal for speech applications such as speech coding and speech recognition. The first block allows you to convert linear prediction coefficients (LPCs) to line spectral pairs (LSPs) or line spectral frequencies (LSFs). The second block converts LSFs or LSPs back to LPCs. When converting LPCs, you can adjust the accuracy of the LSF or LSP output and set the block to flag invalid outputs due to unstable polynomial inputs.

Overwrite Values

The new Overwrite Values block in the Matrix Operations library overwrites either a specified submatrix of the input matrix or a specified portion of the input's main diagonal.

Two-Channel Analysis Subband Filter and Two-Channel Synthesis Subband Filter

The new Two-Channel Analysis Subband Filter block in the Multirate Filters library decomposes a signal into a high-frequency subband and a low-frequency subband using the specified highpass and lowpass FIR filters. Each subband has half the bandwidth and half the sample rate of the original signal.

Similarly, the new Two-Channel Synthesis Subband Filter block uses a filter bank to reconstruct the original signal from the high-frequency and a low-frequency subbands output by the Two-Channel Analysis Subband Filter block.

Enhanced Blocks

For summaries of individual block enhancements for Release 13 and links to their online reference pages, see the following sections:

Audio Blocks — New 24- and 32-Bit Support

The To Wave Device and From Wave Device blocks now support 24-bit audio devices in addition to 8- and 16-bit audio devices.

The To Wave File and From Wave File blocks now support 24 bits per sample and 32 bits per sample in addition to the previous 8- and 16-bit support.

All four audio blocks are now located in a new library. For more information, see "Audio Blocks Relocated" on page 6-21.

Autocorrelation, Correlation, Convolution — New Optimization Options

The Autocorrelation, Correlation, and Convolution blocks used to compute only in the time domain. Now you can set their computation domains to one of the following:

- `Time domain` — Minimizes memory use
- `Frequency domain` — Depending on the input length, may require fewer computations than computing in the time domain

The Correlation and Convolution blocks also have a third computation domain option, `Fastest domain`, which computes in the domain that minimizes the number of computations (time domain or frequency domain). The Autocorrelation block does not yet support this option.

Cumulative Sum — Intraframe Running Sums of Frame-Based Columns

The Cumulative Sum block can now compute a running sum of each column of a frame-based input, independent of the running sum of columns of previous inputs.

DCT and IDCT — New Optimization Options

The DCT and IDCT blocks use sine and cosine values to compute the discrete cosine transform and its inverse. You can now set the blocks to compute sines and cosines in one of the following ways:

- `Table lookup` — Look up sine and cosine values in a speed-optimized table
- `Trigonometric function calls` — Make sine and cosine function calls

Digital Filter Design — Supports More Filter Structures

The Digital Filter Design block, which previously supported only the direct form II transposed structure, now supports all of the following structures:

- Direct form I
- Direct form II
- Direct form I transposed
- Direct form II transposed
- Direct form FIR
- Direct form FIR transposed

Dyadic Analysis Filter Bank and Dyadic Synthesis Filter Bank Enhancements

The Dyadic Analysis Filter Bank block and Dyadic Synthesis Filter Bank block (referred to as "analysis block" and "synthesis block") share several changes and enhancements described below.

New Wavelet Option. Both blocks now allow you to specify wavelet-based highpass and lowpass filters in addition to the previously supported user-defined filters. Use of wavelets requires the Wavelet Toolbox.

New Option for Single-Port Inputs and Outputs. Previously, the analysis block output each subband on a separate output port and the synthesis block accepted each subband through a separate input port. Now the analysis block provides an option to output a single vector or matrix of concatenated subbands, and the synthesis block provides an option to accept such inputs through a single input port.

Frame-Based Support Only. The analysis block now accepts only frame-based input signals. The synthesis block now always outputs frame-based signals. To decompose or reconstruct a sample-based signal using filter banks, you can create your own filter banks using the new Two-Channel Analysis Subband Filter and Two-Channel Synthesis Subband Filter blocks in the Multirate Filters library (see “Two-Channel Analysis Subband Filter and Two-Channel Synthesis Subband Filter” on page 6-8).

Filter Realization Wizard — New Filter Design Options, Better Fixed-Point Support, More Structures

The Filter Realization Wizard block has several significant enhancements described below.

New Interface Provides Filter Design and Analysis Options. The Filter Realization Wizard block interface is now a part of the Filter Design and Analysis Tool (FDATool) GUI. Previously, the block required you to specify the filter by typing in its coefficients (you had to predesign the filter elsewhere). You now have the option to design your filter within the block; the extensive set of filter design and analysis tools in FDATool allow the block to automatically implement your filter design without you having to type its coefficients.

Enhanced Fixed-Point Support. The Filter Realization Wizard block now better supports fixed-point filters due to the following enhancements:

- **New fixed-point filter design capabilities** — The block now allows you to design fixed-point filters by using the **Set Quantization Parameters** panel in the new block interface. Use of this panel requires the Filter Design Toolbox.
- **Better fixed-point filter implementation** — The block now implements a much better fixed-point filter when you install the Fixed Point Blockset and Filter Design Toolbox. To implement a good fixed-point filter, you should first design a fixed-point filter as described above and then set the block to implement the filter using Fixed-Point Blockset blocks.

