Signal Processing Blockset Release Notes

Contents

Summary by Version 1

Version 6.6 (R2007b) Signal Processing Blockset 4

Version 6.5 (R2007a) Signal Processing Blockset 11

Version 6.4 (R2006b) Signal Processing Blockset 15

Version 6.3 (R2006a) Signal Processing Blockset 19

Version 6.2 (R14SP3) Signal Processing Blockset 22

Version 6.1 (R14SP2) Signal Processing Blockset 24

Version 6.0.1 (R14SP1) Signal Processing Blockset 33

Version 6.0 (R14) Signal Processing Blockset 35

Version 5.1 (R13SP1) DSP Blockset 46

Version 5.0 (R13) DSP Blockset 52

Compatibility Summary for Signal Processing Blockset 69

Summary by Version

This table provides quick access to what's new in each version. For clarification, see "About Release Notes" on page 2.

Version (Release)	New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Latest Version V6.6 (R2007b)	Yes Details	Yes Summary	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation
V6.5 (R2007a)	Yes Details	Yes Summary	Bug Reports Includes fixes	No
V6.4 (R2006b)	Yes Details	Yes Summary	Bug Reports Includes fixes	No
V6.3 (R2006a)	Yes Details	No	Bug Reports Includes fixes	No
V6.2 (R14SP3)	Yes Details	No	Bug Reports Includes fixes	No
V6.1 (R14SP2)	Yes Details	Yes Summary	Bug Reports Includes fixes	No
V6.0.1 (R14SP1)	Yes Details	No	Fixed bugs	No
V6.0 (R14)	Yes Details	Yes Summary	Fixed bugs	No
V5.1 (R13SP1)	Yes Details	No	No bug fixes	No
V5.0 (R13)	Yes Details	Yes Summary	Fixed bugs	No

About Release Notes

Use release notes when upgrading to a newer version to learn about new features and changes, and the potential impact on your existing files and practices. Release notes are also beneficial if you use or support multiple versions.

If you are not upgrading from the most recent previous version, review release notes for all interim versions, not just for the version you are installing. For example, when upgrading from V1.0 to V1.2, review the New Features and Changes, Version Compatibility Considerations, and Bug Reports for V1.1 and V1.2.

New Features and Changes

These include

- New functionality
- Changes to existing functionality
- Changes to system requirements (complete system requirements for the current version are at the MathWorks Web site)
- Any version compatibility considerations associated with each new feature or change

Version Compatibility Considerations

When a new feature or change introduces a reported incompatibility between versions, its description includes a **Compatibility Considerations** subsection that details the impact. For a list of all new features and changes that have reported compatibility impact, see the "Compatibility Summary for Signal Processing Blockset" on page 69.

Compatibility issues that are reported after the product has been released are added to Bug Reports at the MathWorks Web site. Because bug fixes can sometimes result in incompatibilities, also review fixed bugs in Bug Reports for any compatibility impact.

Fixed Bugs and Known Problems

MathWorks Bug Reports is a user-searchable database of known problems, workarounds, and fixes. The MathWorks updates the Bug Reports database as new problems and resolutions become known, so check it as needed for the latest information.

Access Bug Reports at the MathWorks Web site using your MathWorks Account. If you are not logged in to your MathWorks Account when you link to Bug Reports, you are prompted to log in or create an account. You then can view bug fixes and known problems for R14SP2 and more recent releases.

The Bug Reports database was introduced for R14SP2 and does not include information for prior releases. You can access a list of bug fixes made in prior versions via the links in the summary table.

Related Documentation at Web Site

Printable Release Notes (PDF). You can print release notes from the PDF version, located at the MathWorks Web site. The PDF version does not support links to other documents or to the Web site, such as to Bug Reports. Use the browser-based version of release notes for access to all information.

Product Documentation. At the MathWorks Web site, you can access complete product documentation for the current version and some previous versions, as noted in the summary table.

