

Communications System Toolbox 5.0

Design and simulate the physical layer of communication systems

Communications System Toolbox provides algorithms and tools for the design, simulation, and analysis of communications systems. These capabilities are provided as **MATLAB**[®] functions, MATLAB System objects, and **Simulink**[®] blocks. The system toolbox includes algorithms for source coding, channel coding, interleaving, modulation, equalization, synchronization, and channel modeling. Tools are provided for bit error rate analysis, generating eye and constellation diagrams, and visualizing channel characteristics.

The system toolbox also provides adaptive algorithms that let you model dynamic communications systems that use OFDM, OFDMA, and MIMO techniques. Algorithms support fixed-point data arithmetic and C or HDL code generation.

The system toolbox includes sample models that provide a starting point for implementing MIMO, OFDM, and other advanced communication system architectures.

Key Features

- Algorithms available as MATLAB functions, MATLAB System objects, and Simulink blocks
- Algorithms for designing the physical layer of communications systems, including source coding, channel coding, interleaving, modulation, channel models, equalization, and synchronization
- Visualization tools, including eye diagrams, constellations, and channel scattering functions
- Graphical tool for comparing the bit error rate of a system with analytical results
- Channel models, including AWGN, Multipath Rayleigh Fading, Rician Fading, COST 207, GSM/EDGE, HF ionospheric, and MIMO
- Interactive tool for visualizing time-varying communications channels
- Basic RF impairments, including nonlinearity, phase noise, thermal noise, and phase and frequency offsets
- Support for fixed-point modeling and C and HDL code generation

System Design, Characterization, and Visualization

The design and simulation of a communications system requires analyzing its response to the noise and interference inherent in real-world environments, studying its behavior using graphical and quantitative means, and determining whether the resulting performance meets standards of acceptability.

Communications System Toolbox implements a variety of tasks for communications system design and simulation. Many of the functions, System objects, and blocks in the system toolbox perform computations associated with a particular component of a communications system, such as a demodulator or equalizer. Other capabilities are designed for visualization or analysis.

System Characterization

The system toolbox offers several standard methods for quantitatively characterizing system performance:

- Bit error rate (BER) computations
- Adjacent channel power ratio (ACPR) measurements
- Error vector magnitude (EVM) measurements
- Modulation error ratio (MER) measurements

Because BER computations are fundamental to the characterization of any communications system, the system toolbox provides the following tools and capabilities for configuring BER test scenarios and accelerating BER simulations:

BERtool — A graphical user interface that enables you to analyze BER performance of communications systems. You can analyze performance via a simulation-based, semianalytic, or theoretical approach.

Error Rate Test Console — A [MATLAB](#) object that runs simulations for communications systems to measure error rate performance. It supports user-specified test points and generation of parametric performance plots and surfaces. Accelerated performance can be realized when running on a multicore computing platform.

Multicore and GPU acceleration — A capability provided by [Parallel Computing Toolbox™](#) that enables you to accelerate simulation performance using multicore and GPU hardware within your computer.

Distributed computing and cloud computing support — Capabilities provided by Parallel Computing Toolbox and [MATLAB Distributed Computing Server™](#) that enable you to leverage the computing power of your server farms and the Amazon EC2 Web service.