Corresponding dfilt and qfilt Methods. `dfilt` (dfilt.calattice and dfilt.calatticepc) and `qfilt` objects now have a new `realizemdl` method. This method allows you to access the filter realization capabilities of the Filter Realization Wizard block from the command line.

Supports More Filter Structures. The block can now realize filters using any of the following filter structures:

- Direct form I
- Direct form II
- Direct form I transposed
- Direct form II transposed
- Second order sections for direct form I and II, and their transposes
- Direct form FIR
- Direct form FIR transposed
- Direct form asymmetric FIR
- Direct form symmetric FIR
- Lattice ARMA
- Lattice AR
- Lattice MA (same as lattice minimum phase)
- Lattice all-pass
- Lattice maximum phase
- Cascade
- Parallel

You may not be able to directly access some of the above structures through the block interface, but you *can* access all of them by creating a `qfilt` or `dfilt` object with the desired structure, and then importing the filter into the block.

Random Source — Better Random Signals and New Option

The Random Source block now outputs much better random signals due to the following algorithmic enhancements:

- **Uses improved random generators** — The block now uses the same random generator algorithms as the current `rand` and `randn` functions. These generators yield much better random signals than the MATLAB Version 4 generators that were previously used by the Random Source block.
- **Better automatic seed generation** — When you opt to specify a single initial seed for a multichannel output, the block automatically generates an initial seed for each channel (using the seed you provide). The block now generates these initial seeds using an improved method that results in better random signals.
- **Better complex random signal generation** — The block now generates complex random values using an improved method that results in better complex random signals.

The Random Source block also now offers a second method of computing Gaussian (normally distributed) random signals based on the central limit theorem. When set to use this new method, the block computes Gaussian random values by adding and scaling uniformly distributed random signals. The block allows you to specify the number of uniform values to sum.

Sine Wave — Accepts Zero and Negative Frequencies

The Sine Wave block now accepts zero and negative frequency values in addition to positive frequency values.

New and Enhanced Demos

You can access all DSP Blockset demos (in the Help browser **Demo** tab) by typing `demo blockset dsp` at the command line. To open the new and

enhanced demos, you can also click the links in the following table in the MATLAB Help browser (*not* in a Web browser).

New and Enhanced Demos	Location in DSP Blockset Entry of Help Browser Demo Tab	Enhancement Description
GSM Digital Down Converter	`Communications`	New
FIR Interpolation	`Filtering`	These demos now provide a fixed-point version that you can run when you install the Fixed-Point Blockset.
Denoising	`Wavelets`	
Wavelet Transmultiplexer (WTM)	`Wavelets`	
Multi-Level PR Filter Bank	`Wavelets`	Formerly a 1-level filter bank demo, this demo now demonstrates a three-level perfect reconstruction wavelet-based filter bank. This demo also provides a new fixed-point version that you can run when you install the Fixed-Point Blockset.

Audio Blocks Relocated to New Block Library

The four audio blocks, formerly in the DSP Sources and DSP Sinks libraries, are now in the new Windows (WIN32) library (a sublibrary of the new Platform-specific I/O library).

New DSP Blockset Block Libraries

- **Platform-specific I/O** — New block library that contains sublibraries for use with specific platforms such as the Windows operating system. To see the library, type `dsppio`, or type `simulink` and navigate to the library using the Simulink Library Browser.
- **Windows (WIN32)** — Sublibrary of the Platform-specific I/O library. Contains blocks (listed below) for use with 32-bit Windows operating

systems. To see the library, type `dspwin32`, or type `simulink` and navigate to the library using the Simulink Library Browser.

Blocks Relocated to the Windows (WIN32) Library

- To Wave File
- To Wave Device
- From Wave File
- From Wave Device

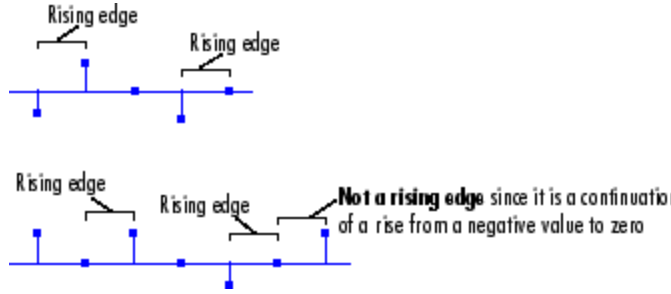
New Options for Event Detection

The following blocks perform an operation when an event-detecting input port (such as a clock or reset port) detects an event:

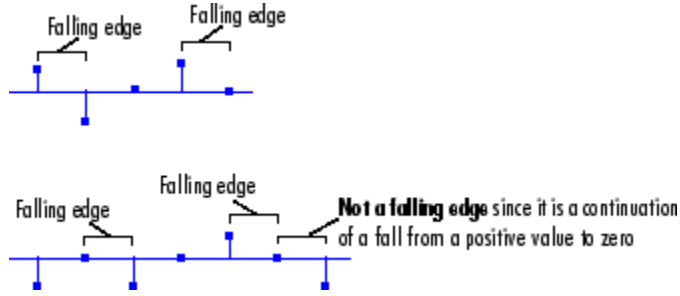
- Counter
- Cumulative Product
- Cumulative Sum
- Histogram
- Integer Delay
- Maximum
- Mean
- Minimum
- N-Sample Enable
- Queue
- RMS
- Stack
- Standard Deviation
- Variance

