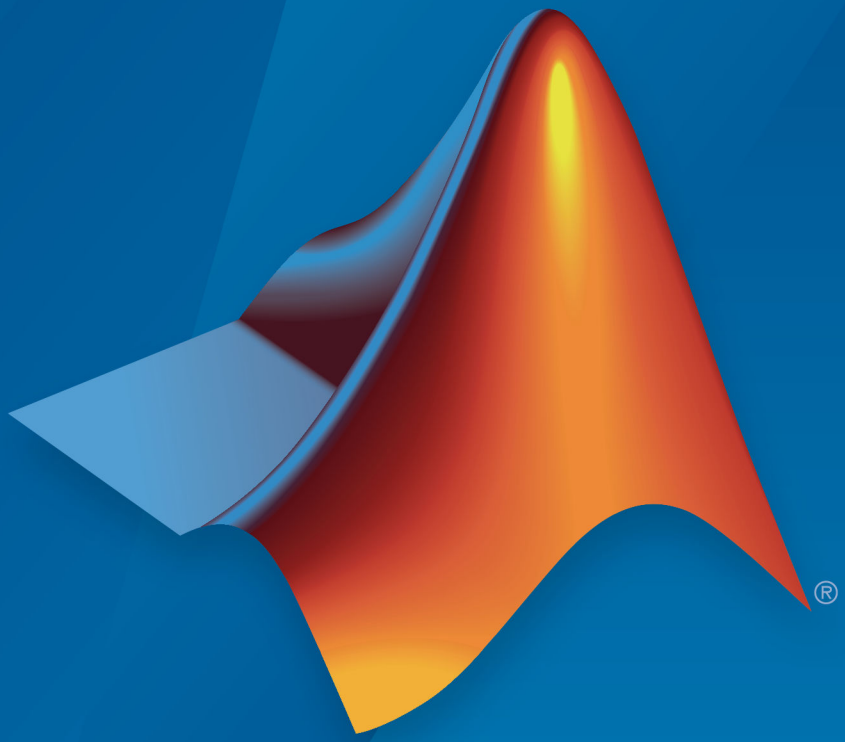
RF Blockset™

User's Guide



MATLAB® & SIMULINK®

R2018a

MathWorks®

How to Contact MathWorks

Latest news: `www.mathworks.com`

Sales and services: `www.mathworks.com/sales_and_services`

User community: `www.mathworks.com/matlabcentral`

Technical support: `www.mathworks.com/support/contact_us`

Phone: 508-647-7000

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

RF Blockset ™ User's Guide

Revision History

September 2010	Online only	New for Version 3.0 (Release 2010b)
April 2011	Online only	Revised for Version 3.0.2 (Release 2011a)
September 2011	Online only	Revised for Version 3.1 (Release 2011b)
March 2012	Online only	Revised for Version 3.2 (Release 2012a)
September 2012	Online only	Revised for Version 3.3 (Release 2012b)
March 2013	Online only	Revised for Version 4.0 (Release 2013a)
September 2013	Online only	Revised for Version 4.1 (Release 2013b)
March 2014	Online only	Revised for Version 4.2 (Release 2014a)
October 2014	Online only	Revised for Version 4.3 (Release 2014b)
March 2015	Online only	Revised for Version 4.4 (Release 2015a)
September 2015	Online only	Revised for Version 4.5 (Release 2015b)
March 2016	Online only	Revised for Version 5.0 (Release 2016a)
September 2016	Online only	Revised for Version 5.1 (Release 2016b)
March 2017	Online only	Revised for Version 6.0 (Release 2017a)
September 2017	Online only	Revised for Version 6.1 (Release 2017b)
March 2018	Online only	Revised for Version 7.0 (Release 2018a)

Contents

Circuit Envelope

1 Sensitivity

2 Interference

3 Intermodulation Distortion

4 Testbenches

5 Analog Devices Transceiver Models

Equivalent Baseband

6 Model an RF System

7 Plot Model Data

A RF Blockset Equivalent Baseband Algorithms

B Model Mixers

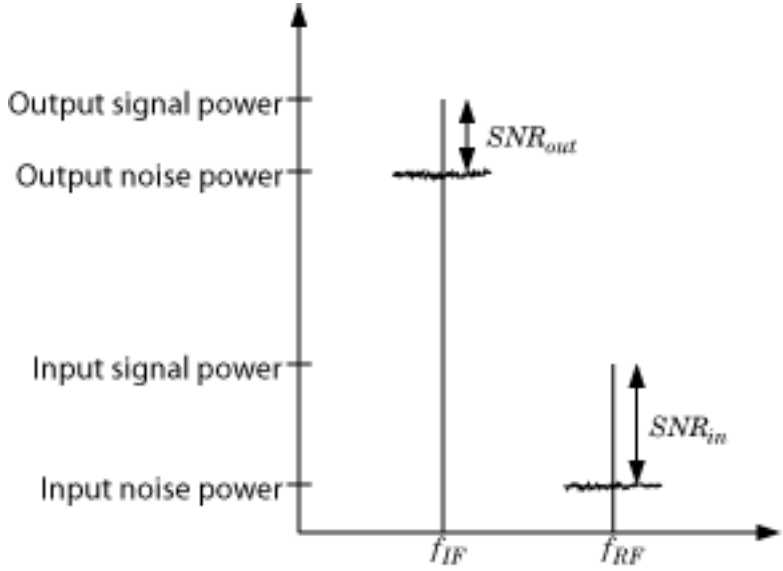
8 Examples

Circuit Envelope

Sensitivity

Model System Noise Figure

RF receivers amplify signals and shift them to lower frequencies. The receiver itself introduces noise that degrades the received signal. The signal-to-noise ratio (SNR) at the receiver output ultimately determines the usability of the receiver.



The preceding figure illustrates the effect of the receiver on the signal. The receiver amplifies a low-power RF signal at the carrier f_{RF} with a high SNR and downconverts the signal to f_{IF}. The noise figure (NF) of the system determines the difference between the SNR at the output and the SNR at the input:

$$SNR_{out} = SNR_{in} - NF_{sys}$$

where the difference is calculated in decibels. Excessive noise figure in the system causes the noise to overwhelm the signal, making the signal unrecoverable.

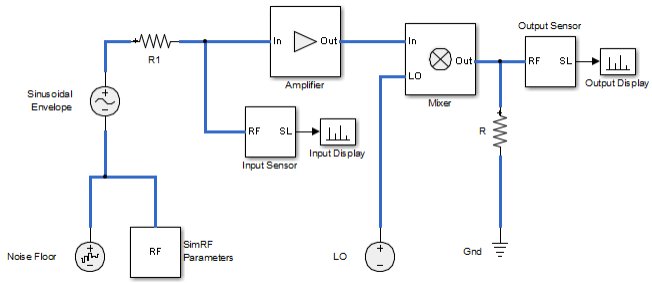
Create a Low-IF Receiver Model

The model `ex_simrf_snr` simulates a simplified IF receiver architecture. A Sinusoid block and a Noise block model a two-tone input centered at f_{RF} and low-level thermal

noise. The RF system amplifies the signal and mixes it with the local oscillator f_{LO} down to an intermediate frequency f_{IF}. A voltage sensor recovers the signal at the IF.

