Communications Blockset Release Notes

The “Communications Blockset 3.0 Release Notes” describe the changes introduced in the latest version of the Communications Blockset. The following topics are discussed in these Release Notes.

The Communications Blockset Release Notes also provide information about recent versions of the product, in case you are upgrading from a version that was released prior to Release 13 with Service Pack 1.

Printing the Release Notes

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

Contents

1 Communications Blockset 3.0 Release Notes

2 Communications Blockset 2.5 Release Notes

3 Communications Blockset 2.0.1 Release Notes

4 Communications Blockset 2.0 Release Notes

1

Communications Blockset 3.0 Release Notes

New Features

This section summarizes the new features and enhancements introduced in the Communications Blockset 3.0:

If you are upgrading from a release earlier than Release 13, then you should also see the Communications Blockset 2.5 Release Notes, starting with "New Features" on page 2-2.

Timing Phase Recovery

The blocks in the table below perform timing phase recovery, determining the best instant within a symbol period to sample a signal at the receiver. Sampling at the best instant improves the receiver's performance on a noisy signal. All blocks listed in the table are in the Timing Recovery sublibrary of the Synchronization library.

Block	Purpose
Early-Late Gate Timing Recovery	Recover the symbol timing phase using the early-late gate method
Gardner Timing Recovery	Recover the symbol timing phase using Gardner's method
MSK-Type Signal Timing Recovery	Recover the symbol timing phase using a fourth-order nonlinearity method

Block (Continued)	Purpose (Continued)
Mueller-Muller Timing Recovery	Recover the symbol timing phase using the Mueller-Muller method
Squaring Timing Recovery	Recover the symbol timing phase using a squaring method

For more information and an example, see "Timing Phase Recovery" in the Using the Communications Blockset documentation. For demos, type `gardner_vfracdelay` or `msk_sync` in the MATLAB Command Window.

Carrier Phase Recovery

The blocks in the table below perform carrier phase recovery. They are in the Carrier Recovery sublibrary of the Synchronization library.

Block	Purpose
M-PSK Phase Recovery	Recover the carrier phase using the M-Power method
CPM Phase Recovery	Recover the carrier phase using the 2P-Power method

For more information and an example, see "Carrier Phase Recovery" in the Using the Communications Blockset documentation. For a demo, type `msk_sync` in the MATLAB Command Window.

Equalizers

The blocks in the table below enable you to equalize a signal using a linear equalizer, a decision feedback equalizer, or a maximum-likelihood sequence

estimation equalizer based on the Viterbi algorithm. All blocks listed in the table are in the Equalizers library.

Block	Purpose
CMA Equalizer	Equalize using the constant modulus algorithm
LMS Decision Feedback Equalizer	Equalize using a decision feedback equalizer that updates weights with the LMS algorithm
LMS Linear Equalizer	Equalize using a linear equalizer that updates weights with the LMS algorithm
MLSE Equalizer	Equalize using the Viterbi algorithm
Normalized LMS Decision Feedback Equalizer	Equalize using a decision feedback equalizer that updates weights with the normalized LMS algorithm
Normalized LMS Linear Equalizer	Equalize using a linear equalizer that updates weights with the normalized LMS algorithm
RLS Decision Feedback Equalizer	Equalize using a decision feedback equalizer that updates weights with the RLS algorithm
RLS Linear Equalizer	Equalize using a linear equalizer that updates weights with the RLS algorithm
Sign LMS Decision Feedback Equalizer	Equalize using a decision feedback equalizer that updates weights with the signed LMS algorithm
Sign LMS Linear Equalizer	Equalize using a linear equalizer that updates weights with the signed LMS algorithm

Block (Continued)	Purpose (Continued)
Variable Step LMS Decision Feedback Equalizer	Equalize using a decision feedback equalizer that updates weights with the variable step size LMS algorithm
Variable Step LMS Linear Equalizer	Equalize using a linear equalizer that updates weights with the variable step size LMS algorithm

For more information, see “Equalizers”. For an example, see the new Defense Communications: US MIL-STD-188-110B demo (`milstd_188110Bmodel`).

Filtering and Pulse Shaping

The blocks in the table below perform filtering and pulse shaping. All blocks listed in the table are in the Comm Filters library.

Block	Purpose
Gaussian Filter	Filter the input signal, possibly downsampling, using a Gaussian FIR filter
Ideal Rectangular Pulse Filter	Shape the input signal using ideal rectangular pulses
Raised Cosine Receive Filter	Filter the input signal, possibly downsampling, using a raised cosine FIR filter
Raised Cosine Transmit Filter	Upsample and filter the input signal using a raised cosine FIR filter

Trellis-Coded Modulation

The blocks in the table below perform trellis-coded modulation. All blocks listed in the table are in the TCM sublibrary of Digital Baseband Modulation, in the Modulation library.

Block	Purpose
General TCM Decoder	Decode trellis-coded modulation data, mapped using an arbitrary constellation
General TCM Encoder	Convolutionally encode binary data and map using an arbitrary constellation
M-PSK TCM Decoder	Decode trellis-coded modulation data, modulated using the PSK method
M-PSK TCM Encoder	Convolutionally encode binary data and modulate using the PSK method
Rectangular QAM TCM Decoder	Decode trellis-coded modulation data, modulated using the QAM method
Rectangular QAM TCM Encoder	Convolutionally encode binary data and modulate using the QAM method

Utility Blocks for Working with Delays

The blocks in the table below help you compute or manipulate the delay through one or more blocks in your model. This is especially useful when you are comparing two signals to compute error rates, or when you need to align boundaries of codewords or other groupings with Simulink frame boundaries. All blocks listed in the table are in the Utility Blocks library.

Block	Purpose
Align Signals	Align two signals by finding the delay between them
Find Delay	Find the delay between two signals

The reference pages for these blocks include examples of how to use them in a variety of situations.

Enhanced Source Coding Blocks

The new Quantizing Encoder and Quantizing Decoder blocks replace the older Sampled Quantizer Encode and Quantizer Decode blocks, which are now obsolete. The new blocks perform scalar quantization encoding and decoding operations, respectively. The new blocks can process frame-based column vectors in addition to other kinds of vectors. The new Quantizing Encoder block does not require you to specify the vector length or sample time as parameters in the dialog box.

The older encoder produced a third output signal that represented the mean square distortion, while the new Quantizing Encoder block does not. The older encoder produced a discrete-time output signal even if its input was continuous-time, whereas the new Quantizing Encoder block preserves sample times.