You can now set the event-detecting ports of the above blocks to respond to one of the following events:

- `Non-zero sample` — When the input sample is nonzero
- `Rising edge` — When the input does one of the following:
  - Rises from a negative value to a positive value or zero
  - Rises from zero to a positive value, where the rise is not a continuation of a rise from a negative value to zero (see the following figure)



- `Falling edge` — When the input does one of the following:
  - Falls from a positive value to a negative value or zero
  - Falls from zero to a negative value, where the fall is not a continuation of a fall from a positive value to zero (see the following figure)



- `Either edge` — When an input is a `Rising edge` or `Falling edge` (as described above)

Major Bug Fixes

The DSP Blockset 5.0 includes several bug fixes made since Version 4.1. This section describes the particularly important Version 5.0 bug fixes.

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

The following topics describe the upgrade issues involved in moving from the DSP Blockset 4.1 to Version 5.0:

New Default Setting Enables Boolean Data Type Support

The Simulink default settings now *enable* Boolean data type support. This allows you to take advantage of the full Boolean data type support in DSP Blockset 5.0.

In some cases, you may want to override the Simulink default and *disable* Boolean support. For instance, for each of the following blocks, the default data type of at least one output has now changed from double precision to Boolean:

- Counter
- Edge Detector
- Event-Count Comparator
- Multiphase Clock
- N-Sample Enable
- Queue
- Stack

If you have a model that uses previous versions of the above blocks, their new default Boolean data type can potentially break your model. If the change

in data type does break your model, you can fix this by disabling Boolean support; then the output data type of the blocks will remain double precision, and your model will behave just as it did before.

For more information about disabling Boolean support, see the following topics in the DSP Blockset documentation:

- Steps to Disabling Boolean Support
- Effects of Enabling and Disabling Boolean Support

For more information about the new Boolean data type support in DSP Blockset 5.0, see the following topics:

- "Full Boolean Data Type Support" on page 6-3 — Description of the new full support of the Boolean data type in DSP Blockset 5.0.
- A list of all DSP Blockset blocks that support Boolean data types in the DSP Blockset documentation.

Replaced Filtering Blocks

The new Digital Filter block in the Filter Designs library replaces the following blocks from DSP Blockset 4.1:

- Biquadratic Filter
- Time-Varying Lattice Filter
- Time-Varying Direct-Form II Transpose Filter

Your models that contain these replaced blocks will still work, and you can still access these blocks by typing `dsparch3` at the MATLAB command line.

However, when creating new models, use the new Digital Filter block, which can implement various filters including those that the above three blocks implement: biquadratic direct form II transposed filters, time-varying lattice filters, and time-varying direct form II transposed filters.

See "Digital Filter" on page 6-6 for more information about the new Digital Filter block.

Wavelet Analysis and Wavelet Synthesis Blocks Replaced

The Wavelet Analysis and Wavelet Synthesis blocks, formerly in the Multirate Filters library, are now replaced by new or enhanced blocks as summarized in the following table.

New or Enhanced Block	Replaces...
Dyadic Analysis Filter Bank	Wavelet Analysis block in frame-based operation
Dyadic Synthesis Filter Bank	Wavelet Synthesis block in frame-based operation
Two-Channel Analysis Subband Filter	Wavelet Analysis block in sample-based or frame-based operation
Two-Channel Synthesis Subband Filter	Wavelet Analysis block in sample-based or frame-based operation

You can still access the Wavelet Analysis and Wavelet Synthesis blocks by typing `dspmlti3` at the command line.

For more information about the new and enhanced replacement blocks, see the following topics:

- "Two-Channel Analysis Subband Filter and Two-Channel Synthesis Subband Filter" on page 6-8
- "Dyadic Analysis Filter Bank and Dyadic Synthesis Filter Bank Enhancements" on page 6-11

Cumulative Sum Block Behaves Differently

The Cumulative Sum block now behaves differently when given frame-based inputs and set to sum along columns. In this situation, the block now computes the running sum of each column of a frame-based input, where the running sum is independent of the running sums of previous inputs.

To get the previous behavior for frame-based inputs when set to sum along columns, set the block to sum along channels.

Contiguous Copy Block Obsolete

The Contiguous Copy block, formerly in the Signal Attributes library, is now obsolete. Your models that currently use the block will still work, but you can no longer access the block in the DSP Blockset 5.0 libraries.

Audio Blocks Relocated

The following blocks, formerly in the DSP Sources and DSP Sinks libraries, are now in the new Windows (WIN32) library:

- From Wave Device
- From Wave File
- To Wave Device
- To Wave File

For more information, see “Audio Blocks Relocated to New Block Library” on page 6-14.