To open this model, at MATLAB® command line, enter:

```
addpath(fullfile(docroot,'toolbox','simrf','examples'))
ex_simrf_snr
```



The amplifier contributes 40 dB of gain and a 15-dB noise figure, and the mixer contributes 0 dB of gain and a 20-dB noise figure, which are values characteristic of a relatively noisy, high-gain receiver. The two-tone input has a specified level of .1 µV. A 1-V level in the local oscillator ensures consistency with the formulation of the conversion gain of the mixer.

To run the model:

1. Open the model by clicking the link or by typing the model name at the Command Window prompt.
2. Select **Simulation** > **Run**.

Set Up the RF Blockset Environment

To maximize performance, the **Fundamental tones** and **Harmonic order** parameters specify the simulation frequencies explicitly in the Configuration block:

- f_{LO}, the frequency of the LO in the first mixing stage, equals 1.9999 GHz. and appears in the list of fundamental tones as `carriers.LO`.
- f_{RF}, the carrier of the desired signal, equals 2 GHz and appears in the list of fundamental tones as `carriers.RF`.
- f_{IF}, the intermediate frequency, equals $f_{RF} - f_{LO}$. The frequency is a linear combination of the first-order (fundamental) harmonics of f_{LO} and f_{RF}. Setting **Harmonic order** to 1

is sufficient to ensure this frequency appears in the simulation frequencies. This minimal value for the harmonic order ensures a minimum of simulation frequencies.

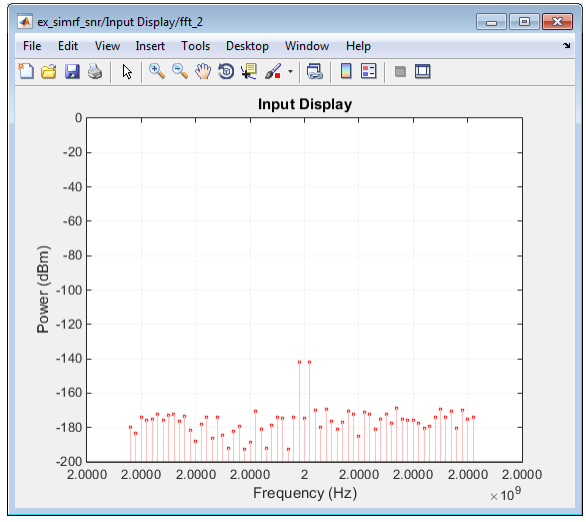
Solver conditions and noise settings are also specified for the Configuration block:

- The **Solver type** is set to `auto`. For more information on choosing solvers, see the reference page for the Configuration block or see Choosing Simulink and Simscape Solvers (Simscape).
- The **Sample time** parameter is set to `1/(mod_freq*64)`. This setting ensures a simulation bandwidth 64 times greater than the envelope signals in the system.
- The **Simulate noise** box is checked, so the environment includes noise parameters during simulation.

View Simulation Output

The model uses subsystems with a MATLAB Coder™ implementation of a fast Fourier transform (FFT) to generate two plots. The FFT uses 64 bins, so for a sampling frequency of 64 Hz, the bandwidth of each bin is 1 Hz. Subsequently, the power levels shown in the figures also represent the power spectral density (PSD) of the signals in dBm/Hz.

- The Input Display plot shows the power spectrum of the signal and noise at the input of the receiver.



The measured power of each tone is consistent with the expected power level of a 0.1-μV two-tone envelope:

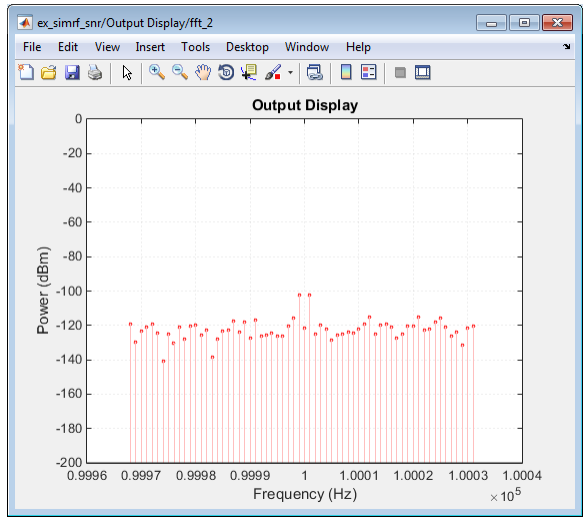
$$P_{in} = 10\log_{10}\left(\frac{V^2}{2R}\right) + 30$$

$$= 10\log_{10}\left(\frac{\left(\frac{1}{2}\cdot\frac{10^{-7}}{2}\right)^2}{2\cdot 50}\right) + 30 = -142 \text{ dBm}$$

A factor of 1/2 is due to voltage division across source and load resistors, and another factor of 1/2 is due to envelope scaling. See the featured example Two-Tone Envelope Analysis Using Real Signals for more discussion on scaling envelope signals for power calculation.

The measured noise floor at -177 dBm/Hz is reduced by 3 dB from the specified -174 dBm/Hz noise floor. The difference is due to power transfer from the source to the input of the amplifier. The amplifier also models a thermal noise floor, so although this decrease is unrealistic, it does not affect accuracy at the output stage.

- The Output Display plot shows the power spectrum of the signal and noise at the output of the receiver.



The measured PSD of -102 dBm/Hz for each tone is consistent with the 40-dB combined gain of the amplifier and mixer. The noise PSD in the figure is shown to be approximately 50 dB higher at the output, due to the gain and noise figure of the system.

If you have DSP System Toolbox™ software installed, you can replace the MATLAB Coder subsystems with a Spectrum Analyzer block.

Simulating Thermal Noise Floor

Thermal noise power can be modeled according to the equation

$$P_{noise} = 4k_BTR_s\Delta f$$

where:

- k_B is Boltzmann's constant, equal to 1.38065×10^{-23} J/K.
- T is the noise temperature, specified as 293.15 K in this example.
- R_s is the noise source impedance, specified as 50 Ω in this example to agree with the resistance value of the Resistor block labeled `R1`.
- Δf is the noise bandwidth.

To model the noise floor on the RF signal at the resistor, the model includes a Noise block:

- The **Noise Power Spectral Density (Watts/Hz)** parameter is calculated as $P_{noise} / \Delta f = 4k_B T R_s$.
- The **Carrier frequencies** parameter, set to `carriers.RF`, places noise on the RF carrier only.

Computing System Noise Figure

To model RF noise from component noise figures:

1. Select **Simulate noise** in the RF Blockset Parameters block dialog box, if it is not already selected.
2. Specify a value for the **Noise figure (dB)** parameter of an Amplifier and Mixer blocks.

The noise figures are not strictly additive. The amplifier contributes more noise to the system than the mixer because it appears first in the cascade. To calculate the total noise figure of the RF system with *n* stages, use the Friis equation:

$$F_{sys} = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2} + \ldots + \frac{F_n - 1}{G_1 G_2 \ldots G_{n-1}}$$

where F_i and G_i are the noise factor and gain of the ith stage, and $NF_i = 10\log_{10}(F_i)$.