AWGN Channel Enhancement for RSim Target

Selected parameters of the AWGN Channel block are now compatible with the Real-Time Workshop rapid simulation (RSim) target. This means that if you use Real-Time Workshop to build an RSim executable, then you can tune selected parameters without recompiling the model. This is useful for Monte Carlo simulations in which you run the simulation multiple times (perhaps on multiple computers) with different amounts of noise. The table below indicates, for different modes of the block, which parameters are tunable.

Mode	Tunable Parameters
`Eb/No`	**Eb/No**, **Input signal power**
`Es/No`	**Es/No**, **Input signal power**
`SNR`	**SNR**, **Input signal power**
`Variance from mask`	**Variance**

For more information about the RSim target, see the Real-Time Workshop documentation set.

New Demos

New demos in Release 14 are listed in the table below. You can open the demos by finding them in the **Demos** pane of the MATLAB Help browser or by entering the corresponding model names in the MATLAB Command Window.

Title	Model Name
Convolutional Encoder with Uncoded Bits and Feedback	`conv_encoderdemo`
Soft-Decision GMSK Demodulator	`gmsk_softdecision`
Adjacent and Co-Channel Interference	`adjcochanint`
Adaptive Equalization Using Embedded MATLAB	`equalizer_eml`
Gardner Timing Phase Recovery	`gardner_vfracdelay`
MSK Signal Recovery	`msk_sync`
IEEE 802.11a WLAN Physical Layer	`wlan80211a`
Physical Layer Model of the cdma2000® Standard	`cdma2000_phlayer`
Defense Communications: US MIL-STD-188-110B	`milstd_188110Bmodel`

Demonstration models have also been reorganized into categories to make it easier for you to find relevant demos. You can view the categories using the **Demos** pane of the MATLAB Help browser.

Major Bug Fixes

The Communications Blockset 3.0 includes several bug fixes made since Version 2.0.1. This section describes the particularly important Version 3.0 bug fixes.

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

If you are upgrading from a release earlier than Release 13 with Service Pack 1, then you should see the list (HTML only) of important Version 2.5 bug fixes.

Upgrading from an Earlier Release

The issues involved in upgrading from the Communications Blockset 2.5 to Version 3.0 are described below.

If you are upgrading from a version earlier than 2.5, you should see "Upgrading from an Earlier Release" on page 2-8 in the Communications Blockset 2.5 Release Notes.

Changes in BCH Encoder and BCH Decoder

In Release 14, the BCH Decoder block has been changed such that the second output port is optional and the error-correction capability is no longer a parameter. Also, this block and the BCH Encoder block no longer accept sample-based inputs. If you built models with earlier versions of these two blocks, then you should

- Resave the models using Release 14, to avoid producing Simulink warnings.
- Revise the models so that the inputs to the BCH blocks are frame-based column vectors rather than sample-based vectors. To change the shape or frame status of a signal, you can use the Reshape block in Simulink, or the Frame Status Conversion block in the Signal Processing Blockset. Because the outputs from the BCH blocks are now frame-based column vectors, you might need to revise other parts of your model as well.

Changes in Fading Channel Blocks

The Multipath Rayleigh Fading Channel and Rician Fading Channel blocks are designed to process only sample-based scalars or frame-based column vectors. In Release 13, the blocks mistakenly accepted sample-based column vectors as input. In Release 14, the blocks correctly produce an error message if the input signal is a sample-based vector or a matrix.

Changes in Integrators

The new Communications Filters library contains a new Integrate and Dump block and a new Windowed Integrator block. These blocks behave differently compared to the respective blocks of the same names in Release 13.

The new Integrate and Dump block

- Does not reduce the sum modulo a constant. The **Absolute value bound** parameter is not part of the new block.
- Does not require you to enter the sample time in the dialog box. The **Sample time** parameter is not part of the new block.
- Measures the **Integration period** parameter value in samples, not seconds.
- Can process sample-based scalars and frame-based matrices, but not sample-based vectors of length greater than 1. In a frame-based matrix, a given column is interpreted as a set of samples from a single channel.
- Can optionally discard a specified number of input samples at the beginning of the simulation. In frame-based mode, the number of samples to discard can be different for each channel (column) of the input matrix.
- Can optionally suppress the intermediate cumulative sums and output only the final sum.

The new Windowed Integrator block

- Does not require you to enter the sample time or vector size in the dialog box. The **Sample time** and **Input vector size** parameters are not part of the new block.
- Measures the integration period in samples, not seconds.
- Can process sample-based scalars and frame-based matrices, but not sample-based vectors of length greater than 1. In a frame-based matrix, a given column is interpreted as a set of samples from a single channel.
- Processes only discrete-time signals, not continuous-time signals.
- Uses cumulative sums as integrals and does not offer a choice of integration methods. The **Integration method** parameter is not part of the new block.

To learn more about the new blocks, see the Integrate and Dump and Windowed Integrator online reference entries, respectively.

Legacy Models Containing Integrator Blocks

If you built models with the older Integrate and Dump block or the older Windowed Integrator block, then the block is unchanged there. You can update the block manually by replacing it with the newer block from the Communications Filters library. You might need to change parameters or other parts of your model to make the new block fit into your model.

To find the older blocks in their default library setting, type `comminteg2` in the MATLAB Command Window.

Note The older Integrate and Dump block and the older Windowed Integrator blocks are obsolete and might be removed from a future release of the Communications Blockset.

Change in Error Rate Calculation Block

If you set **Output data** to `Workspace` in the Error Rate Calculation block, then the variable containing the output data resides in the base MATLAB workspace. In previous releases, the variable resided in the calling workspace.

This change is relevant if you invoke the simulation from a function. If you need to access the output data within the function, use `evalin`. For example, in a function, the command below accesses a variable called `ErrorVec` in the base MATLAB workspace and assigns its value to a variable by the same name in the function workspace.

```
ErrorVec = evalin('base','ErrorVec;');
```

If you invoke the simulation directly from the model window or by entering a `sim` command in the MATLAB Command Window, then the change in behavior of the Error Rate Calculation block does not affect you.

Version 1.3 Libraries Removed

The block libraries from the Communications Toolbox Version 1.3 (Release 10) are no longer installed as part of Release 14. The block libraries from the Communications Toolbox Version 1.5 (Release 11) might be removed from a future release.