Version 6.6 (R2007b) Signal Processing Blockset

This table summarizes what's new in Version 6.6 (R2007b):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes Summary	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation

New features and changes introduced in this version are

- "New To Audio Device and From Audio Device Blocks" on page 4
- "New Array-Vector Math Blocks" on page 5
- "New CIC Filter Block" on page 5
- "FFT and IFFT Blocks Are More Optimized for Fixed-Point Signals" on page 6
- "Rounding Modes Ceiling and Zero Added to Fixed-Point Blocks" on page 6
- "Increased N-Dimensional Support" on page 6
- "Increased Scaled Doubles Support" on page 6
- "Increased Multichannel Support" on page 7
- "DirectX Component Registration Limitations Removed from To Multimedia File and From Multimedia File Blocks" on page 7
- "Tunability Status Changed for Some Block Parameters" on page 7
- "Levinson-Durbin Block Now Treats Frame-Based Row Vectors Differently" on page 9

New To Audio Device and From Audio Device Blocks

The From Audio Device and To Audio Device blocks have been added to the Signal Processing Sources and Signal Processing Sinks libraries, respectively.

These blocks offer support for more than two audio channels and for Windows, Macintosh, and Linux platforms. See the block reference pages for more information.

Compatibility Considerations

These blocks replace the To Wave Device and From Wave Device blocks, which are obsolete as of this release, and might be completely removed from the product in a future release. Replace To Wave Device and From Wave Device blocks in your models with the new To Audio Device and From Audio Device blocks.

New Array-Vector Math Blocks

The following new array-vector math blocks perform arithmetic operations along a specified dimension of an N-dimensional array:

- Array-Vector Add
- Array-Vector Divide
- Array-Vector Multiply
- Array-Vector Subtract

See the block reference pages for more information.

Note The Array-Vector Multiply block replaces the Matrix Scaling block, which is removed from the product as of this release. Matrix Scaling blocks in your existing models will be automatically replaced with Array-Vector Multiply blocks.

New CIC Filter Block

The CIC Filter block has been added to the Filter Design Toolbox library. See the block reference page for more information.

FFT and IFFT Blocks Are More Optimized for Fixed-Point Signals

The double-signal and half-length optimizations that the FFT and IFFT blocks used to apply only to floating-point signals now also apply to fixed-point signals. See “Algorithms Used for FFT Computation” and “Algorithms Used for IFFT Computation” in the respective block reference pages for more information.

Rounding Modes Ceiling and Zero Added to Fixed-Point Blocks

The **Rounding Mode** parameter of each fixed-point-capable block has two new rounding modes:

- `Ceiling` rounds the result of a calculation to the closest representable number in the direction of positive infinity.
- `Zero` rounds the result of a calculation to the closest representable number in the direction of zero.

Increased N-Dimensional Support

The following blocks now have support for N-D signals:

- Array-Vector Add
- Array-Vector Divide
- Array-Vector Multiply
- Array-Vector Subtract
- Constant Ramp
- Difference
- Inherit Complexity
- Maximum
- Minimum

Increased Scaled Doubles Support

The following blocks now support the scaled doubles data type:

- Difference
- Normalization
- Matrix Product
- Matrix Sum

Increased Multichannel Support

The following blocks now support multichannel signals:

- LPC to LSF/LSP Conversion
- LPC to/from Cepstral Coefficients
- LPC to/from RC
- LPC/RC to Autocorrelation

DirectX Component Registration Limitations Removed from To Multimedia File and From Multimedia File Blocks

You are now able to use the From Multimedia File or To Multimedia File blocks without first having someone with system administrator privileges register the DirectX components associated with these blocks on your Windows machine.

Tunability Status Changed for Some Block Parameters

The tunability status for the block parameters in the following table has been changed. This was done to maintain consistency of the tunability status for any given parameter across all simulation and code generation modes.