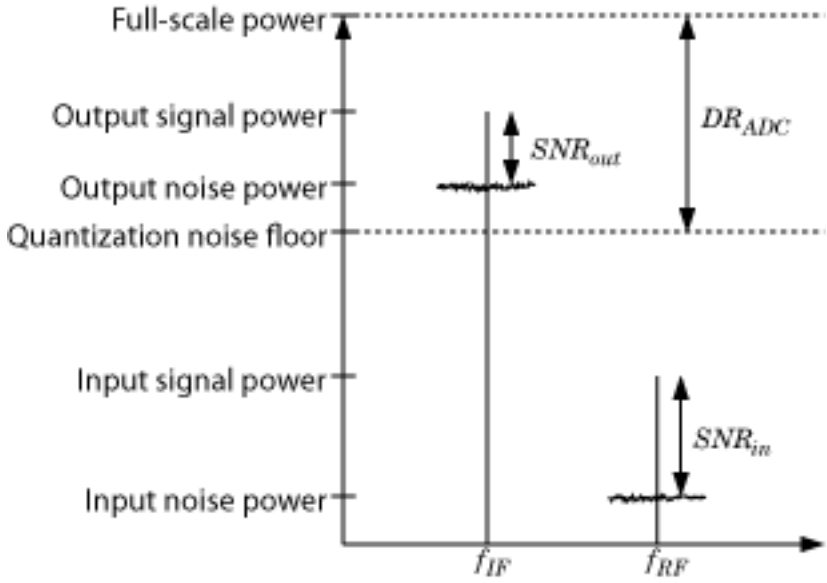
In this example, the noise figure of the amplifier is 10 dB, and the noise figure of the mixer is 15 dB, so the noise figure of the system is:

$$10\log_{10}\left(10^{10/10} + \frac{10^{15/10} - 1}{10000}\right) = 10.0 \text{ dB}$$

The Friis equation shows that although the mixer has a higher noise figure, the amplifier contributes more noise to the system.

Designing a Receiver with an ADC

Most RF receivers in modern communications or radar systems feed signals to an analog-to-digital converter (ADC). Due to their finite resolution, ADCs introduce quantization error into the system. The resolution of the ADC is determined by the number of bits and the full-scale (FS) range of the ADC.



The preceding figure illustrates an RF signal that falls within the dynamic range (DR) of an ADC. The input signal and noise at the carrier f_{RF} has high signal-to-noise ratio (SNR). The received signal at f_{IF} has reduced SNR due to system noise figure. However, if the quantization error is near or above the receiver noise, system performance degrades.

To ensure that the ADC contributes no more than 0.1 dB of noise to the signal at f_{IF}, the quantization noise floor must be 16 dB lower than the receiver noise. This condition can be met by:

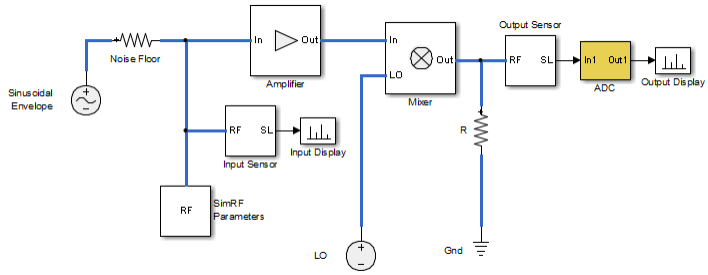
- Reducing the full-scale (FS) range or increasing the resolution of the ADC, which lowers the quantization noise floor.
- Increasing the gain of the RF receiver, which raises the receiver noise floor.

Overcome Quantization Error of an ADC

The model `ex_simrf_adc` simulates a low-IF receiver with an ADC. This model is based on the model `ex_simrf_snr` described in the section “Create a Low-IF Receiver Model” on page 1-2. At the output of the RF system, the ADC subsystem models an ADC with an FS range of `sqrt(100e-3)` V and a resolution of 16 bits.

To open this model, at MATLAB command line, enter:

```
addpath(fullfile(docroot,'toolbox','simrf','examples'))
ex_simrf_adc
```



The power of a voltage signal at the full-scale range of the ADC is

$$10\log_{10}\sqrt{100^{-3}}+30=0\ \text{dBm}$$

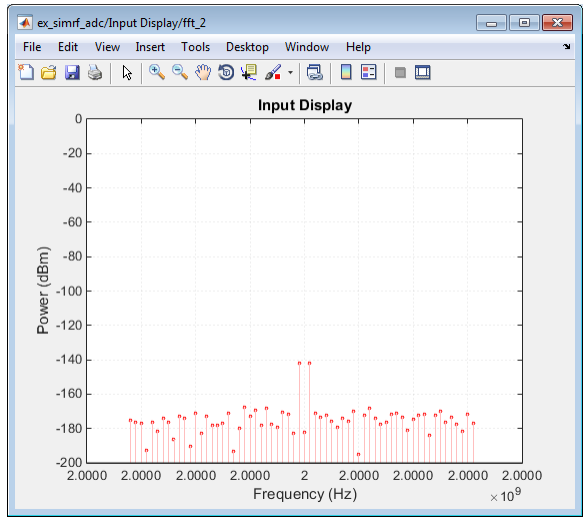
To maximize performance, the model uses the same simulation settings as `ex_simrf_snr`. To run this model:

1. Open the model by clicking the link or by typing the model name at the Command Window prompt.
2. Select **Simulation** > **Run**.

View Simulation Output

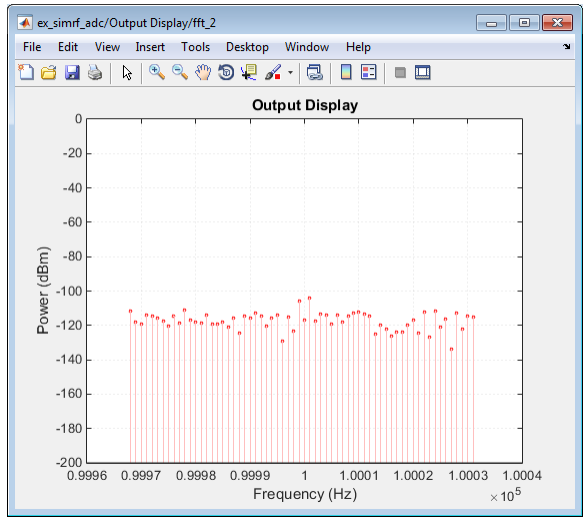
The model uses subsystems with a MATLAB Coder implementation of a fast Fourier transform (FFT) to generate two plots. The FFT uses 64 bins, so for a sampling frequency of 64 Hz, the bandwidth of each bin is 1 Hz. Subsequently, the power levels shown in the figures also represent the power spectral density (PSD) of the signals in dBm/Hz.

- The Input Display plot shows the power spectrum of the two-tone signal and noise at the input of the receiver-ADC system.



  The measured power of each tone of -142 dBm is consistent with the expected power level of a .1-μV signal. The power level of the noise is consistent with a -174 dBm/Hz noise floor.

- The Output Display plot shows power spectrum of the output signal.



The quantization error exceeds the receiver noise.