Obsolete Blocks

The table below lists blocks from Release 13 that are obsolete as of Release 14. In particular, all digital passband modulation, digital passband demodulation, analog baseband modulation, and analog baseband demodulation blocks are obsolete. In place of digital passband blocks, use their digital baseband

counterparts. In place of analog baseband blocks, use their analog passband counterparts.

Note For backward compatibility, the obsolete blocks in the table below are still provided in Release 14 in the `matlabroot/commblks/commblksobsolete` directory tree. However, they might be removed in a future release and it is recommended that you avoid using these obsolete blocks in your models.

Where applicable, the second column lists blocks that provide similar functionality. In some cases, the similar block requires different parameter settings, data formats, or signal attributes compared to the original block. Therefore, you should read the documentation for the similar block before using it in your model.

Obsolete Block	Similar Block(s), if Any
Continuous-Time Eye and Scatter Diagrams	Discrete-Time Eye Diagram Scope, Discrete-Time Scatter Plot Scope, Discrete-Time Signal Trajectory Scope
CPFSK Demodulator Passband	CPFSK Demodulator Baseband
CPFSK Modulator Passband	CPFSK Modulator Baseband
CPM Demodulator Passband	CPM Demodulator Baseband
CPM Modulator Passband	CPM Modulator Baseband
Discrete Modulo Integrator	
DPCM Decoder	
DPCM Encoder	
DSB AM Demodulator Baseband	DSB AM Demodulator Passband
DSB AM Modulator Baseband	DSB AM Modulator Passband

Obsolete Block (Continued)	Similar Block(s), if Any (Continued)
DSBSC AM Demodulator Baseband	DSBSC AM Demodulator Passband
DSBSC AM Modulator Baseband	DSBSC AM Modulator Passband
Enabled Quantizer Encode	Quantizing Encoder
FM Demodulator Baseband	FM Demodulator Passband
FM Modulator Baseband	FM Modulator Passband
General QAM Demodulator Passband	General QAM Demodulator Baseband
General QAM Modulator Passband	General QAM Modulator Baseband
GMSK Demodulator Passband	GMSK Demodulator Baseband
GMSK Modulator Passband	GMSK Modulator Baseband
M-DPSK Demodulator Passband	M-DPSK Demodulator Baseband
M-DPSK Modulator Passband	M-DPSK Modulator Baseband
M-FSK Demodulator Passband	M-FSK Demodulator Baseband
M-FSK Modulator Passband	M-FSK Modulator Baseband
Modulo Integrator	
M-PAM Demodulator Passband	M-PAM Demodulator Baseband
M-PAM Modulator Passband	M-PAM Modulator Baseband
M-PSK Demodulator Passband	M-PSK Demodulator Baseband
M-PSK Modulator Passband	M-PSK Modulator Baseband
MSK Demodulator Passband	MSK Demodulator Baseband
MSK Modulator Passband	MSK Modulator Baseband
OQPSK Demodulator Passband	OQPSK Demodulator Baseband

Obsolete Block (Continued)	Similar Block(s), if Any (Continued)
OQPSK Modulator Passband	OQPSK Modulator Baseband
PM Demodulator Baseband	PM Demodulator Passband
PM Modulator Baseband	PM Modulator Passband
Quantizer Decode	Quantizing Decoder
Rectangular QAM Demodulator Passband	Rectangular QAM Demodulator Baseband
Rectangular QAM Modulator Passband	Rectangular QAM Modulator Baseband
Sampled Quantizer Encode	Quantizing Encoder
SSB AM Demodulator Baseband	SSB AM Demodulator Passband
SSB AM Modulator Baseband	SSB AM Modulator Passband
Triggered Read From File	From File (Simulink)
Triggered Write to File	To File (Simulink)

Blocks Now in Different Library Locations

The table below lists blocks that reside in different libraries in Release 14, compared to Release 13. If you used these blocks in models that you saved in Release 13, then the blocks will still work in Release 14. However, you should be aware of the changed locations in case you look for these blocks in Release 14 in the library windows or the Simulink Library Browser.

Block	Release 13 Location	Release 14 Location
Baseband PLL	Synchronization	Components sublibrary of Synchronization
Binary Error Pattern Generator	Data Sources sublibrary of Comm Sources	Noise Generators sublibrary of Comm Sources
Charge Pump PLL	Synchronization	Components sublibrary of Synchronization
Complex Phase Difference	Sequence Operations sublibrary of Basic Comm Functions	Utility Blocks
Complex Phase Shift	Sequence Operations sublibrary of Basic Comm Functions	Utility Blocks
Discrete-Time VCO	Controlled Sources sublibrary of Comm Sources	Components sublibrary of Synchronization
Integrate and Dump	Integrators sublibrary of Basic Comm Functions	Communications Filters
Linearized Baseband PLL	Synchronization	Components sublibrary of Synchronization
Phase-Locked Loop	Synchronization	Components sublibrary of Synchronization
Voltage-Controlled Oscillator	Controlled Sources sublibrary of Comm Sources	Components sublibrary of Synchronization
Windowed Integrator	Integrators sublibrary of Basic Comm Functions	Communications Filters

Utility Functions Library Renamed

The Utility Functions library is now called Utility Blocks.

Contents of Basic Comm Function Library Moved

The Basic Comm Functions library, which consisted of the Integrators sublibrary and the Sequence Operations sublibrary, is no longer in the Communications Blockset. Sequence Operations has become a top-level library. The Integrate and Dump block and the Windowed Integrator block, formerly in the Integrators sublibrary, are now in the Communications Filters library. The Discrete Modulo Integrator and Modulo Integrator blocks are now obsolete.

Changes in Block Dialog Boxes

A few blocks have renamed some of their parameters or made other dialog box changes. Legacy models might issue warnings when you first open them with Release 14. After you resave the models with Release 14, the warnings will not recur. Specific changes are listed below.

Block	Release 13 Characteristic	Changes in Release 14
BCH Decoder	**Show number of errors** check box	**Output number of corrected errors** check box
Binary-Output RS Decoder	**Output port for number of corrected errors** check box	**Output number of corrected errors** check box
Discrete-Time Eye Diagram Scope	Dialog box uses check boxes to show or hide groups of parameters	Dialog box uses tabbed panels to organize parameters
Discrete-Time Scatter Plot Scope		
Discrete-Time Signal Trajectory Scope		
Discrete-Time VCO	**Oscillation frequency** parameter	Renamed as **Quiescent frequency** parameter
Voltage-Controlled Oscillator		

Changes in commstartup Function

The `commstartup` function, which changes the default Simulink model settings to values more appropriate for the simulation of communication systems, has changed some of its settings. When you run `commstartup`, it

- Changes the default solver to a discrete solver.
- Changes the default value of a Simulink diagnostic setting so that Simulink does not issue a warning when a source block uses an inherited sample time. Some Communications Blockset blocks internally inherit sample times, which can be a useful and valid modeling technique.

