

Communications System Toolbox™ Release Notes

How to Contact MathWorks



www.mathworks.com
comp.soft-sys.matlab
www.mathworks.com/contact_TS.html

Web
Newsgroup
Technical Support



suggest@mathworks.com
bugs@mathworks.com
doc@mathworks.com
service@mathworks.com
info@mathworks.com

Product enhancement suggestions
Bug reports
Documentation error reports
Order status, license renewals, passcodes
Sales, pricing, and general information



508-647-7000 (Phone)



508-647-7001 (Fax)



The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

Communications System Toolbox™ Release Notes

© COPYRIGHT 2011 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Summary by Version	1
Version 5.0 (R2011a) Communications System Toolbox	3
Compatibility Summary for Communications System Toolbox	18

Summary by Version

This table provides quick access to what's new in each version. For clarification, see “Using Release Notes” on page 1.

Version (Release)	New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems
New Product V 5.0 (R2011a)	Yes Details	Yes Summary	Bug Reports Includes fixes

Using Release Notes

Use release notes when upgrading to a newer version to learn about:

- New features
- Changes
- Potential impact on your existing files and practices

Review the release notes for other MathWorks® products required for this product (for example, MATLAB® or Simulink®). Determine if enhancements, bugs, or compatibility considerations in other products impact you.

If you are upgrading from a software version other than the most recent one, review the current release notes and all interim versions. For example, when you upgrade from V1.0 to V1.2, review the release notes for V1.1 and V1.2.

What Is in the Release Notes

New Features and Changes

- New functionality
- Changes to existing functionality

Version Compatibility Considerations

When a new feature or change introduces a reported incompatibility between versions, the **Compatibility Considerations** subsection explains the impact.

Compatibility issues reported after the product release appear under Bug Reports at the MathWorks Web site. Bug fixes can sometimes result in incompatibilities, so review the fixed bugs in Bug Reports for any compatibility impact.

Fixed Bugs and Known Problems

MathWorks offers a user-searchable Bug Reports database so you can view Bug Reports. The development team updates this database at release time and as more information becomes available. Bug Reports include provisions for any known workarounds or file replacements. Information is available for bugs existing in or fixed in Release 14SP2 or later. Information is not available for all bugs in earlier releases.

Access Bug Reports using your MathWorks Account.

Documentation on the MathWorks Web Site

Related documentation is available on mathworks.com for the latest release and for previous releases:

- Latest product documentation
- Archived documentation

Version 5.0 (R2011a) Communications System Toolbox

This table summarizes what's new in V 5.0 (R2011a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems
Yes Details below	Yes—Details labeled as Compatibility Considerations , below. See also Summary.	Bug Reports Includes fixes

New features and changes introduced in this version are:

- “Product Restructuring” on page 4
- “LDPC Encoder and Decoder System Objects” on page 4
- “LDPC GPU Decoder System Object” on page 4
- “Variable-Size Support” on page 4
- “Algorithm Improvements for CRC Blocks” on page 6
- “MATLAB Compiler Support for System Objects” on page 6
- “Heterogeneous Arrays of System Objects” on page 6
- “Internal rule’ System Object Property Values Changed to ‘Full precision’” on page 6
- “System Object Code Generation Support” on page 7
- “LDPC Decoder Block Warnings” on page 7
- “Phase/Frequency Offset Block and System Object Change” on page 7
- “Derepeat Block Changes” on page 7
- “Version 2 Blocks Removed” on page 8
- “System Objects Input and Property Warnings Changed to Errors” on page 8
- “Frame-Based Processing” on page 9
- “New Demos” on page 16

Product Restructuring

The Communications System Toolbox™ product replaces two pre-existing products: Communications Blockset and Communications Toolbox. You can access archived documentation for both products on the MathWorks Web site.

LDPC Encoder and Decoder System Objects

This release adds new `comm.LDPCEncoder` and `comm.LDPCDecoder` System objects. These new System objects provide simulation of low-density, parity-check codes.

LDPC GPU Decoder System Object

This release adds a new `comm.gpu.LDPCDecoder` System object, which uses a graphics processing unit (GPU) to decode low-density, parity-check codes. This new System object procures simulation results more quickly than a CPU.