If you have DSP System Toolbox software installed, you can replace the MATLAB Coder subsystems with a Spectrum Analyzer block.

Measuring the Quantization Noise Floor

To calculate the quantization noise floor (QNF) of the ADC, subtract the dynamic range from the full-scale power, which is 0 dBm. To calculate the dynamic range PSD for the ADC, use the equation:

$$DR_{ADC} = 6.02 \cdot N_{bits} + 10\log_{10}(\Delta f) + 1.76 = 116.1 \text{ dBm/Hz}$$

where

- N_{bits} is the resolution. The ADC in this example uses 16 bits.
- Δf is the bandwidth of the FFT, which is 64 in this example. Oversampling in an ADC yields lower quantization noise.
- The value 1.76 is a correction factor for a pure sinusoidal input.

Therefore, the quantization noise floor is -116 dBm/Hz, in agreement with the measured output levels.

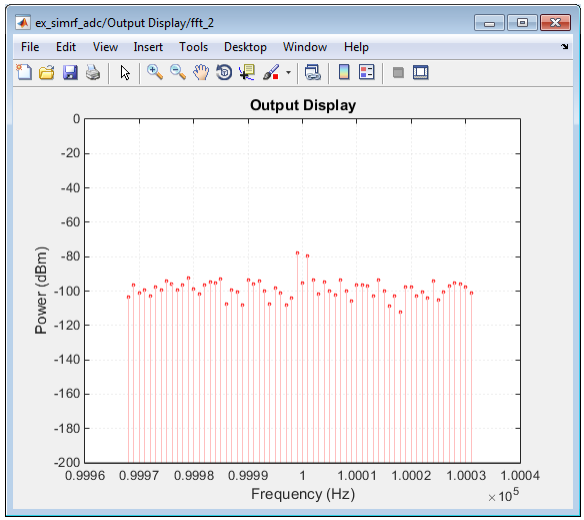
Improving Receiver-ADC Performance

Increasing the gain in the mixer raises the receiver noise without increasing the noise figure. Calculate the mixer gain required to achieve a 16-dB margin between the quantization noise floor and the receiver noise:

$$\begin{aligned} G_{mixer} &= (QNF_{ADC} + 16) - (-174 + G_{sys} + NF_{sys}) \\ &= (-116.1 + 16) - (-174 + 40 + 10.0) \\ &= -100.1 + 124.0 \\ &= 23.9 \text{ dB} \end{aligned}$$

To simulate a receiver that clears the quantization noise floor:

1. Set the **Available power gain** parameter of the mixer to `23.9`.
2. Select **Simulation > Run**.

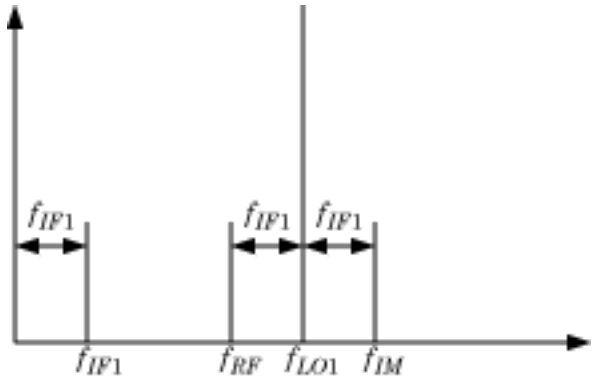


The figure shows that the receiver noise is 16 dB above the quantization noise floor.

Interference

Carrier to Interference Performance of Weaver Receiver

A classic superheterodyne architecture filters images prior to frequency conversion. In contrast, image-reject receivers remove the images at the output without filtering but are sensitive to phase offsets.

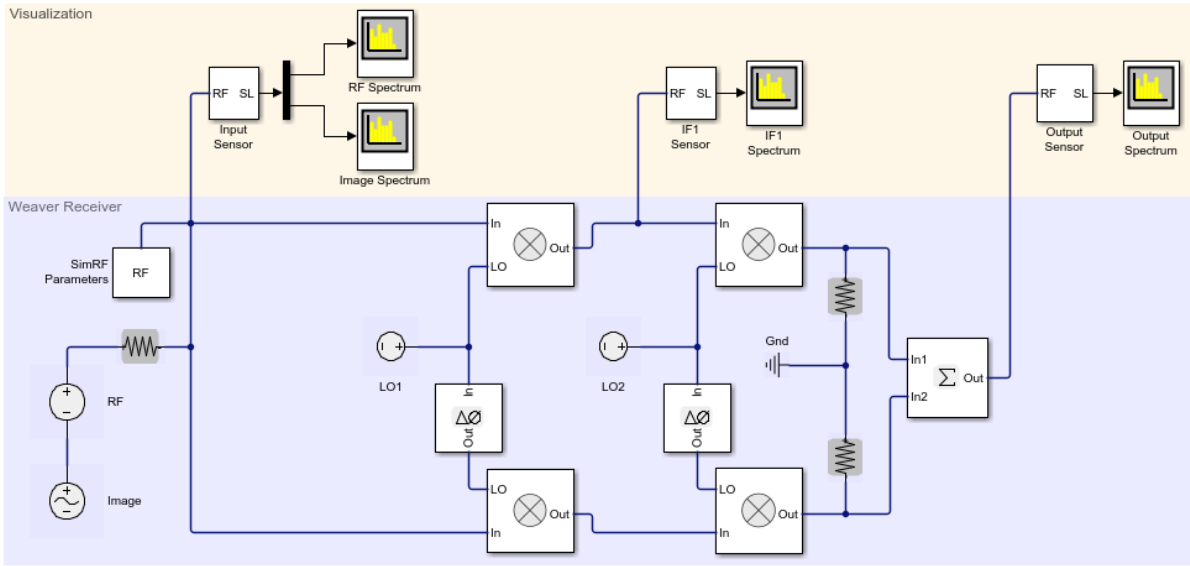


The preceding figure illustrates two input signals at the carriers f_{RF} and f_{IM} that both differ from the LO frequency, f_{LO1} , by an amount f_{IF1}. Mixing translates both input signals down to f_{IF1}. Perfect image rejection in the final stage of the receiver removes the image signal from the output entirely.

Create Model with RF Interference

The model `ex_simrf_ir` performs image rejection with a Weaver architecture. The receiver downconverts the signals f_{RF} and f_{IM} to f_{IF1} and f_{IF2} in two sequential stages.

```
open_system('ex_simrf_ir')
```



Set Up RF Blockset Environment

To maximize performance, the **Fundamental tones** and **Harmonic order** parameters are explicitely set in the `Configuration` block. To create the minimal set of simulation frequencies, the following carrier frequencies are set or created within the model.

- f_{RF}, the RF carrier, equals 100 MHz.
- f_{IM}, the image of the RF carrier relative to f_{LO1}, equals 200 MHz.
- f_{LO1}, the frequency of the LO in the first mixing stage, equals 150 MHz.
- f_{IF1}, the intermediate frequency of the signal after the first mixing stage equals: $f_{LO1} - f_{RF} = f_{IM} - f_{LO1}$ = 50 MHz
- f_{LO2}, the frequency of the LO second mixing stage equals 75 MHz.
- f_{IF2}, the intermediate frequency of the signal after the second mixing stage equals: $f_{LO2} - f_{IF1}$ = 25 MHz.