Simulation Settings of Legacy Models

Your legacy models might issue warnings if they use settings other than the ones listed in "Changes in commstartup Function" above. You can suppress the warnings by changing certain settings and resaving the model.

Discrete Solver. If you have legacy models that issue a warning like

```
Warning: The model 'untitled' does not have continuous states, hence
using the solver 'VariableStepDiscrete' instead of the solver
'ode45' specified in the Configuration Parameters dialog.
```

when you start the simulation in R14, then consider changing the solver to a discrete solver and resaving the model. To change the solver, use the **Configuration Parameters** option on the model window's **Simulation** menu.

Sample Time of Source Blocks. Some Communications Blockset blocks internally inherit sample times, which can be a useful and valid modeling technique. If you have legacy models that issue a warning like

```
Warning: Source 'untitled/DSP Constant' specifies that its sample
time (-1) should be back-inherited. You should explicitly specify
the sample time of sources.
```

when you start the simulation in R14, then consider changing the diagnostic setting manually and resaving the model. To change the setting manually, choose **Configuration Parameters** option on the model window's **Simulation** menu, expand **Diagnostics** in the left pane, select **Sample Time** in the left pane, and then set **Source block specifies -1 sample time** to `none` in the right pane.

Known Software and Documentation Problems

This section includes a link to a description of known software and documentation problems in Version 3.0.

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

2

Communications Blockset 2.5 Release Notes

New Features

This section summarizes the new features and enhancements introduced in the Communications Blockset 2.5:

- RF Impairments Library
- Sequence Generators Library
- Eye Diagram, ScatterPlot, and Signal Trajectory Scopes
- CRC Library
- Enhancements to Reed-Solomon Blocks
- New Demos
- Enhancements to CPM Modulator Block

If you are upgrading from a release earlier than Release 12.1, then you should also see the Communications Blockset 2.0.1 Release Notes, starting with "New Features" on page 3-2.

RF Impairments Library

The new RF Impairments library contains blocks to simulate radio frequency (RF) impairments at the receiver. The blocks in the library are listed in the following table.

Block Name	Purpose
Free Space Path Loss	Reduce the amplitude of the input signal by the amount specified
I/Q Imbalance	Create a complex baseband model of the signal impairments caused by imbalances between in-phase and quadrature receiver components
Memoryless Nonlinearity	Apply a memoryless nonlinearity to a complex, baseband signal
Phase/Frequency Offset	Apply residual phase and frequency offsets to a complex, baseband signal

Block Name (Continued)	Purpose (Continued)
Phase Noise	Apply receiver phase noise to a complex, baseband signal
Receiver Thermal Noise	Apply receiver thermal noise to a complex, baseband signal

Sequence Generators Library

The Comm Sources library is now divided into four sublibraries for Version 2.5. Three of these sublibraries contain the blocks from the Version 2.0.1 Comm Sources library:

- Data Sources
- Noise Sources
- Controlled Sources

The fourth, the Sequence Generators sublibrary, contains the PN Sequence Generator block and five new blocks for Version 2.5. You can use the blocks in the Sequence Generators sublibrary to generate sequences for spreading or synchronization in a communication system. The following table lists the blocks in the Sequence Generators sublibrary.

Block Name	Purpose
Barker Code Generator	Generate a Barker Code
Gold Sequence Generator	Generate a Gold sequence from a set of sequences
Kasami Sequence Generator	Generate a Kasami sequence from the set of Kasami sequences
Hadamard Code Generator	Generate a Hadamard code from an orthogonal set of codes
OVSF Code Generator	Generate an orthogonal variable spreading factor (OVSF) code from a set of orthogonal codes

Block Name (Continued)	Purpose (Continued)
PN Sequence Generator	Generate a pseudonoise sequence
Walsh Code Generator	Generate a Walsh code from an orthogonal set of codes

Eye Diagram, ScatterPlot, and Signal Trajectory Scopes

The Version 2.0.1 Discrete-Time Eye and Scatter Diagram block, in the Comm Sinks library, has been replaced by three new blocks for Version 2.5, as described in the following table.

Block Name	Purpose
Discrete-Time Eye Diagram Scope	Display multiple traces of a modulated signal
Discrete-Time Scatter Plot Scope	Display a modulated signal in its signal space by plotting its in-phase component against its quadrature component
Discrete-Time Signal Trajectory Scope	Display a modulated signal in its signal space by plotting its in-phase component versus its quadrature component

These blocks greatly enhance the features of the Discrete-Time Eye and Scatter Diagram.

CRC Library

The Channel Coding library has been renamed the Error Correction and Detection library, and a new sublibrary, CRC, has been added to the Error Detection and Correction library. The CRC library contains new blocks for appending cylic redundancy check (CRC) bits to data and for detecting errors in transmission.

The following table lists the blocks in the CRC library.

Block Name	Purpose
CRC-N Generator	Generate CRC bits according to the selected CRC method and append them to input data
CRC-N Syndrome Detector	Detect errors in the input data according to the specified CRC method
General CRC Generator	Generate CRC bits according to the generator polynomial and append them to input data
General CRC Syndrome Detector	Detect errors in the input data according to the generator polynomial

Enhancements to Reed-Solomon Blocks

The following four blocks, in the Block sublibrary of the Error Detection and Correction Library, have new features:

- Binary-Input RS Endoder
- Binary-Input RS Decoder
- Integer-Input RS Encoder
- Integer-Input RS Decoder

You can now specify the primitive polynomial and generator polynomial, which are used to generate the codes. This enables you to use a much wider range of Reed-Solomon Codes. There is also a new option to output the number of corrected errors from the Binary-Input RS Decoder and Integer-Input RS Decoder blocks.

New Demos

The Communications Blockset contains eleven new demos for Version 2.5. These include a large-scale demo model of a commercial application of a third generation (3G) wireless system using wide-band code division multiple access (WCDMA). The demo presents an end-to-end transmission between a base

station and a mobile station, as specified by the Third Generation Partnership Project (3GPP).