Variable-Size Support

The following blocks now support variable-size input signals:

- M-PSK Modulator Baseband
- QPSK Modulator Baseband
- BPSK Modulator Baseband
- M-PAM Modulator Baseband
- Rectangular QAM Modulator Baseband
- General QAM Modulator Baseband
- M-PSK Demodulator Baseband
- QPSK Demodulator Baseband
- BPSK Demodulator Baseband
- M-PAM Demodulator Baseband
- Rectangular QAM Demodulator Baseband
- General QAM Demodulator Baseband
- Bit to Integer Converter

- Integer to Bit Converter
- Convolutional Encoder
- Viterbi Decoder

The following source blocks can now output variable-size signals:

- Gold Sequence Generator
- Kasami Sequence Generator
- PN Sequence Generator

The following System objects now support variable-size input signals:

- `comm.PSKModulator`
- `comm.QPSKModulator`
- `comm.BPSKModulator`
- `comm.PAMModulator`
- `comm.RectangularQAMModulator`
- `comm.GeneralQAMModulator`
- `comm.PSKDemodulator`
- `comm.QPSKDemodulator`
- `comm.BPSKDemodulator`
- `comm.PAMDemodulator`
- `comm.RectangularQAMDemodulator`
- `comm.GeneralQAMDemodulator`
- `comm.IntegerToBit`
- `comm.BitToInteger`

The following System objects now output variable-size signals:

- `comm.GoldSequence`
- `comm.KasamiSequence`

- comm.PNSequence

Algorithm Improvements for CRC Blocks

This release introduces a new encoding algorithm for all blocks in the CRC sublibrary residing in the Error Detection and Correction library. In this new implementation, the block processes multiple input bits in one step, resulting in faster processing times. The previous implementation always processed one input bit at each step.

MATLAB Compiler Support for System Objects

The Communications System Toolbox supports the MATLAB® Compiler™ for most System objects. With this capability, you can use the MATLAB Compiler to take MATLAB files, which can include System objects, as input and generate standalone applications.

The following System objects are not supported by the MATLAB Compiler software:

Heterogeneous Arrays of System Objects

You can now create arrays that contain System objects of different classes. This capability allows you to run a subset of methods on multiple System objects simultaneously. You can run these methods on System object arrays: `clone`, `getNumInputs`, `getNumOutputs`, `isLocked`, `release`, and `reset`. Note that the `step` method is not supported for System object arrays.

'Internal rule' System Object Property Values Changed to 'Full precision'

To clarify the value of many `DataType` properties, the 'Internal rule' option has been changed to 'Full precision'.

Compatibility Consideration

The objects allow you to enter either 'Internal rule' or 'Full precision'. If you enter 'Internal rule', that option is stored as 'Full precision'.

System Object Code Generation Support

The following System objects support code generation:

- `comm.PSKTCMMModulator`
- `comm.RectangularQAMTCMMModulator`
- `comm.GeneralQAMTCMMModulator`
- `comm.EarlyLateGateTimingSynchronizer`
- `comm.GardnerTimingSynchronizer`
- `comm.GMSKTimingSynchronizer`
- `comm.MSKTimingSynchronizer`
- `comm.MuellerMullerTimingSynchronizer`
- `comm.KasamiSequence`

LDPC Decoder Block Warnings

Communications System Toolbox software uses a new implementation of the LDPC Decoder block. If you open a previously existing model that contains the LDPC block, the model generates a warning at the MATLAB command line. Simply resave the model to prevent any subsequent warnings.

Phase/Frequency Offset Block and System Object Change

In previous releases, when the frequency offset input signal to the Phase/Frequency Offset block or `comm.PhaseFrequencyOffset` System object was constant, or time-invariant, the block and System object generated the correct output. However, the block and System object produced incorrect results for a time-varying frequency offset input signal. The new implementation generates the correct output for a time-varying frequency offset input signal.

Derepeat Block Changes

The Derepeat block now contains the **Input processing** and **Rate options** parameters. See Frame-Based Processing for more information.

Version 2 Blocks Removed

All the block libraries associated with Communications Blockset version 2 Release 12 and version 2.5 Release 13 have been removed from this product. The removal includes the following libraries:

- `commanabbnd2`
- `commanapbnd2`
- `commbkcod`
- `commchan2`
- `commcnvcod`
- `commdigbbnd2`
- `commdigpbnd2`
- `commsink`
- `commsource`
- `commsrccod`
- `commsync`
- `commutil`

Compatibility Considerations

Communications System Toolbox software does not support any of the blocks from Release 12 and Release 13. The Communications System Toolbox block libraries provide some of the same functionality in the form of upgraded blocks.