Block	Parameter	Old Tunability Status	New Tunability Status
Chirp	**Frequency sweep**	Simulation only	Never
	Initial frequency	Simulation only	Always
	Target frequency	Simulation only	Always
Digital Filter	**SOS matrix**	Simulation only	Always
	Scale values	Simulation only	Always
Extract Triangular Matrix	**Extract**	Simulation only	Never
Histogram	**Normalized**	Simulation only	Never
Multiphase Clock	**Starting phase**	Always	Never
	Number of phase intervals over which clock is active	Simulation only	Never
	Active level	Always	Never
Normalization	**Norm**	Simulation only	Never
	Normalization bias	Simulation only	Always

Block	Parameter	Old Tunability Status	New Tunability Status
Sine Wave	**Frequency**	In some modes	Always when **Computation method** is `Trigonometric fcn` or `Differential`
	Phase offset	In some modes	Always when **Computation method** is `Trigonometric fcn` or `Differential`
Sort	**Sort order**	Simulation only	Never

Compatibility Considerations

Due to these changes, some parameters that were previously tunable during simulation are no longer tunable. To change these parameters while you are working with a model, you now have to stop a running simulation, change the parameter, and then start the simulation again.

Levinson-Durbin Block Now Treats Frame-Based Row Vectors Differently

The Levinson-Durbin block now treats a 1-by-N frame-based row vector on its input port as N channels with one sample each. Previously, the Levinson-Durbin block treated such an input as one channel with N samples. This change makes the Levinson-Durbin block consistent with the way most Signal Processing Blockset blocks treat frame-based row vectors.

Be aware that the block now errors for a 1-by-N frame-based row vector input when reflection coefficients (K) are output, since the block is required to have at least 2 samples per input channel to calculate K.

Compatibility Considerations

To get the old behavior in an existing model, you can introduce a Frame Conversion block before a Levinson-Durbin block in your model to convert the block input to a sample-based signal.

Version 6.5 (R2007a) Signal Processing Blockset

This table summarizes what's new in Version 6.5 (R2007a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes Summary	Bug Reports Includes fixes	No

New features and changes introduced in this version are

- "R11.1 Blocks Have Been Removed in R2007a — Run Helper Script Before Upgrading" on page 11
- "New Kalman Adaptive Filter Block" on page 12
- "Increased Unsigned Integer and Fixed-Point Support" on page 12
- "Increased N-Dimensional Support" on page 12
- "X-Axis Control Added to Spectrum Scope and Vector Scope Blocks" on page 13
- "New Filter Design Toolbox Library Blocks" on page 13
- "Fixed-Point Support and Tunability Added to Filter Design Toolbox Library Blocks" on page 13
- "New FFT Length Parameters on FFT and IFFT Blocks" on page 13
- "Zero Pad Block Removed" on page 14
- "Pad Block Can Truncate Either End of an Input Signal" on page 14
- "New and Updated Demos" on page 14

R11.1 Blocks Have Been Removed in R2007a — Run Helper Script Before Upgrading

The R11.1 DSP Blockset blocks have been deprecated since R14SP2. These blocks have been completely removed from Signal Processing Blockset in R2007a.

Compatibility Considerations

We are providing a script and documentation to help you remove any R11.1 DSP Blockset blocks from your models and replace them with current Signal Processing Blockset blocks. You must run this script *before* upgrading to R2007a. Refer to our MATLAB Central submission titled "Tool for Removing R11 DSP Blockset Blocks from Models" on the Web to download the script and its associated documentation.

New Kalman Adaptive Filter Block

The Kalman Adaptive Filter block has been added to the Filtering > Adaptive Filters library. This block predicts or estimates the state of a dynamic system from a series of incomplete or noisy measurements. See the block reference page for more information.

Increased Unsigned Integer and Fixed-Point Support

Unsigned integer and fixed-point data type support has been added to the following blocks:

- Cumulative Product
- Cumulative Sum
- Difference
- FIR Decimation
- FIR Interpolation
- FIR Rate Conversion

Increased N-Dimensional Support

Support for N-D signals has been added to the following blocks:

- dB Conversion
- dB Gain
- Check Signal Attributes
- Frame Conversion
- Normalization

- Pad

X-Axis Control Added to Spectrum Scope and Vector Scope Blocks

More *x*-axis control has been added to the Spectrum Scope and Vector Scope blocks:

- You can now specify the range of the *x*-axis for the Spectrum Scope and Vector Scope blocks.
- You can now specify an *x*-offset for the Vector Scope block.