The new demos are as follows:

- WCDMA End-to-End Physical Layer Demo
- WCDMA Coding and Multiplexing Demo
- WCDMA Spreading and Modulation Demo
- RF Satellite Link Demo
- HiperLAN/2 Demo
- Bluetooth Voice Transmission Demo
- Adaptive Equalization Demo
- CPM Phase Tree Demo
- GMSK vs. MSK Demo
- Filtered QPSK vs. MSK Demo
- Raleigh Fading Channel Demo

Enhancements to CPM Modulator Block

The CPM modulator block now enables you to specify both the entire pulse length and the pulse main lobe length when simulating an LSRC frequency pulse length. This feature enables you to simulate a modulation such as 3SRC6.

Major Bug Fixes

The Communications Blockset 2.5 includes several bug fixes made since Version 2.0.1. This section describes the particularly important Version 2.5 bug fixes.

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

If you are upgrading from a release earlier than Release 12.1, then you should see “Major Bug Fixes” on page 3-4.

Upgrading from an Earlier Release

The issues involved in upgrading from the Communications Blockset 2.0.1 to Version 2.5 are described below.

If you are upgrading from a version earlier than 2.0.1, then you should see "Upgrading from an Earlier Release" on page 3-5 in the Communications Blockset 2.0.1 Release Notes.

Old Models Using the Baseband or Passband SSB Modulators Must be Resaved

The baseband and passband SSB modulators have been updated for Release 13 to now include a pop-up menu enabling you to choose between upper and lower sideband modulation. You should resave any models using the old SSB modulators before running them in Release 13 to avoid producing Simulink warnings.

Change the Boolean Logic Signals Parameter to Off

The Communications Blockset does not support signals with `boolean` data type. In Release 13, the Simulink **Boolean logic signals** parameter is now set to **On** by default. If you use Simulink blocks, such as the Logical Operator block, together with Communications Blockset blocks in a model, you must change the default setting of the **Boolean logic signals** parameter setting to **Off**. To do so, type

```
commstartup
```

at the beginning of each MATLAB session, before you create a model. This sets the **Boolean logic signals** parameter to **Off** for every model you create during the current MATLAB session.

To manually change the **Boolean logic signals** parameter in a model to **Off**, do the following steps:

1. Select **Simulation parameters** from the **Simulation** menu on the toolbar.
2. Click the **Advanced** tab in the **Simulation Parameters** dialog box.
3. Select **Boolean logic signals** in the **Optimizations** field.

4 Under **Action**, select the **Off** check box.

5 Click **OK**.

Note that this changes the **Boolean logic signals** parameter to **Off** only for the current model.

Since the default setting of the **Boolean logic signals** parameter prior to Release 13 was **Off**, it is not necessary to make changes to models that you created prior to Release 13.

3

Communications Blockset 2.0.1 Release Notes

New Features

Note The Communications Blockset was a new product that was introduced with Release 12.0. The Communications Blockset incorporates the functionality of the blocks that were included in the Communications Toolbox 1.4 (Release 11), with the addition of the new features summarized below. The Communications Toolbox is described in the *Communications Toolbox 2.0 Release Notes*.

This section introduces the new features and enhancements added in the Communications Blockset 2.0.1 since the Communications Blockset 2.0 (Release 12.0).

For information about Communications Blockset features that are incorporated from the previous release, see "New Features" on page 4-2.

Setting Simulink Preferences Automatically

The new `commstartup.m` script sets certain Simulink preferences to values that are most appropriate for the simulation of communication systems. To use this script, type the command `commstartup` in your `startup.m` file or in the MATLAB Command Window.

Converting Between Bipolar and Unipolar Signals

The Utility Functions library contains new blocks that convert between bipolar and unipolar signals. The blocks are Bipolar to Unipolar Converter and Unipolar to Bipolar Converter.

Choosing Seeds for Random-Output Blocks

The `randseed` function is a new function that generates prime numbers for use as **Initial seed** parameters in blocks that produce random output. Compared to composite seeds, prime seeds yield output that has better statistical properties.

Using Error Counts to Control Simulation Duration

You can now configure the Error Rate Calculation block so that it automatically stops the simulation upon detecting a specified number of errors. You do not need to know in advance how long it will take to accumulate that many errors.

Choosing the Algorithm for Integrator Blocks

The Discrete Modulo Integrator block now allows you to choose the integration method using a mask parameter. The corresponding mask parameter in the Windowed Integrator block has changed its name from **Method** to **Integration method** for consistency with other integration blocks.

Major Bug Fixes

The Communications Blockset 2.0.1 includes several bug fixes, including the following:

- The M-FSK Baseband Modulator, M-FSK Baseband Demodulator, M-FSK Passband Modulator, and M-FSK Passband Demodulator blocks now use the correct tone spacing.
- The PN Sequence Generator block now generates only binary values, and the numbers in the sequence do not depend on the frame status or size.

Bug Fixes Incorporated from Release 12.0

The Communications Blockset 2.0.1 includes several bug fixes that were made in Release 12.0; see “Major Bug Fixes” on page 4-4.

Upgrading from an Earlier Release

If you are upgrading from an earlier release to the Communications Blockset 2.0.1, then note these issues:

- The Binary Symmetric Channel block dialog box now omits the **Input vector length** and **Sample time** parameters because the block now determines these quantities automatically. However, if you open a model in Release 12.1 that contains the Release 12.0 Binary Symmetric Channel block, then the Command Window might display warnings about block parameters. To suppress these warnings in the future, simply save the model from Release 12.1.
- In the Communications Blockset 2.0.1, any model that includes a digital passband modulator block or a digital passband demodulator block must use a variable-step solver rather than a fixed-step solver. To configure a model so that it uses a variable-step solver, select **Simulation parameters** from the model window's **Simulation** menu and then set the **Type** parameter on the **Solver** panel to **Variable-step**.

See "Upgrading from an Earlier Release" on page 4-5 in the Communications Blockset 2.0 Release Notes for upgrade issues involved in moving from the Communications Toolbox 1.4 (Release 11) to the Communications Blockset 2.0.1.

4

Communications Blockset 2.0 Release Notes

New Features

The Communications Blockset is a new product being introduced with Release 12. The Communications Blockset incorporates the functionality of the blocks that were included in the Communications Toolbox 1.4 (Release 11), with the addition of the new features summarized below.

Note The Communications Toolbox is described in a separate section.