System Objects Input and Property Warnings Changed to Errors

When a System object is locked (for example, after the `step` method has been called), the following situations now produce an error. This change prevents the loss of state information.

- Changing the input data type
- Changing the number of input dimensions

- Changing the input complexity from real to complex
- Changing the data type, dimension, or complexity of tunable property
- Changing the value of a nontunable property

Compatibility Consideration

Previously, the object issued a warning for these situations. The object then unlocked, reset its state information, relocked, and continued processing. To update existing code so that it does not produce an error, use the `release` method before changing any of the items listed above.

Frame-Based Processing

In signal processing applications, you often need to process sequential samples of data at once as a group, rather than one sample at a time. Communications System Toolbox documentation refers to the former as frame-based processing and the latter as sample-based processing. A frame is a collection of samples of data, sequential in time.

Historically, Simulink-family products that can perform frame-based processing propagate frame-based signals throughout a model. The frame status is an attribute of the signals in a model, just as data type and dimensions are attributes of a signal. The Simulink engine propagates the frame attribute of a signal by means of a frame bit, which can either be on or off. When the frame bit is on, Simulink interprets the signal as frame based and displays it as a double line, rather than the single line sample-based signal.

General Product-Wide Changes

Beginning in R2010b, MathWorks started to significantly change the handling of frame-based processing. In the future, frame status will no longer be a signal attribute. Instead, individual blocks will control whether they treat inputs as frames of data or as samples of data. To learn how a particular block handles its input, you can refer to the block reference page.

To transition to the new paradigm of frame-based processing, many blocks have received new parameters. The following sections provide more detailed information about the specific Communications System Toolbox

software changes that are helping to enable the transition to the new way of frame-based processing:

- “Blocks with a New Input Processing Parameter” on page 11
- “Multirate Processing Parameter Changes” on page 13
- “Sample-Based Row Vector Processing Changes” on page 15

Compatibility Considerations. During this transition to the new way of handling frame-based processing, both the old way (frame status as an attribute of a signal) and the new way (each block controls whether to treat inputs as samples or as frames) will coexist for a few releases. For now, the frame bit will still flow throughout a model, and you will still see double signal lines in your existing models that perform frame-based processing.

- **Backward Compatibility** — By default, when you load an existing model in R2010b any new parameters related to the frame-based processing change will be set to their backward-compatible option. For example, if any blocks in your existing models received the **Input processing** parameter, the parameter will be set to **Inherited** (this choice will be removed - see release notes) when you load your model. This setting enables your existing models to continue working as expected until you upgrade them. Because the inherited option will be removed in a future release, you should upgrade your existing models as soon as possible.
- **slupdate Function** — To upgrade your existing models to the new way of handling frame-based processing, you can use the `slupdate` function. Your model must be compilable in order to run the `slupdate` function. The function detects all blocks in your model that are in need of updating, and asks you whether you would like to upgrade each block. If you select yes, the `slupdate` function updates your blocks accordingly.
- **Timely Update to Avoid Unexpected Results** — It is important to update your existing models as soon as possible because the frame bit will be removed in a future release. At that time, any blocks that have not yet been upgraded to work with the new paradigm of frame-based processing will automatically transition to perform their library default behavior. The library default behavior of the block might not produce the results you expected, thus causing undesired results in your models. Once the frame bit is removed, you will no longer be able to upgrade your models using the

slupdate function. Therefore, you should upgrade your existing modes using slupdate as soon as possible.

For more detailed information about the specific compatibility considerations related to the R2010b frame-based processing changes, see the following Compatibility Considerations sections.