See the block reference pages for more information.

New Filter Design Toolbox Library Blocks

The following blocks have been added to the Filter Design Toolbox library:

- Arbitrary Magnitude Filter
- Octave Filter
- Parametric Equalizer
- Peak-Notch Filter

See the block reference pages for more information.

Fixed-Point Support and Tunability Added to Filter Design Toolbox Library Blocks

The blocks in the Filter Design Toolbox library now support fixed-point and integer data types on their input and output ports. In addition, parameters of these blocks that do not change filter order or structure are now tunable.

New FFT Length Parameters on FFT and IFFT Blocks

The **Inherit FFT length from input dimensions** and **FFT length** parameters have been added to the FFT and IFFT blocks. See the block reference pages for more information.

Zero Pad Block Removed

The Zero Pad block has been removed from Signal Processing Blockset.

Compatibility Considerations

You can use the Pad block with the **Pad value** parameter set to `0` to exactly replicate the functionality of the Zero Pad block. Any Zero Pad blocks in existing models will be automatically replaced by Pad blocks with the **Pad value** parameter set to `0`. Your models will continue to work correctly.

Pad Block Can Truncate Either End of an Input Signal

You can use the Pad block to truncate a signal by specifying an output length that is shorter than the input length in a given dimension. In previous releases, the block ignored the value of the **Pad signal at** parameter and always truncated the end of a signal.

Compatibility Considerations

The Pad block now obeys the **Pad signal at** parameter for truncation as well as for padding, enabling you to truncate a signal at its beginning, end, or both. To get the previous behavior, make sure that the **Pad signal at** parameter is set to `End` for any Pad blocks in your model that are truncating the input signal.

New and Updated Demos

The Vorbis Decoder demo has been added to the Audio Processing library. This demo implements the Vorbis decoder, which is a freeware, open-source alternative to the MP3 standard. This audio decoding standard supports the segmentation of encoded data into small packets for network transmission. Open this demo by typing `dspvorbisdec`.

The Internet Low Bit-Rate Codec (iLBC) demo in the Audio Processing library has been improved. This demo now supports single-precision floating-point data, and both builds and runs faster. Open this demo by typing `dspilbc`.

Version 6.4 (R2006b) Signal Processing Blockset

This table summarizes what's new in Version 6.4 (R2006b):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes Summary	Bug Reports Includes fixes	No

New features and changes introduced in this version are

- "R11.1 Blocks Will Be Removed in R2007a" on page 15
- "New Filter Design Toolbox block Library" on page 16
- "More Blocks with Fixed-Point Support" on page 16
- "From Multimedia File Block Supports Uncompressed AVI Files on UNIX" on page 17
- "To Wave File and From Wave File Blocks Extended to Support More than Two Channels" on page 17
- "Enabled Subsystem Support for From Wave File Block" on page 17
- "Diagnostic Output Port Added to Report a Failure to Converge" on page 17
- "2-D Support Added" on page 18
- "Multichannel Support Added" on page 18
- "Blocks Removed from Product" on page 18

R11.1 Blocks Will Be Removed in R2007a

The R11.1 Signal Processing Blockset blocks have been deprecated since R14SP2. In the next release, R2007a, these blocks will be completely removed from the product.

Compatibility Considerations

We strongly recommend that you replace any R11.1 blocks that you are using in your models at this time. For more information, refer to "Obsolete Blocks" on page 25.

New Filter Design Toolbox block Library

A new Filter Design Toolbox block library has been added for the design and implementation of single- and multirate FIR and IIR filters. The library contains the following blocks:

- Bandpass Filter
- Bandstop Filter
- CIC Compensator
- Differentiator Filter
- Fractional Delay Filter
- Halfband Filter
- Highpass Filter
- Hilbert Filter
- Inverse Sinc Filter
- Lowpass Filter
- Nyquist Filter

More Blocks with Fixed-Point Support

Support for fixed-point data types has been added to the following blocks:

- Backward Substitution
- Forward Substitution
- LDL Factorization
- LU Factorization

From Multimedia File Block Supports Uncompressed AVI Files on UNIX

The From Multimedia File block now supports uncompressed AVI files on UNIX platforms. As a result, you no longer need to use separate blocks to import multimedia files if you are working on both Windows and UNIX platforms.