Digital Modulation Libraries

The digital modulation libraries have been replaced with new ones. The new libraries contain baseband and passband sublibraries for:

- Amplitude modulation (PAM, QAM)
- Phase modulation (PSK, DPSK)
- Frequency modulation (FSK)
- Continuous phase modulation (CPM), including MSK and GMSK

For a list of blocks, see the reference sections for the baseband and passband digital modulation libraries. For a discussion of the capabilities of the new libraries, see "Digital Modulation" in the *Communications Blockset User's Guide*.

Interleaving Libraries

A new Interleaving library contains sublibraries for block interleaving and convolutional interleaving. These sublibraries support general block interleavers and general multiplexed interleavers, as well as several special cases of these. For more information, see "Interleaving" in the *Communications Blockset User's Guide*.

Fading Channels

The new Multipath Rayleigh Fading Channel and Rician Fading Channel blocks implement baseband simulations of fading propagation channels. These blocks model real-world mobile communication effects and are useful for modeling mobile wireless communication systems. For more information, see "Fading Channels" in the *Communications Blockset User's Guide*.

Enhanced Support for Convolutional Coding

The new APP Decoder block implements *a posteriori probability* decoding. The enhanced Convolutional Encoder and Viterbi Decoder blocks now support a more general class of convolutional codes by accepting a trellis parameter in their dialog boxes. The new `poly2trellis` function in the Communications Toolbox supports this enhancement, by converting a polynomial description of an encoder into a corresponding trellis description. For more information, see "Convolutional Coding" in the *Communications Blockset User's Guide*.

Sequence Operations

These new blocks in the Sequence Operations library manipulate data sequences in various ways:

- Bit to Integer Converter and Integer to Bit Converter convert between integers and their binary representations.
- Complex Phase Shift and Complex Phase Difference manipulate or analyze the phase of a complex signal.
- Derepeat is an inverse of the DSP Blockset's Repeat block.
- Interlacer and Deinterlacer can be useful for combining or separating in-phase and quadrature components of a signal.
- Puncture and Insert Zero are useful for processing punctured codes.

Major Bug Fixes

The Communications Blockset includes several bug fixes, including descriptions (online only) of particularly important bug fixes.

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

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving from the Communications Toolbox 1.4 (Release 11) to the Communications Blockset 2.0. Such issues relate to

- The new set of block libraries
- Support of new signal types, particularly frame-based signals and two-dimensional (that is, matrix) signals
- Functionality changes in specific blocks
- Blocks from the previous *Communications Toolbox User's Guide* that have changed their names in Version 2.0 of the Communications Blockset
- Blocks from the previous *Communications Toolbox User's Guide* that are not in Version 2.0 of the Communications Blockset

Note that before Release 12, the Communications Toolbox was a collection of functions and blocks, so that the previous *Communications Toolbox User's Guide* documented both functions and blocks. Release 12 is the first release of the Communications Blockset as a distinct product.

New Block Libraries in Release 12

The Communications Blockset uses a new set of block libraries, although it also includes the previous set of block libraries for backwards compatibility. The new set of libraries is what appears in the Simulink Browser (on PC) and what opens if you type `commlib` at the MATLAB prompt. You should build new models using this new set. New blocks in the new set of libraries are described in "New Features" on page 4-2.

Your previous models link to the previous set of libraries unless you choose to replace individual blocks manually. You can access the previous set of libraries by typing `commlib 1.5` at the MATLAB prompt.

Reorganization of Utility Functions in New Set of Libraries

The Utility Functions library has been reorganized. The table below lists blocks in Release 12 that were in the Release 11 Utility Functions library.

Block	New Location
Data Mapper	Utility Functions
Derepeat	Sequence Operations sublibrary
Descrambler	Sequence Operations sublibrary
Differential Decoder	Source Coding
Differential Encoder	Source Coding
Discrete Modulo Integrator (formerly called Discrete Time Modulo Integrator)	Integrators sublibrary
Discrete-Time VCO	Comm Sources
Windowed Integrator	Integrators sublibrary
Modulo Integrator	Integrators sublibrary
Integrate and Dump (formerly called Scheduled Reset Integrator)	Integrators sublibrary
Scrambler	Sequence Operations sublibrary
Voltage-Controlled Oscillator	Comm Sources

The Sequence Operations and Integrators sublibraries are in the Basic Comm Functions library.

Some blocks from the Release 11 Utility Functions library have become obsolete; for possible substitutions in Release 12, see “Obsolete Blocks from Previous Manual” on page 4-12.

New Signal Support

As of Release 12, Simulink supports matrix signals in addition to one-dimensional arrays, and frame-based signals in addition to sample-based signals. The Communications Blockset processes certain kinds of matrix and frame-based signals.

Because a future release is planned to include more comprehensive matrix and frame support, some Release 12 blocks avoid conflict with future features by using strict guidelines to determine the kinds of signals that they now accept. As a consequence, if you used vector signals in a model before Release 12, then you might need to use a particular kind of vector signal in Release 12 (such as a frame-based column vector, a frame-based row vector, or a sample-based vector of a particular shape or dimension).

As another consequence of frame support, the AWGN Channel and Derepeat blocks no longer have the **Frame-based inputs** check box and the **Number of channels** parameter as in the Communications Toolbox 1.5. Instead, these blocks inherit the frame status and number of channels from their inputs.

The *Communications Blockset User's Guide* includes more detail about signal support. To learn the terminology associated with signal attributes, see the Technical Conventions section. For general information about signal support in the Communications Blockset, see the "Signal Support" section.

Functionality Changes in Specific Blocks

Aside from signal support and naming issues, some blocks have changed the way that they behave:

- The Continuous-Time Eye and Scatter Diagrams and Discrete-Time Eye and Scatter Diagrams blocks process *complex* signals, whereas their counterparts before Release 12 (called Eye-Diagram Scatter Plot and Sample-Time Eye-Diagram Scatter) processed real vectors that listed in-phase and quadrature components separately.
- The blocks for Reed-Solomon and BCH coding no longer have a second input port for an enabler signal. The change affects the Binary-Input RS Encoder, Binary-Output RS Decoder, Integer-Input RS Encoder, Integer-Output RS Decoder, and BCH Decoder blocks.
- The Scrambler, Descrambler, and PN Sequence Generator blocks no longer have a trigger input. The Scrambler and Descrambler blocks no longer have

a state output. The PN Sequence Generator block produces output from the last register in the generator, not the first.