In this system, every carrier is a multiple of f_{IF2}. The largest carrier, f_{IM}, is the 8th harmonic of f_{IF2}, so setting **Fundamental tones** to f_{IF2} and the **Harmonic order** to 8 ensures that every carrier is in the set of simulation frequencies.

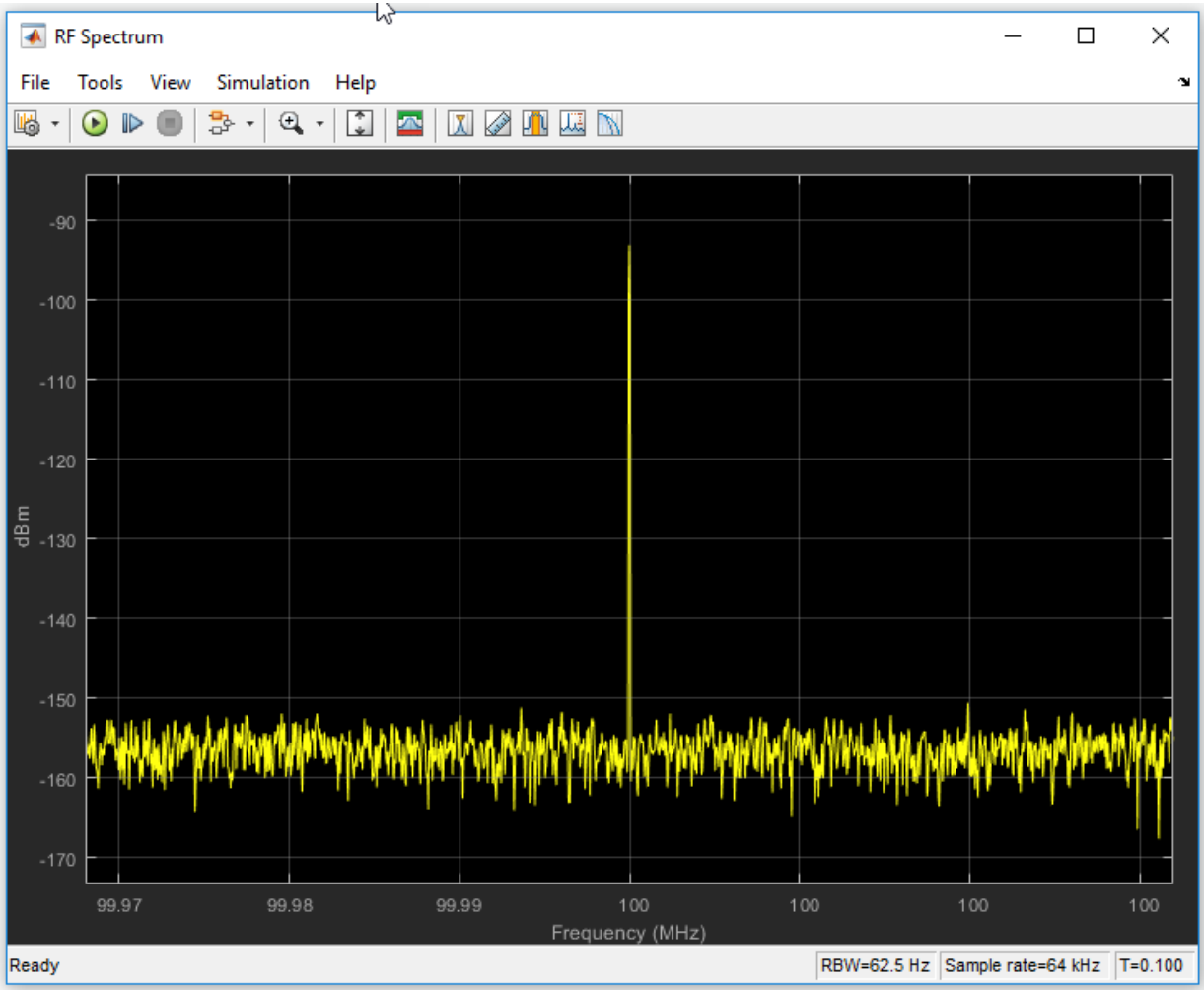
Solver conditions and noise settings are also specified for the `Configuration` block:

- The **Solver type** is set to auto. For more information on choosing solvers, see the reference page for the Configuration block or see Choosing Simulink and Simscape Solvers.
- The **Step size** parameter is set to 1/(mod_freq*64). This setting ensures a simulation bandwidth 64 times greater than the envelope signals in the system.
- The **Simulate noise** box is checked, so the environment includes noise in the simulation.

View Simulation Output

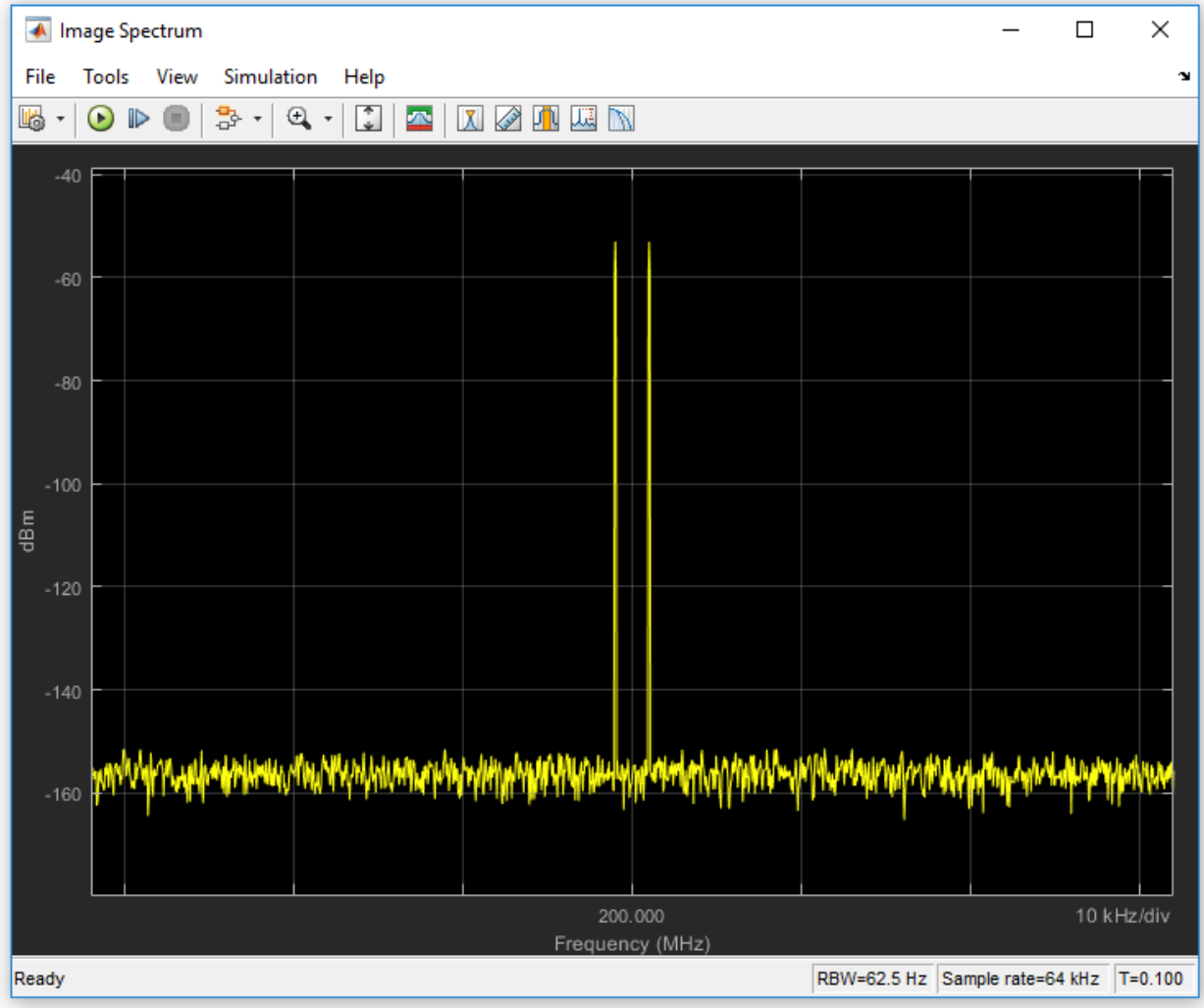
The model uses `Spectrum Analyzer` to generate four plots.