To Wave File and From Wave File Blocks Extended to Support More than Two Channels

The To Wave File and From Wave File blocks now support an arbitrary number of audio channels, instead of just mono and stereo.

Enabled Subsystem Support for From Wave File Block

The From Wave File block now supports enabled subsystems.

Diagnostic Output Port Added to Report a Failure to Converge

A new diagnostic output port has been added to the following blocks to report a failure to converge:

- Pseudoinverse
- Singular Value Decomposition
- SVD Solver

To make this port appear, select the **Show error status port** check box on the block dialog.

Compatibility Considerations

In prior releases, these blocks returned an error when the computation failed to converge. This error no longer occurs. Instead, select the **Show error status port** check box on the block dialog to make the error port E appear. You can then connect this port to a block such as the Simulink Assertion block to receive information about the convergence of the output.

2-D Support Added

2-D support has been added to the following blocks:

- Matrix Product
- Matrix Sum
- Maximum
- Minimum

Multichannel Support Added

Multichannel support has been added to the following blocks:

- Autocorrelation LPC
- Levinson-Durbin
- LSF/LSP to LPC Conversion
- Yule-Walker AR Estimator
- Zero Crossing

Blocks Removed from Product

The DSP Gain, DSP Sum, DSP Product, and DSP Fixed-Point Attributes blocks have been removed from Signal Processing Blockset.

Compatibility Considerations

You can replace any DSP Gain, DSP Sum, and DSP Product blocks in your models with Simulink Gain, Sum, and Product blocks, respectively. There is no replacement for the DSP Fixed-Point Attributes block.

Version 6.3 (R2006a) Signal Processing Blockset

This table summarizes what's new in Version 6.3 (R2006a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	No	Bug Reports Includes fixes	No

New features and changes introduced in this version are

- "Integration of Filter Blocks with Signal Processing Toolbox Filter Objects and FVTool" on page 19
- "Transposed Direct Form Structure Added to FIR Decimation Block" on page 20
- "Data Type Specification Modes Added to CIC Decimation and CIC Interpolation Blocks" on page 20
- "Taylor Window Type Added to Window Function Block" on page 21
- "Reduced Simulation Memory Footprint for Fixed-Point Capable Blocks" on page 21
- "Improved Usability for the To Wave Device Block" on page 21
- "New Demos" on page 21

Integration of Filter Blocks with Signal Processing Toolbox Filter Objects and FVTool

Significant enhancements were made to the following filter blocks for this release:

- CIC Decimation
- CIC Interpolation
- FIR Decimation
- FIR Interpolation

- FIR Rate Conversion

The changes made to these blocks bring them into closer alignment with Signal Processing Toolbox:

- These filter blocks can now operate in two different modes, which you select in the **Coefficient source** group box. If you select **Dialog parameters**, you enter information about the filter in the block mask. If you select **Multirate filter object (MFILT)**, you can now specify the filter using a `mfilt` object from Signal Processing Toolbox.
- You can now open the Signal Processing Toolbox `fvtool` from the block masks to view the filter response.

A few minor changes have also been made to the Digital Filter block mask to bring it into closer alignment with these blocks and with Signal Processing Toolbox. However, most of the updates to this block for this improvement were made in the previous release. See "Digital Filter Block Enhancements" on page 22.

Transposed Direct Form Structure Added to FIR Decimation Block

You can now implement either a transposed direct form or a direct form structure with the FIR Decimation block using the **Filter structure** parameter.

The addition of the transposed direct form structure to this block brings it into closer alignment to the Signal Processing Toolbox `mfilt.firdecim` object.

Data Type Specification Modes Added to CIC Decimation and CIC Interpolation Blocks

The **Data type specification mode** parameter has been added to the CIC Decimation and CIC Interpolation blocks. This parameter allows you to choose how the word and fraction lengths are specified for the filter sections and outputs. You can choose to fully specify the word and fraction lengths of the filter sections and outputs yourself, or have one or more of these quantities automatically selected for you.