Performance Visualization

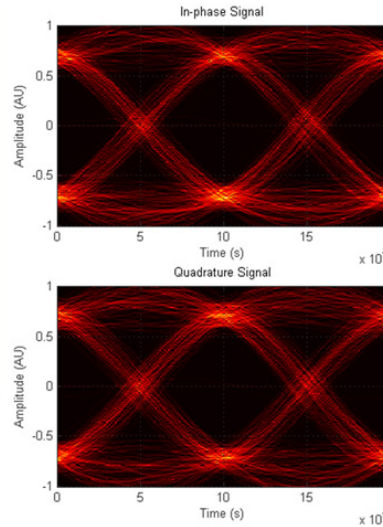
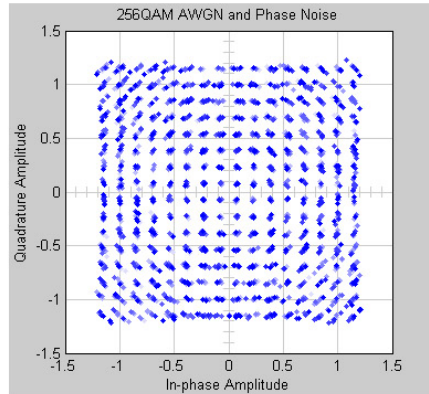
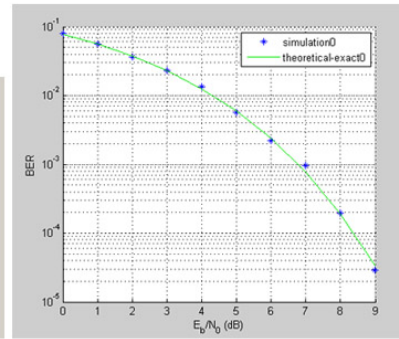
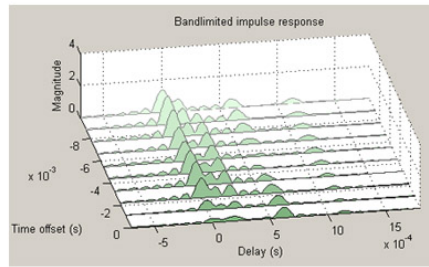
The system toolbox provides the following capabilities for visualizing system performance:

Channel visualization tool — For visualizing the characteristics of a fading channel

Eye diagrams and signal constellation scatter plots — For a qualitative, visual understanding of system behavior that enables you to make initial design decisions

Signal trajectory plots — For a continuous picture of the signal's trajectory between decision points

BER plots — For visualizing quantitative BER performance of a design candidate, parameterized by metrics such as SNR and fixed-point word size



Communication-specific displays for visualizing and analyzing signals at any point or step in your model. Displays include (clockwise from top left): Channel impulse response history, I/Q signal eye diagrams, BER performance plot for theoretical vs. simulated results, and received signal scatter plot.

Analog and Digital Modulation

Analog and digital modulation techniques encode the information stream into a signal that is suitable for transmission. Communications System Toolbox provides a number of modulation and corresponding demodulation capabilities. These capabilities are available as [MATLAB](#) functions and objects, [MATLAB System](#) objects, and [Simulink](#) blocks.

Modulation types provided by the toolbox are:

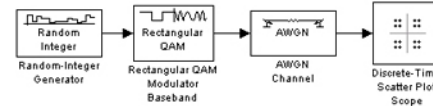
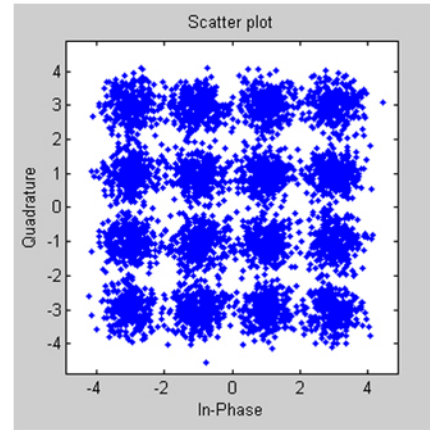
Analog, including AM, FM, PM, SSB, and DSBSC

Digital, including FSK, PSK, BPSK, DPSK, OQPSK, MSK, PAM, QAM, and TCM

```

C:\Program Files\MATLAB\R2011a\work\QAM16_modulation.m*
File Edit Text Go Cell Tools Debug Desktop Window Help
- 1.0 + + 1.1 x
1 % Create a random digital message
2 M = 16; % Alphabet size
3 x = randi([0 M-1],5000,1); % Random symbols
4
5 % Use 16-QAM modulation.
6 hMod = modem.qammod(M);
7 hDemod = modem.qamdemod(hMod);
8
9 % Create a scatter plot and show constellation
10 scatterPlot = commscope.ScatterPlot('SamplesPerSymbol',1,...
11 'Constellation',hMod.Constellation);
12 scatterPlot.PlotSettings.Constellation = 'on';
13
14 % Modulate
15 y = modulate(hMod,x);
16
17 % Transmit signal through an AWGN channel.
18 ynoisy = awgn(y,15,'measured');
19
20 % Create scatter plot from noisy data.
21 update(scatterPlot,ynoisyy);
22
23 % Demodulate ynoisy to recover the message.
24 z = demodulate(hDemod,ynoisyy);
25
26 % Check symbol error rate.
27 [num,zt] = symerr(x,z);
script Ln 29 Col 1 OVR

```



MATLAB function (left) and Simulink model (right) with scatter plot for 16 QAM simulation.