The RF spectrum scope shows the power spectrum of the carrier signal f_{RF}, specified as `carriers.RF` in the **Carrier frequencies** parameter of the Input Sensor `Outport` block.



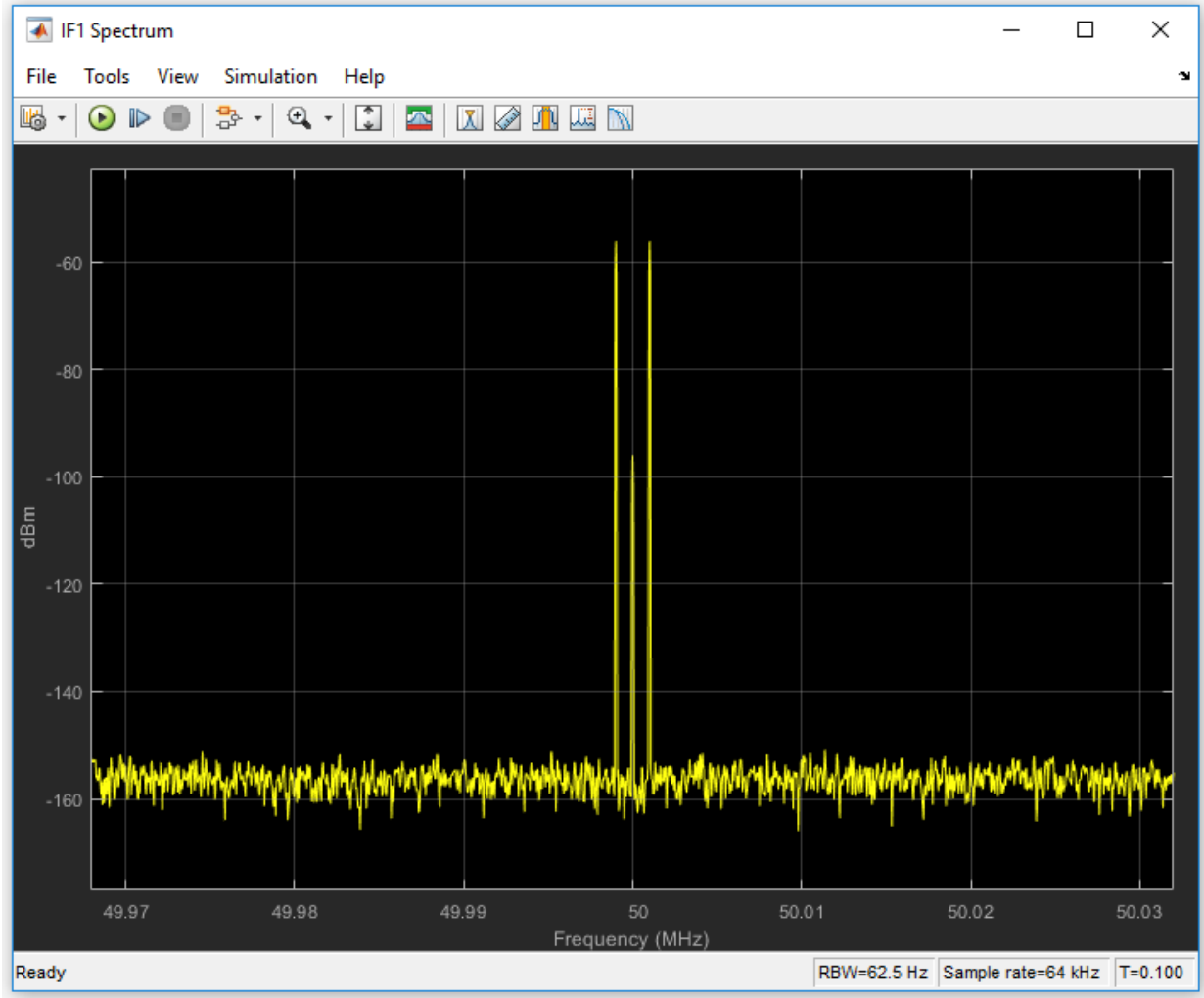
The modulation of the RF carrier is a constant envelope generated by a `Continuous Wave` block which generates a single peak centered at the carrier.

The Image Spectrum scope shows the power spectrum of the image. The signal is recovered from the carrier f_{IM}, specified as `carriers.IM` in the **Carrier frequencies** parameter of the Input Sensor `Outport` block.