Blocks with a New Input Processing Parameter

Some Communications System Toolbox blocks are able to process both sample- and frame-based signals. After the transition to the new way of handling frame-based processing, signals will no longer carry information about their frame status. Blocks that can perform both sample- and frame-based processing will require a new parameter that allows you to specify the appropriate processing behavior. To prepare for this change, many blocks received a new **Input processing** parameter. You can select **Columns as channels (frame based)** or **Elements as channels (sample based)**, depending upon the type of processing you want. The third choice, **Inherited** (this choice will be removed - see release notes), is a temporary selection. This additional option will help you to migrate your existing models from the old paradigm of frame-based processing to the new paradigm. Refer to the “Compatibility Summary for Communications System Toolbox” on page 18 section for more information about migrating your existing models to the new paradigm of frame-based processing.

For a list of blocks that received a new **Input processing** parameter, expand the following list.

Blocks with New Input Processing Parameter

- Derepeat
- Gaussian Filter
- Windowed Integrator
- AWGN Channel (with only two options)

Compatibility Considerations. When you load an existing model R2010b, any block with the new **Input processing** parameter will show a setting of Inherited (this choice will be removed - see release notes). This setting enables your existing models to continue to work as expected until you upgrade them. Although your old models will still work when you open and run them in R2010b, you should upgrade them as soon as possible.

You can upgrade your existing models, using the `slupdate` function. The function detects all blocks that have Inherited (this choice will be removed - see release notes) selected for the **Input processing** parameter, and asks you whether you would like to upgrade each block. If you select yes for the Gaussian Filter or Windowed Integrator, the function detects the status of the frame bit on the input port of the block. If the frame bit is 1 (frames), the function sets the **Input processing** parameter to Columns as channels (frame based). If the bit is 0 (samples), the function sets the parameter to Elements as channels (sample based).

In a future release, the frame bit and the Inherited (this choice will be removed - see release notes) option will be removed. At that time, the **Input processing** parameter in models that have not been upgraded will automatically be set to either Columns as channels (frame based) or Elements as channels (sample based), depending on the library default setting for each block. If the library default setting does not match the parameter setting in your model, your model will produce unexpected results. Additionally, after the frame bit is removed, you will no longer be able to upgrade your models using the `slupdate` function. Therefore, you should upgrade your existing modes using `slupdate` as soon as possible.

AWGN Channel Block Changes

The AWGN Channel block uses the new method of “Frame-Based Processing” on page 9. In previous releases, the frame status of the input signal determined how the AWGN Channel block processed the signal. In R2010b, the default behavior of the AWGN Channel block is to always perform frame-based processing.

Unless you specify otherwise, the block now treats each column of the input signal as an individual channel, regardless of its frame status. To enable the behavior change in the AWGN Channel block while still allowing for backward compatibility, an **Input processing** parameter has been added.

This parameter will be removed in a future release, at which point the block will always perform frame-based processing.

Compatibility Considerations. The **Input processing** parameter will be removed in a future release. At that point in time, the AWGN Channel block will always perform frame-based processing.

You can use the `slupdate` function to upgrade your existing models that contain an AWGN Channel block. The function detects all AWGN Channel blocks in your model and, if you allow it to, performs the following actions:

- If the input to the block is an M -by-1 or unoriented sample-based signal, the `slupdate` function performs three actions. First, a Transpose block is placed in front of the AWGN Channel block in your model. This block transposes the M -by-1 or unoriented sample-based input into a 1-by- M row vector. By converting the input to a row vector, the block continues to produce the same results as in previous releases. The `slupdate` function also sets the **Input processing** parameter to `Columns as channels (frame based)`. This setting ensures that your model will continue to produce the same results when the **Input processing** parameter is removed in a future release. The `slupdate` function also adds a Transpose block after the AWGN channel block in your model for an M -by-1 sample-based input and a Reshape block for unoriented inputs. By converting the row vector output of the AWGN channel to the input dimension, the model continues to behave as in prior releases.
- If the input to the block is *not* an M -by-1 or unoriented sample-based signal, the `slupdate` function sets the **Input processing** parameter to `Columns as channels (frame based)`. This setting does not affect the behavior of your current model. However, the change does ensure that your model will continue to produce the same results when the **Input processing** parameter is removed in a future release.

Multirate Processing Parameter Changes

In R2010a and earlier releases, many Communications System Toolbox blocks that supported multirate processing had a **Framing** parameter. This parameter allowed you to specify whether the block should `Maintain input frame size` or `Maintain input frame rate` when processing the input signal. Beginning in R2010b, a new **Rate options** parameter replaced the **Framing** parameter. The **Rate options** parameter allows you to specify

whether the block should **Enforce single-rate processing** or **Allow multirate processing**.