Source and Channel Coding

Communications System Toolbox provides source and channel coding capabilities that let you develop and evaluate communications architectures quickly, enabling you to explore what-if scenarios and avoid the need to create coding capabilities from scratch.

Source Coding

Source coding, also known as quantization or signal formatting, is a way of processing data in order to reduce redundancy or prepare it for later processing. The system toolbox provides a variety of types of algorithms for implementing source coding and decoding, including:

- Quantizing
- Companding (μ -law and A-law)
- Differential pulse code modulation (DPCM)
- Huffman coding
- Arithmetic coding

Channel Coding

To combat the effects noise and channel corruption, the system toolbox provides block and convolutional coding and decoding techniques to implement error detection and correction. For simple error detection with no inherent correction, a cyclic redundancy check capability is also available. Channel coding capabilities provided by the system toolbox include:

- BCH encoder and decoder
- Reed-Solomon encoder and decoder
- LDPC encoder and decoder
- Convolutional encoder and Viterbi decoder
- Orthogonal space-time block code (OSTBC) (encoder and decoder for MIMO channels)

- Turbo encoder and decoder demos

The system toolbox provides utility functions for creating your own channel coding. You can create generator polynomials and coefficients and syndrome decoding tables, as well as product parity-check and generator matrices.

The system toolbox also provides block and convolutional interleaving and deinterleaving functions to reduce data errors caused by burst errors in a communication system:

Block, including General block interleaver, algebraic interleaver, helical scan interleaver, matrix interleaver, and random interleaver

Convolutional, including General multiplexed interleaver, convolutional interleaver, and helical interleaver

Channel Modeling and RF Impairments

Channel Modeling

Communications System Toolbox provides algorithms and tools for modeling noise, fading, interference, and other distortions that are typically found in communications channels. The system toolbox supports the following types of channels:

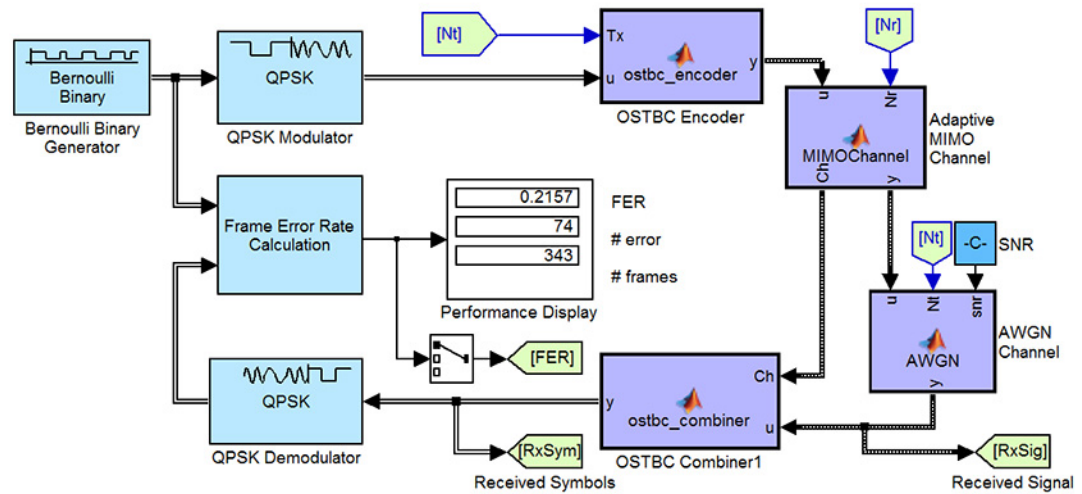
- Additive white Gaussian noise (AWGN)
- Multiple-input multiple-output (MIMO) fading
- Single-input single-output (SISO), Rayleigh, and Rician fading
- Binary symmetric