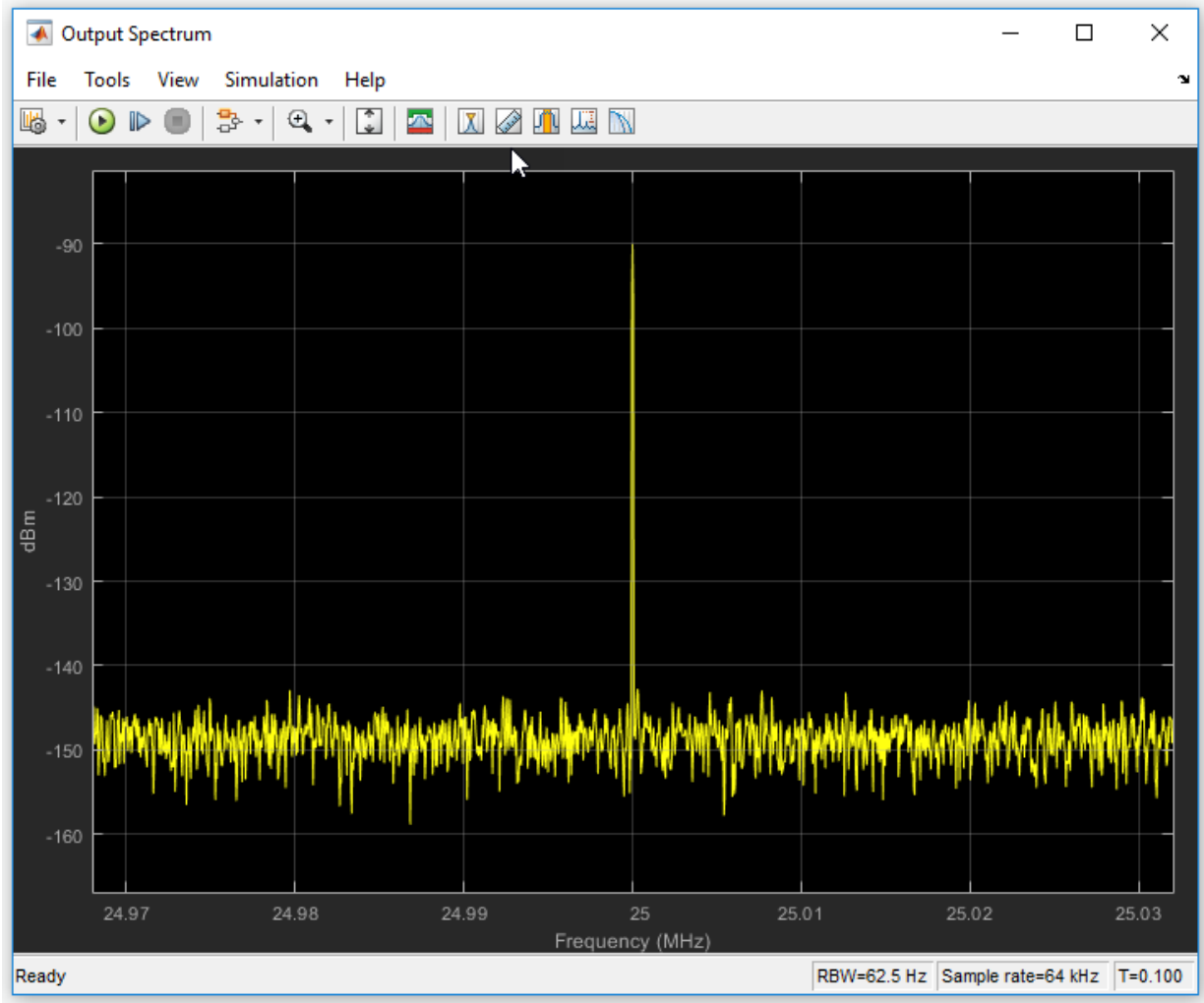


The Image `Sinusoidal Source` block generates a two-tone signal centered at f_{IM}.

The IF1 spectrum scope plot shows a power spectrum centered at the first intermediate frequency, measured between the first and second stages. The sensor outputs the modulation from the carrier f_{IF1}, specified as `carriers.IF1` in the **Carrier frequencies** parameter.



The Output spectrum scope shows the complete effects of the RF system. The sensor outputs the modulation from the carrier f_{IF2}, specified as `carriers.IF2` in the **Carrier frequencies** parameter.



Model RF and Blocker Sources

To model more robust input signals, you can use a RF Blockset `Inport` block to specify a circuit envelope signal generated using blocks from other Simulink™ libraries. For example, see the featured example `Impact of an RF Receiver on Communication System Performance`. This example uses Communications System Toolbox™ to model a

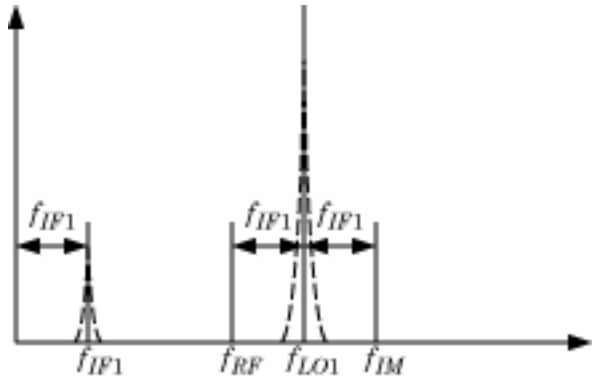
QPSK-modulated waveform of random bits with RF Blockset Inport that brings the signal into the RF Blockset environment.

Simulating LO Phase Offset

The phase shifters have specified Phase shift parameters of 90. Deviation from this value results in a phase offset and causes imperfect image rejection. The featured example `Measuring Image Rejection Ratio in Receivers` analyzes the IRR of the Weaver and Hartley architectures several times, calculating the image rejection ratio (IRR) for several different phase offsets.

Model LO Phase Noise

A mixer transfers local oscillator (LO) phase noise directly to its output.

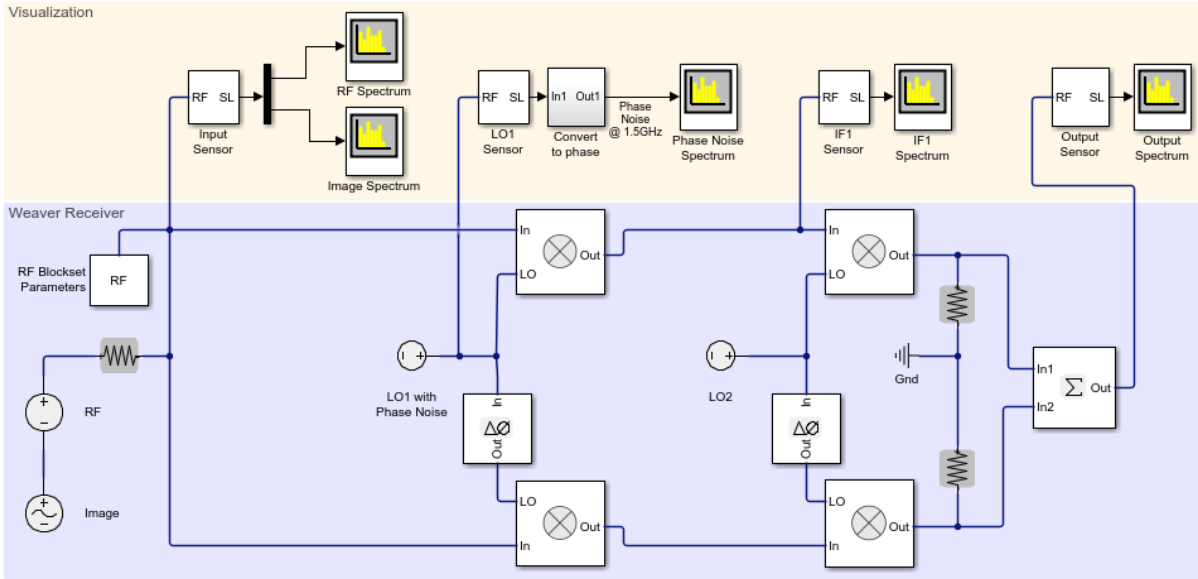


The preceding figure shows the transfer of phase noise from f_{LO1} to f_{IF1} .