This feature brings these blocks into closer alignment with the Signal Processing Toolbox `mfilt.cicdecim` and `mfilt.cicinterp` objects.

Taylor Window Type Added to Window Function Block

The Taylor window type has been added to the Window Function block. The block functionality in this mode is identical to that of the Signal Processing Toolbox `taylorwin` function.

Reduced Simulation Memory Footprint for Fixed-Point Capable Blocks

Fixed-point capable blocks in Signal Processing Blockset now use less memory as they simulate. There is no change to the memory requirements for the generated code from these blocks.

Improved Usability for the To Wave Device Block

The usability of the To Wave Device block has been improved with the addition of the **Automatically determine internal buffer size** and **User-defined internal buffer size** parameters. These parameters allow you to define the size of the chunks of data that are written to the hardware audio device by the block, independently of the input dimensions. The block reference page in the documentation also has significant updates, including a "Troubleshooting" section. Refer to the reference page for more information.

New Demos

Demo Name	Signal Processing Demo Library Location	Launch Command
DTMF Generator and Receiver	Communications	`dspdtmf`
Envelope Detection	Miscellaneous	`dspenvdet`
Internet Low Bitrate Codec (iLBC)	Audio Processing	`dspilbc`

Version 6.2 (R14SP3) Signal Processing Blockset

This table summarizes what's new in Version 6.2 (R14SP3):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	No	Bug Reports Includes fixes	No

New features and changes introduced in this version are

- "New Numerically Controlled Oscillator (NCO) Block" on page 22
- "Digital Filter Block Enhancements" on page 22
- "Fixed-Point Support Added to the Matrix Multiply Block" on page 23
- "Simulink Virtual Bus Support Added to Key Blocks" on page 23
- "New Audio Sample Rate Conversion Demo" on page 23

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 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 “Using 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**.

Version 6.1 (R14SP2) Signal Processing Blockset

This table summarizes what's new in Version 6.1 (R14SP2):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports Includes fixes	No

New features and changes introduced in this version are

- "Broader Support for the Logging of Simulation Minimums and Maximums and Fixed-Point Autoscaling" on page 24
- "Fixed-Point Support for the DCT and IDCT Blocks" on page 24
- "New Audio File Source and Sink Blocks" on page 25
- "Multirate Support for CIC Filter Blocks" on page 25
- "Obsolete Blocks" on page 25

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

An increased number of fixed-point capable blocks in Signal Processing Blockset now support the logging of simulation minimums and maximums and autoscaling via the Fixed-Point Settings interface.

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.

Obsolete Blocks

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 Signal Processing Blockset. We recommend that you use the replacement blocks listed in the third column.

Compatibility Considerations

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.

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`

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
Buffered FFT Frame Scope	dspsnks2	Spectrum Scope	dspsnks4
Burg AR Estimator	dspparest2	Burg AR Estimator	dspparest3
Burg Method	dspspect2	Burg Method	dspspect3
Chirp	dspsrcs2	Chirp	dspsrcs4
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	dspparest2	Covariance AR Estimator	dapparest3
Covariance Method	dspspect2	Covariance Method	dspspect3
Create Diagonal Matrix	dspmtrx2	Create Diagonal Matrix	dspmtrx3
Cumulative Sum	dspvect2	Cumulative Sum	dspmathops

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
Counter	`dspswit2`	Counter	`dspswit3`
dB	`dspelem2`	dB Conversion	`dspmathops`
dB Gain	`dspelem2`	dB Gain	`dspmathops`
DCT	`dspxfrm2`	DCT	`dspxfrm3`
Detrend	`dspbdsp2`	Detrend	`dspstat3`
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`

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
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
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

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
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
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	`dspparest2`	Modified Covariance AR Estimator	`dspparest3`
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`

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
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
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

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
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`
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`

Obsolete (R11.1) Block	Obsolete Block Library	Replacement Block(s)	Replacement Block(s) Library
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`
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`

Version 6.0.1 (R14SP1) Signal Processing Blockset

This table summarizes what's new in Version 6.0.1 (R14SP1):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	No	Fixed bugs	No

New features and changes introduced in this version are

- "Changes from the Previous Release" on page 33
- "New Demos" on page 34
- "Enhanced Blocks" on page 34

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 "Changed Source Dialog Box Behavior" section in the Simulink Release Notes.