- The Convolutional Encoder and Viterbi Decoder blocks have new interfaces because they can now accept a more general trellis description of a convolutional encoder.
- The Version 1.4 Error Rate Calculation block considers a vector input to be a sample, whereas the current block considers a vector input to be a frame of multiple samples. For vector inputs of length n, a **Receive delay** parameter value of k in the Version 1.4 block is equivalent to a **Receive delay** of k*n in the current block.
- The Voltage-Controlled Oscillator block now uses the cosine, not sine, function to produce its waveform. This change affects the phase of the output signal.
- The blocks in the Synchronization library no longer use a **Gain at the output** parameter. The remaining parameters that define characteristics of the voltage-controlled oscillator have changed slightly. Also, the Baseband PLL and Linearized Baseband PLL blocks now include three output ports instead of one, to match the Phase-Locked Loop and Charge Pump PLL blocks.

Block Name Changes Since Previous Manual

The table below lists the old and new names of blocks that were part of the Communications Toolbox before Release 12 and that have changed their names. The old names are from the last printed version of the *Communications Toolbox User's Guide*. Because the libraries have been reorganized since that document was printed, the third column of the table lists the current library name for each block.

Names of Blocks in Version 1.x and Version 2, Where Different

Old Block Name (Version 1.x)	New Block Name (Version 2)	Library Location
ADM with Carrier	DSB AM Demodulator Passband	Analog Passband
ADM with Carrier CE	DSB AM Demodulator Baseband	Analog Baseband
AM with Carrier	DSB AM Modulator Passband	Analog Passband

Names of Blocks in Version 1.x and Version 2, Where Different (Continued)

Old Block Name (Version 1.x)	New Block Name (Version 2)	Library Location
AM with Carrier CE	DSB AM Modulator Baseband	Analog Baseband
BCH Decode Vector In/Out	BCH Decoder	Block Codes
BCH Encode Vector In/Out	BCH Encoder	Block Codes
Baseband Model PLL	Baseband PLL	Synchronization
Bernoulli Random Binary Noise Generator	Bernoulli Binary Generator	Comm Sources
Binary Error Channel	Binary Symmetric Channel	Channels
Cyclic Decode Vector In/Out	Binary Cyclic Decoder	Block Codes
Cyclic Encode Vector In/Out	Binary Cyclic Encoder	Block Codes
DPCM Decode	DPCM Decoder	Source Coding
DPCM Encode	DPCM Encoder	Source Coding
DSB-SC ADM	DSBSC AM Demodulator Passband	Analog Passband
DSB ADM CE	DSBSC AM Demodulator Baseband	Analog Baseband
DSB-SC AM	DSBSC AM Modulator Passband	Analog Passband
DSB AM CE	DSBSC AM Modulator Baseband	Analog Baseband
Discrete Time VCO	Discrete-Time VCO	Comm Sources
Discrete Time Modulo Integrator	Discrete Modulo Integrator	Integrators
Eye-Pattern & Scatter Plot	Continuous-Time Eye and Scatter Diagrams	Comm Sinks
FDM	FM Demodulator Passband	Analog Passband

Names of Blocks in Version 1.x and Version 2, Where Different (Continued)

Old Block Name (Version 1.x)	New Block Name (Version 2)	Library Location
FDM CE	FM Demodulator Baseband	Analog Baseband
FM	FM Modulator Passband	Analog Passband
FM CE	FM Modulator Baseband	Analog Baseband
Gaussian Random Noise Generator	Gaussian Noise Generator	Comm Sources
Hamming Decode Vector In/ Out	Hamming Decoder	Block Codes
Hamming Encode Vector In/ Out	Hamming Encoder	Block Codes
Linear Block Decode Vector In/Out	Binary Linear Decoder	Block Codes
Linear Block Encode Vector In/Out	Binary Linear Encoder	Block Codes
Linearized Baseband Model PLL	Linearized Baseband PLL	Synchronization
μ-Law Compressor	Mu-Law Compressor	Source Coding
μ-Law Expander	Mu-Law Expander	Source Coding
PDM	PM Demodulator Passband	Analog Passband
PDM CE	PM Demodulator Baseband	Analog Baseband
PLL	Phase-Locked Loop	Synchronization
PM	PM Modulator Passband	Analog Passband
PM CE	PM Modulator Baseband	Analog Baseband
Poisson Random Integer Generator	Poisson Integer Generator	Comm Sources
Quantization Decode	Quantizer Decode	Source Coding

Names of Blocks in Version 1.x and Version 2, Where Different (Continued)

Old Block Name (Version 1.x)	New Block Name (Version 2)	Library Location
Reed-Solomon Decode Binary Vector In/Out	Binary-Output RS Decoder	Block Codes
Reed-Solomon Decode Integer Vector In/Out	Integer-Output RS Decoder	Block Codes
Reed-Solomon Encode Binary Vector In/Out	Binary-Input RS Encoder	Block Codes
Reed-Solomon Encode Integer Vector In/Out	Integer-Input RS Encoder	Block Codes
Rician Random Noise Generator	Rician Noise Generator	Comm Sources
SSB ADM	SSB AM Demodulator Passband	Analog Passband
SSB ADM CE	SSB AM Demodulator Baseband	Analog Baseband
SSB-AM	SSB AM Modulator Passband	Analog Passband
SSB-AM CE	SSB AM Modulator Baseband	Analog Baseband
Sample Time Eye-Pattern Diagram & Scatter Plot	Discrete-Time Eye and Scatter Diagrams	Comm Sinks
Scheduled Reset Integrator	Integrate and Dump	Integrators
Signal Quantizer	Sampled Quantizer Encode	Source Coding
Triggered Signal Quantizer	Enabled Quantizer Encode	Source Coding
Uniform Random Noise Generator	Uniform Noise Generator	Comm Sources
Uniform Random Integer Generator	Random-Integer Generator	Comm Sources
VCO	Voltage-Controlled Oscillator	Comm Sources

Obsolete Blocks from Previous Manual

The table below lists blocks that appear in the previous version of the *Communications Toolbox User's Guide* but that are not included in the Release 12 Communications Blockset. Where applicable, the second column lists blocks that provide similar functionality. In some cases, the similar block requires different parameter settings, data formats, or signal attributes compared to the original block. Therefore, you should read the documentation for the similar block before using it in your model.