A [MATLAB](#) channel object provides a concise, configurable implementation of channel models, enabling you to specify parameters such as:

- Path delays
- Average path gains
- Maximum Doppler shifts
- K-Factor for Rician fading channels
- Doppler spectrum parameters

For MIMO systems, the MATLAB MIMO channel object expands these parameters to also include:

- Number of transmit antennas (up to 8)
- Number of receive antennas (up to 8)
- Transmit correlation matrix
- Receive correlation matrix



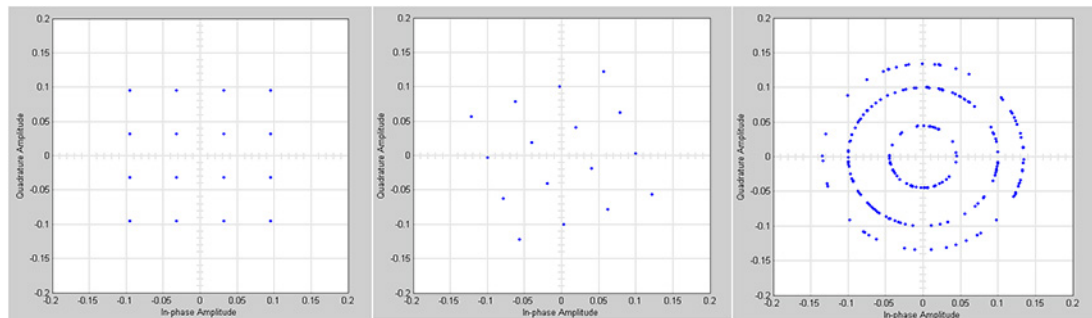
Simulink model of an adaptive MIMO system with orthogonal space-time block codes (OSTBC).

RF Impairments

To model the effects of a nonideal RF front end, you can introduce the following impairments into your communications system, enabling you to explore and characterize performance with real-world effects:

- Memoryless nonlinearity
- Phase and frequency offset
- Phase noise
- Thermal noise

You can include more complex RF impairments and RF circuit models in your design using [SimRF™](#).



An ideal 16 QAM scatter plot (left) impaired by a phase offset (middle) and a frequency offset (right).

Equalization and Synchronization

Communications System Toolbox lets you explore equalization and synchronization techniques. These techniques are generally adaptive in nature and challenging to design and characterize. The system toolbox provides algorithms and tools that let you rapidly select the appropriate technique in your communications system.

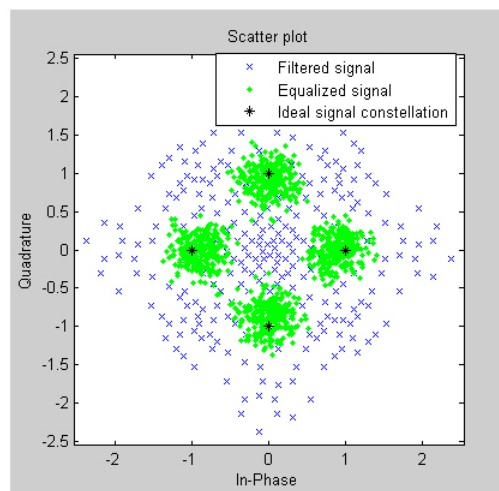
Equalization

To evaluate different approaches to equalization, the system toolbox provides you with adaptive algorithms such as:

- LMS

- Normalized LMS
- Variable step LMS
- Signed LMS
- MLSE (Viterbi)
- RLS
- CMA

These adaptive equalizers are available as nonlinear decision feedback equalizer (DFE) implementations and as linear (symbol or fractionally spaced) equalizer implementations.



Scatter plot of a QPSK signal that shows the signal before and after equalization, as well as the ideal signal constellation.