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

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.

Version 6.0 (R14) Signal Processing Blockset

This table summarizes what's new in Version 6.0 (R14):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Fixed bugs	No

New features and changes introduced in this version are

- "Product Name Change" on page 36
- "Additional Fixed-Point Support" on page 36
- "New Blocks" on page 38
- "Enhanced Blocks" on page 40
- "Renamed Blocks" on page 41
- "New Demos" on page 42
- "Triggered Subsystem Support" on page 42
- "Constant Sample Time Support" on page 43
- "Source Blocks Obey New Simulink Inherited Sample Time Parameter" on page 43
- "Signal & Scope Manager Support" on page 43
- "Multitasking Support" on page 44
- "Multirate Models" on page 44
- "Scalar Quantizer Block Obsoleted" on page 44
- "Obsolete Product Versions" on page 45

Product Name Change

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

Additional Fixed-Point Support

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

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 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 Signal Processing Blockset now support the logging of simulation minimums and maximums and autoscaling via the Fixed-Point Settings interface.

New Blocks

This section gives a brief description of each of the new blocks.

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

This section gives a brief description of each of the block enhancements.

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 Periodogram demo.

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

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
- Multiphase Clock
- Sine Wave
- Blocks with multiple sample times

Constant Sample Time Support

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. 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.

Obsolete Product Versions

As of Version 6.0 (Release 14) of Signal Processing Blockset, DSP Blockset Versions 2.2 (Release 10) and earlier are obsolete and no longer supported. DSP Blockset Version 3.x (Release 11) might also be obsoleted in a future release.

Compatibility Considerations

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.

Version 5.1 (R13SP1) DSP Blockset

This table summarizes what's new in Version 5.1 (R13SP1):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	No	No bug fixes	No

New features and changes introduced in this version are

- "Additional Fixed-Point Support" on page 46
- "Extended Adaptive Filter Support" on page 48
- "Triggered Subsystem Support" on page 48
- "New Blocks" on page 48
- "Enhanced Blocks" on page 50
- "New and Enhanced Demos" on page 51
- "New Options for Delay Line Block" on page 51

Additional Fixed-Point Support

For this release, significant support for fixed-point development has been added to 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

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

This section gives a brief description of each of the new blocks.

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.

Version 5.0 (R13) DSP Blockset

This table summarizes what's new in Version 5.0 (R13):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Fixed bugs	No

New features and changes introduced in this version are

- "Full Single-Precision Support" on page 53
- "Full Support of Embedded Real-Time (ERT) C Code Generation" on page 53
- "Smaller, Faster Generated C Code That Requires Less RAM" on page 53
- "Full Boolean Data Type Support" on page 53
- "Expanded Fixed-Point Data Type Support" on page 54
- "New Blocks" on page 55
- "Enhanced Blocks" on page 58
- "New and Enhanced Demos" on page 62
- "Audio Blocks Relocated to New Block Library" on page 63
- "New Options for Event Detection" on page 64
- "New Default Setting Enables Boolean Data Type Support" on page 66
- "Replaced Filtering Blocks" on page 67
- "Wavelet Analysis and Wavelet Synthesis Blocks Replaced" on page 67
- "Cumulative Sum Block Behaves Differently" on page 68
- "Contiguous Copy Block Obsolete" on page 68

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 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 Real-Time Workshop Embedded Coder).

Smaller, Faster Generated C Code That Requires Less RAM

Real-Time Workshop 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 66. 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

Expanded Fixed-Point Data Type Support

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 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 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)	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	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	

New Blocks

The key new block for Release 13 is the Digital Filter block. This section gives a brief description of each of the new blocks.