Blocks Not in v2, and Similar v2 Blocks

Obsolete Block	Similar Block(s), if Any
Array Function	See Math library in Simulink
BCH Code View Table	Use `bchpoly` in Communications Toolbox
BCH Decode Sequence In/Out	BCH Decoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
BCH Encode Sequence In/Out	BCH Encoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
Coherent MFSK Corr Demod	
Coherent MFSK Demod	
Coherent MFSK Demod CE	
Complex Filter	See Filtering library in DSP Blockset
Convolutional Decode Sequence In/Out	Viterbi Decoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
Convolutional Decode Vector In/Out	Viterbi Decoder

Blocks Not in v2, and Similar v2 Blocks (Continued)

Obsolete Block	Similar Block(s), if Any
Convolutional Encode Sequence In/Out	Convolutional Encoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
Convolutional Encode Vector In/Out	Convolutional Encoder
Cyclic Decode Sequence In/Out	Binary Cyclic Decoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
Cyclic Encode Sequence In/Out	Binary Cyclic Encoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
DPSK Demod	M-DPSK Demodulator Passband
DPSK Mod	M-DPSK Modulator Passband
D-TDMA Demux	
D-TDMA Mux	
Edge Detector	Edge Detector in DSP Blockset
Envelope Detector	Maximum, Minimum in DSP Blockset
Error Counter	Counter, in DSP Blockset
Error Rate Meter	Error Rate Calculation
Hamming Decode Sequence In/Out	Hamming Decoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*

Blocks Not in v2, and Similar v2 Blocks (Continued)

Obsolete Block	Similar Block(s), if Any
Hamming Encode Sequence In/ Out	Hamming Encoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
Hilbert Filter	Remez FIR Filter Design in DSP Blockset
Integer Scalar to Vector	Integer to Bit Converter
Integer Vector to Scalar	Bit to Integer Converter
Interleave	Matrix Interleaver
K-Step Delay	Integer Delay in DSP Blockset
Limited Binary Error Channel	Binary Vector Noise Generator
Linear Block Decode Sequence In/ Out	Binary Linear Decoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
Linear Block Encode Sequence In/Out	Binary Linear Encoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
MASK Demap	
MASK Demod	M-PAM Demodulator Passband
MASK Demod CE	M-PAM Demodulator Baseband
MASK Map	
MASK Mod	M-PAM Modulator Passband
MASK Mod CE	M-PAM Modulator Baseband
Mean and Variance	Mean, Variance in DSP Blockset

Blocks Not in v2, and Similar v2 Blocks (Continued)

Obsolete Block	Similar Block(s), if Any
Mean and Std	Mean, Standard Deviation in DSP Blockset
MFSK Map	
MFSK Mod	M-FSK Modulator Passband
MFSK Mod CE	M-FSK Modulator Baseband
Min/Max Demap	
Min/Max Index	Maximum, Minimum in DSP Blockset
Modulo	Math Function in Simulink
MPSK Correlation Demodulation	
MPSK Demod	M-PSK Demodulator Passband
MPSK Demod CE	M-PSK Demodulator Baseband
MPSK Map	
MPSK Mod	M-PSK Modulator Passband
MPSK Mod CE	M-PSK Modulator Baseband
MSK Demod	MSK Demodulator Passband
MSK Mod	MSK Modulator Passband
Noncoherent MFSK Corr Demod	
Noncoherent MFSK Demod	M-FSK Demodulator Passband
Noncoherent MFSK Demod CE	M-FSK Demodulator Baseband
Number Counter	Counter, in DSP Blockset
OQPSK Demod	OQPSK Demodulator Passband
OQPSK Mod	OQPSK Modulator Passband

Blocks Not in v2, and Similar v2 Blocks (Continued)

Obsolete Block	Similar Block(s), if Any
QADM	General QAM Demodulator Passband
QADM CE	General QAM Demodulator Baseband
QAM	General QAM Modulator Passband
QAM CE	General QAM Modulator Baseband
QASK Demap Arbitrary Constellation	
QASK Demap Circle Constellation	
QASK Demap Square Constellation	
QASK Demod Arbitrary Constellation	General QAM Demodulator Passband
QASK Demod CE Arbitrary Constellation	General QAM Demodulator Baseband
QASK Demod CE Circle Constellation	General QAM Demodulator Baseband
QASK Demod CE Square Constellation	Rectangular QAM Demodulator Baseband
QASK Demod Circle Constellation	General QAM Demodulator Passband
QASK Demod Square Constellation	Rectangular QAM Demodulator Passband
QASK Map Arbitrary Constellation	
QASK Map Square Constellation	

Blocks Not in v2, and Similar v2 Blocks (Continued)

Obsolete Block	Similar Block(s), if Any
QASK Mod Arbitrary Constellation	General QAM Modulator Passband
QASK Mod CE Arbitrary Constellation	General QAM Modulator Baseband
QASK Mod CE Circle Constellation	General QAM Modulator Baseband
QASK Mod CE Square Constellation	Rectangular QAM Modulator Baseband
QASK Mod Circle Constellation	General QAM Modulator Passband
QASK Mod Square Constellation	Rectangular QAM Modulator Passband
Raised Cosine Filter	
Rayleigh Fading CE Channel	Multipath Rayleigh Fading Channel
Rayleigh Noise CE Channel	Rayleigh Noise Generator
Reed-Solomon Decode Binary Sequence In/Out	Binary-Output RS Decoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
Reed-Solomon Decode Integer Sequence In/Out	Integer-Output RS Decoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
Reed-Solomon Encode Binary Sequence In/Out	Binary-Input RS Encoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*

Blocks Not in v2, and Similar v2 Blocks (Continued)

Obsolete Block	Similar Block(s), if Any
Reed-Solomon Encode Integer Sequence In/Out	Integer-Input RS Encoder. See "Using Serial Signals" in the *Communications Blockset User's Guide*
Register Shift	Queue in DSP Blockset
Rician Noise CE Channel	Rician Noise Generator
Sampled Read From Workspace	Signal From Workspace in DSP Blockset
Sinc	
Time-Share Demux	
Time-Share Mux	
Triggered Read from Workspace	Triggered Signal From Workspace in DSP Blockset
Triggered Write to Workspace	Triggered To Workspace in DSP Blockset
Varying AWGN Channel	
Varying Rayleigh Fading CE Channel	
Varying Rayleigh Noise CE Channel	
Varying Rician Noise CE Channel	
Vector Pulse	Discrete Pulse Generator in Simulink
Vector Redistributor	