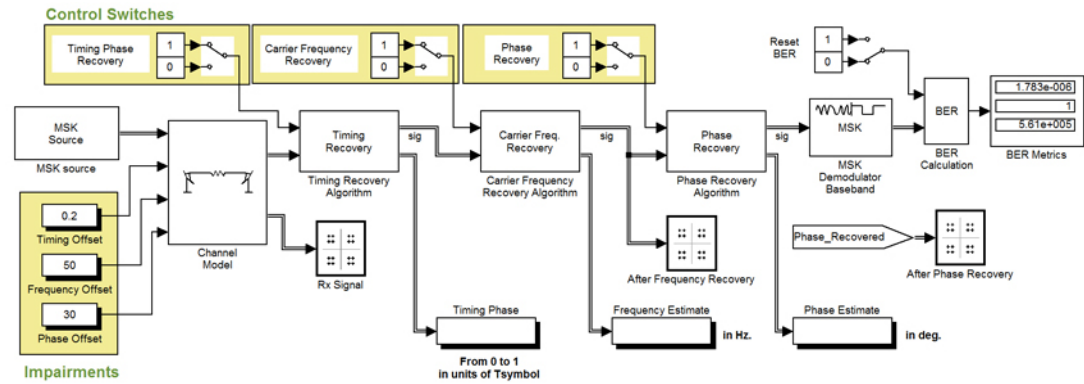
Synchronization

The system toolbox provides algorithms for both carrier phase synchronization and timing phase synchronization.

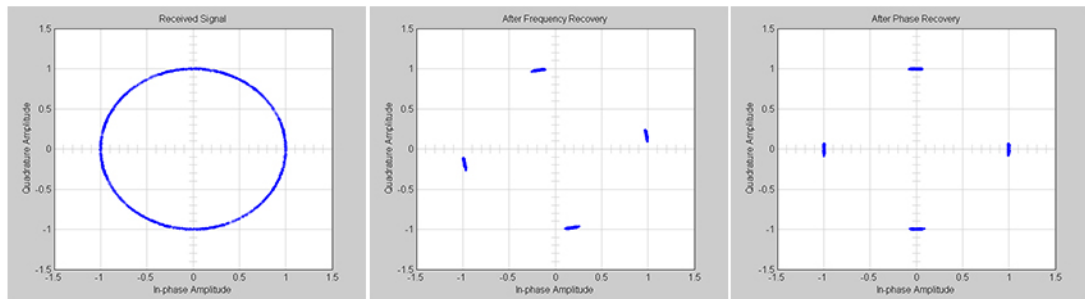
For timing phase synchronization, the system toolbox provides a [MATLAB Timing Phase Synchronizer](#) object that offers the following implementation methods:

- Early-late gate timing method
- Gardner's method
- Fourth-order nonlinearity method

- Mueller-Muller method



Simulink model of timing, carrier frequency, and carrier phase recovery for an MSK receiver.



Received signal scatter plot (left), after frequency recovery (middle), and after phase recovery (right).

Stream Processing in MATLAB and Simulink

Most communication systems handle streaming and frame-based data using a combination of temporal processing and simultaneous multifrequency and multichannel processing. This type of streaming multidimensional processing can be seen in advanced communication architectures such as OFDM and MIMO. Communications System Toolbox enables the simulation of advanced communications systems by supporting stream processing and frame-based simulation in [MATLAB](#) and [Simulink](#).

In MATLAB, stream processing is enabled by [System objects](#), which use MATLAB objects to represent time-based and data-driven algorithms, sources, and sinks. System objects implicitly manage many details of stream processing, such as data indexing, buffering, and management of algorithm state. You can mix System objects with standard MATLAB functions and operators. Most System objects have a corresponding Simulink block with the same capabilities.

Simulink handles stream processing implicitly by managing the flow of data through the blocks that make up a Simulink model. Simulink is an interactive graphical environment for modeling and simulating dynamic systems that uses hierarchical diagrams to represent a system model. It includes a library of general-purpose, predefined blocks to represent algorithms, sources, sinks, and system hierarchy.