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 Filter Design Toolbox and the `filter` function in 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 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

This section gives a brief description of each of the block enhancements.

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 to New Block Library" on page 63.

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 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.

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 Filter Design Toolbox.
- **Better fixed-point filter implementation** — The block now implements a much better fixed-point filter when you install 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 Fixed-Point Blockset.
Denoising	`Wavelets`	
Wavelet Transmultiplexer (WTM)	`Wavelets`	
Multi-Level PR Filter Bank	`Wavelets`	Formerly a one-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 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

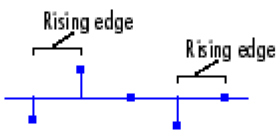
New Options for Event Detection

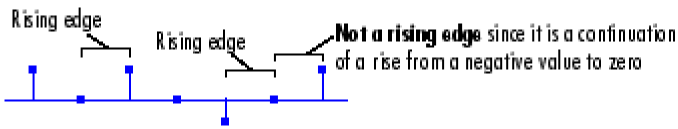
The following blocks perform an operation when an event-detecting input port (such as a clock or reset port) detects an event:

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

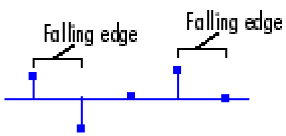
You can now set the event-detecting ports of the above blocks to respond to one of the following events:

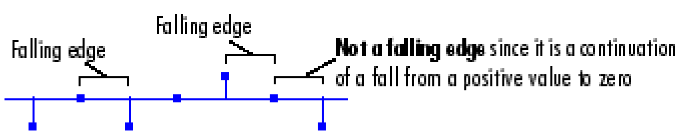
- `Non-zero sample` — When the input sample is nonzero
- `Rising edge` — When the input does one of the following:
  - Rises from a negative value to a positive value or zero
  - Rises from zero to a positive value, where the rise is not a continuation of a rise from a negative value to zero (see the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

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

Compatibility Considerations

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 53 — 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

Compatibility Considerations

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.

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.

Compatibility Considerations

This table shows what blocks to use in place of the Wavelet Analysis and Wavelet Synthesis blocks:

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.

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.

Compatibility Considerations

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.

Compatibility Considerations

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.

Compatibility Summary for Signal Processing Blockset

This table summarizes new features and changes that might cause incompatibilities when you upgrade from an earlier version, or when you use files on multiple versions. Details are provided in the description of the new feature or change.

Version (Release)	New Features and Changes with Version Compatibility Impact
Latest Version V6.6 (R2007b)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "New To Audio Device and From Audio Device Blocks" on page 4 • "Tunability Status Changed for Some Block Parameters" on page 7 • "Levinson-Durbin Block Now Treats Frame-Based Row Vectors Differently" on page 9

Version (Release)	New Features and Changes with Version Compatibility Impact
V6.5 (R2007a)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "R11.1 Blocks Have Been Removed in R2007a — Run Helper Script Before Upgrading" on page 11 • "Zero Pad Block Removed" on page 14 • "Pad Block Can Truncate Either End of an Input Signal" on page 14
V6.4 (R2006b)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "R11.1 Blocks Will Be Removed in R2007a" on page 15 • "Diagnostic Output Port Added to Report a Failure to Converge" on page 17 • "Blocks Removed from Product" on page 18
V6.3 (R2006a)	None
V6.2 (R14SP3)	None
V6.1 (R14SP2)	See the **Compatibility Considerations** subheading for this new feature or change: • "Obsolete Blocks" on page 25

Version (Release)	New Features and Changes with Version Compatibility Impact
V6.0.1 (R14SP1)	None
V6.0 (R14)	See the **Compatibility Considerations** subheading for this new feature or change: • "Obsolete Product Versions" on page 45
V5.1 (R13SP1)	None
V5.0 (R13)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "New Default Setting Enables Boolean Data Type Support" on page 66 • "Replaced Filtering Blocks" on page 67 • "Wavelet Analysis and Wavelet Synthesis Blocks Replaced" on page 67 • "Cumulative Sum Block Behaves Differently" on page 68 • "Contiguous Copy Block Obsolete" on page 68