Some blocks that supported multirate processing in R2010a and earlier releases did not have a **Framing** parameter. These blocks used the frame status of the input signal to determine whether they performed single-rate or multirate processing. Because of the upcoming frame-based processing changes, signals will no longer carry their frame status. Thus, multirate blocks can no longer rely on the frame status of the input signal to determine whether they perform single-rate or multirate processing. You must now specify a value for the **Rate options** parameter on the block dialog box.

To see a full list of blocks that have a new **Rate options** parameter, expand the following section.

Multirate Blocks with a New Rate Options Parameter

- Raised Cosine Receive Filter
- Raised Cosine Transmit Filter
- Ideal Rectangular Pulse Filter
- OQPSK Modulator Baseband
- OQPSK Demodulator Baseband
- CPM Modulator Baseband
- CPM Demodulator Baseband
- MSK Modulator Baseband
- MSK Demodulator Baseband
- GMSK Modulator Baseband
- GMSK Demodulator Baseband
- CPFSK Modulator Baseband
- CPFSK Demodulator Baseband
- M-FSK Demodulator Baseband
- M-FSK Modulator Baseband

- Derepeat

Sample-Based Row Vector Processing Changes

The following blocks do not process sample-based row vectors:

- APP Decoder
- Convolutional Encoder
- Viterbi Decoder
- Algebraic Deinterleaver
- Algebraic Interleaver
- General Block Deinterleaver
- General Block Interleaver
- Matrix Deinterleaver
- Matrix Helical Scan Deinterleaver
- Matrix Helical Scan Interleaver
- Matrix Interleaver
- Random Deinterleaver
- Random Interleaver
- M-PAM Modulator Baseband
- Rectangular QAM Modulator Baseband
- DQPSK Modulator Baseband
- M-DPSK Modulator Baseband
- M-PSK Modulator Baseband
- OQPSK Modulator Baseband
- QPSK Modulator Baseband
- M-FSK Modulator Baseband
- CPFSK Modulator Baseband
- CPM Modulator Baseband

- Insert Zero
- Puncture
- Bit to Integer Converter
- Integer to Bit Converter

Compatibility Considerations. Using existing models that contain these blocks to process sample-based row vectors generates an error message.

CMA Equalizer Changes

The CMA Equalizer block now handles input signals like the other equalizer blocks in the Communications Blockset library. Therefore, the block no longer accepts scalar input signals in symbol-spaced mode.

Differential Encoder Changes

The Differential Encoder block supports scalar-valued and column vector input signals. It does not support frame-based or sample-based row vectors.

Find Delay and Align Signal Block Changes

The **Correlation window length** parameter specifies the number of samples the block uses to calculate the cross-correlation of two signals. You must specify a window lengths of at least 2 for the cross-correlation calculations. If you set the **Correlation window length** parameter to 1, the block generates an error message. The following blocks contain the **Correlation window length** parameter:

- Find Delay
- Align Signals

New Demos

This release contains the following new demos:

- Parallel Concatenated Convolutional Coding: Turbo Codes
- Go-Back-N ARQ with PHY Layer
- Adaptive MIMO System with OSTBC

- CORDIC-Based QPSK Carrier Synchronization
- DVB-S.2 Link, Including LDPC Coding
- DVB-S.2 System Simulation Using a GPU-Based LDPC Decoder System Object

Compatibility Summary for Communications System Toolbox

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
<p>Latest Version V5.0 (R2011a)</p>	<p>See the Compatibility Considerations subheading for each of these new features or changes:</p> <ul style="list-style-type: none"> • “Internal rule’ System Object Property Values Changed to ’Full precision” on page 6 • “General Product-Wide Changes” on page 9 • “LDPC Decoder Block Warnings” on page 7 • “ Phase/Frequency Offset Block and System Object Change” on page 7 • “Version 2 Blocks Removed” on page 8 • “System Objects Input and Property Warnings Changed to Errors” on page 8
<p>V5.0 (R2010b) Communications Blockset</p>	<p>Previous Communications Blockset Compatibility Considerations</p>
<p>V4.6 (R2010b) Communications Toolbox</p>	<p>Previous Communications Toolbox Compatibility Considerations</p>