Implementing a Communications System

Fixed-Point Modeling

Many communications systems use hardware that requires a fixed-point representation of your design. Communications System Toolbox supports fixed-point modeling in all relevant blocks and System objects with tools that help you configure fixed-point attributes.

Fixed-point support in the system toolbox includes:

- Word sizes from 1 to 128 bits
- Arbitrary binary-point placement
- Overflow handling methods (wrap or saturation)
- Rounding methods: ceiling, convergent, floor, nearest, round, simplest, and zero

Fixed-Point Tool in [Simulink Fixed Point™](#) facilitates the conversion of floating-point data types to fixed point. For configuration of fixed-point properties, the tool tracks overflows and maxima and minima.

Code Generation

Once you have developed your algorithm or communications system, you can automatically generate C code from it for verification, rapid prototyping, and implementation. Most System objects, functions, and blocks in Communications System Toolbox can generate ANSI/ISO C code using [MATLAB Coder™](#), [Simulink Coder™](#), or [Embedded Coder™](#). A subset of System objects and [Simulink](#) blocks can also generate HDL code.

To leverage existing intellectual property, you can select optimizations for specific processor architectures and integrate legacy C code with the generated code. You can also generate C code for both floating-point and fixed-point data types.

DSP Prototyping

DSPs are used in communication system implementation for verification, rapid prototyping, or final hardware implementation. Using the processor-in-the-loop (PIL) simulation capability found in Embedded Coder, you can verify generated source code and compiled code by running your algorithm's implementation code on a target processor.

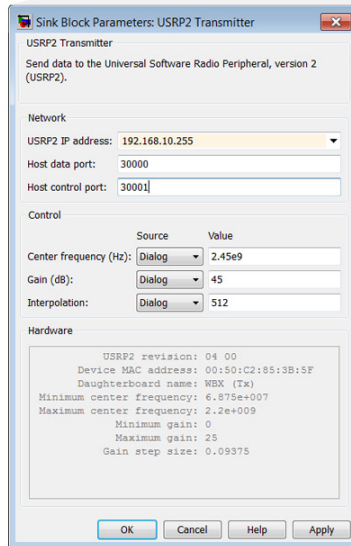
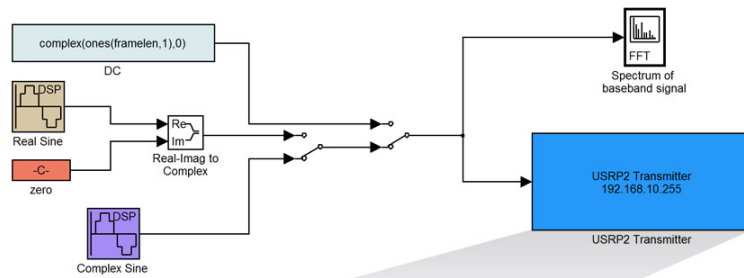
FPGA Prototyping

FPGAs are used in communication systems for implementing high-speed signal processing algorithms. Using the FPGA-in-the-loop (FIL) capability found in [EDA Simulator Link™](#), you can test RTL code in real hardware for any existing HDL code, either manually written or automatically generated HDL code.

USRP2 SDR Device Interface

To support the real-time processing of received signals and the real-time generation of waveforms for transmission, the system toolbox provides a Simulink interface to the Universal Software Radio Peripheral 2 (USRP2) software defined radio (SDR) device.

The interface consists of two Simulink blocks: a receiver block to stream data from the USRP2 and a transmitter block to stream data to the USRP2. Both blocks provide a concise configuration capability, consisting of specifying operating center frequency, gain, and interpolation/decimation rate.



Simulink transmitter test bench showing USRP2 Transmitter block (top right) and parameter mask (bottom).

Resources

Product Details, Demos, and System Requirements

www.mathworks.com/products/communications

Trial Software

www.mathworks.com/trialrequest

Sales

www.mathworks.com/contactsales

Technical Support

www.mathworks.com/support

Online User Community

www.mathworks.com/matlabcentral

Training Services

www.mathworks.com/training

Third-Party Products and Services

www.mathworks.com/connections

Worldwide Contacts

www.mathworks.com/contact