Create Model with Phase Noise

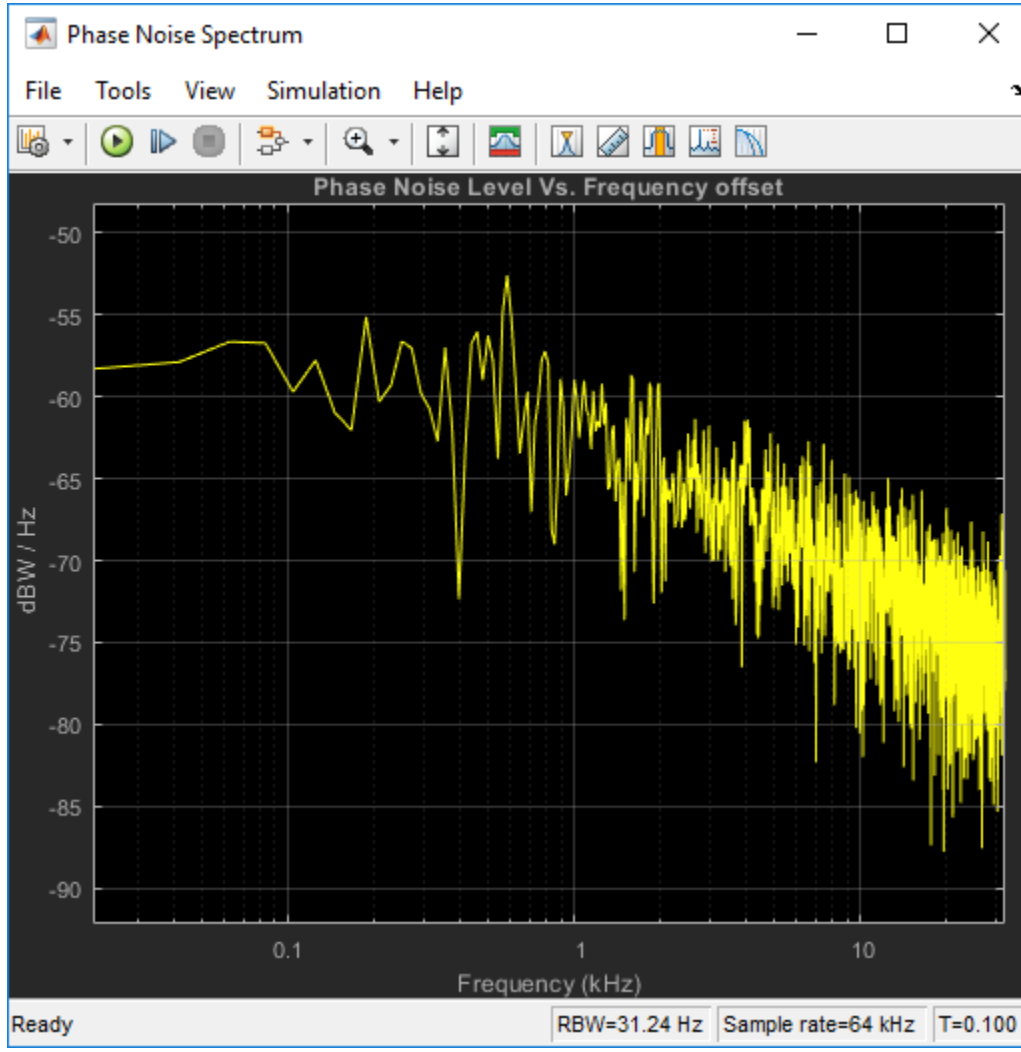
The model `ex_simrf_phase_noise` introduces phase noise into the model from the section `Create a Model with RF Interference`. The first mixing stage downconverts the RF and image to f_{IF}. To open the model:

```
open_system('ex_simrf_phase_noise')
```



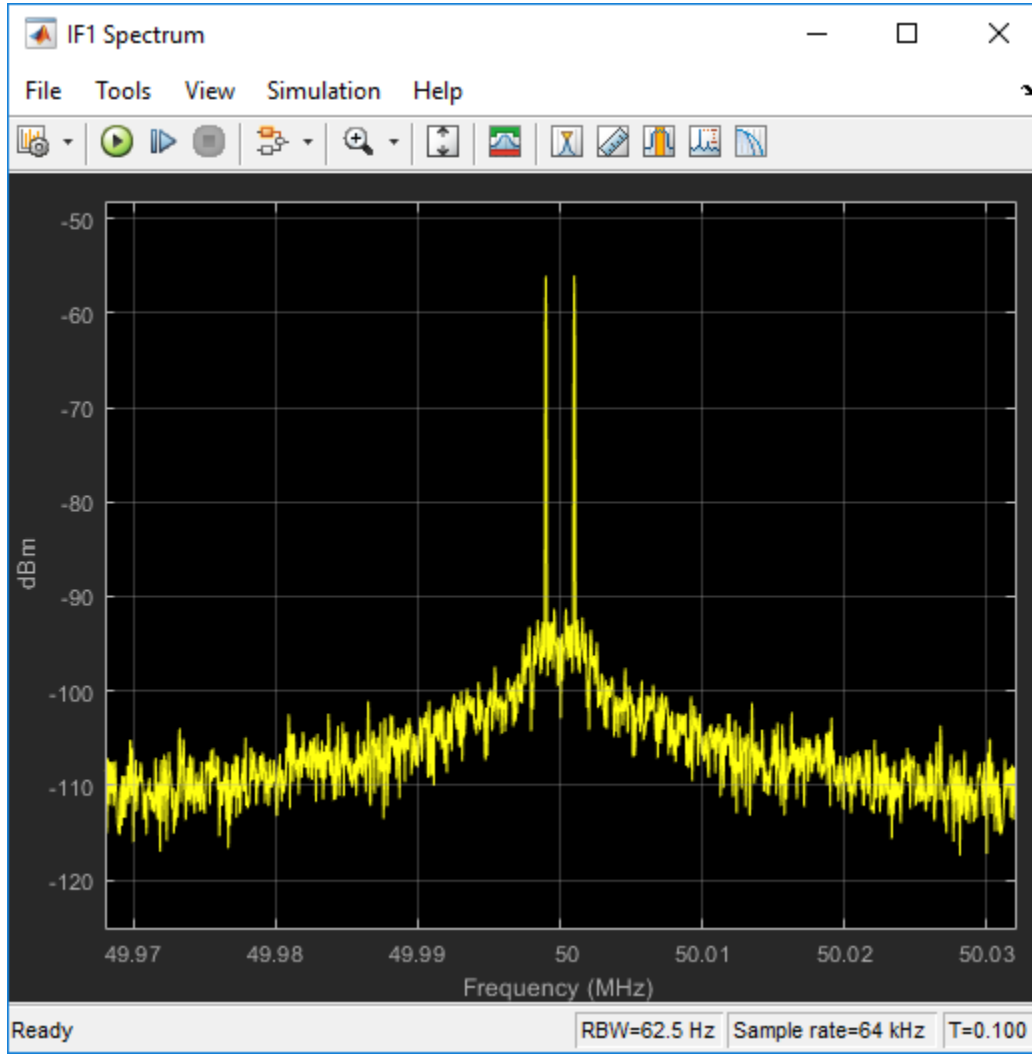
View Simulation Output

The model uses `Spectrum Analyzer` to generate 5 plots.



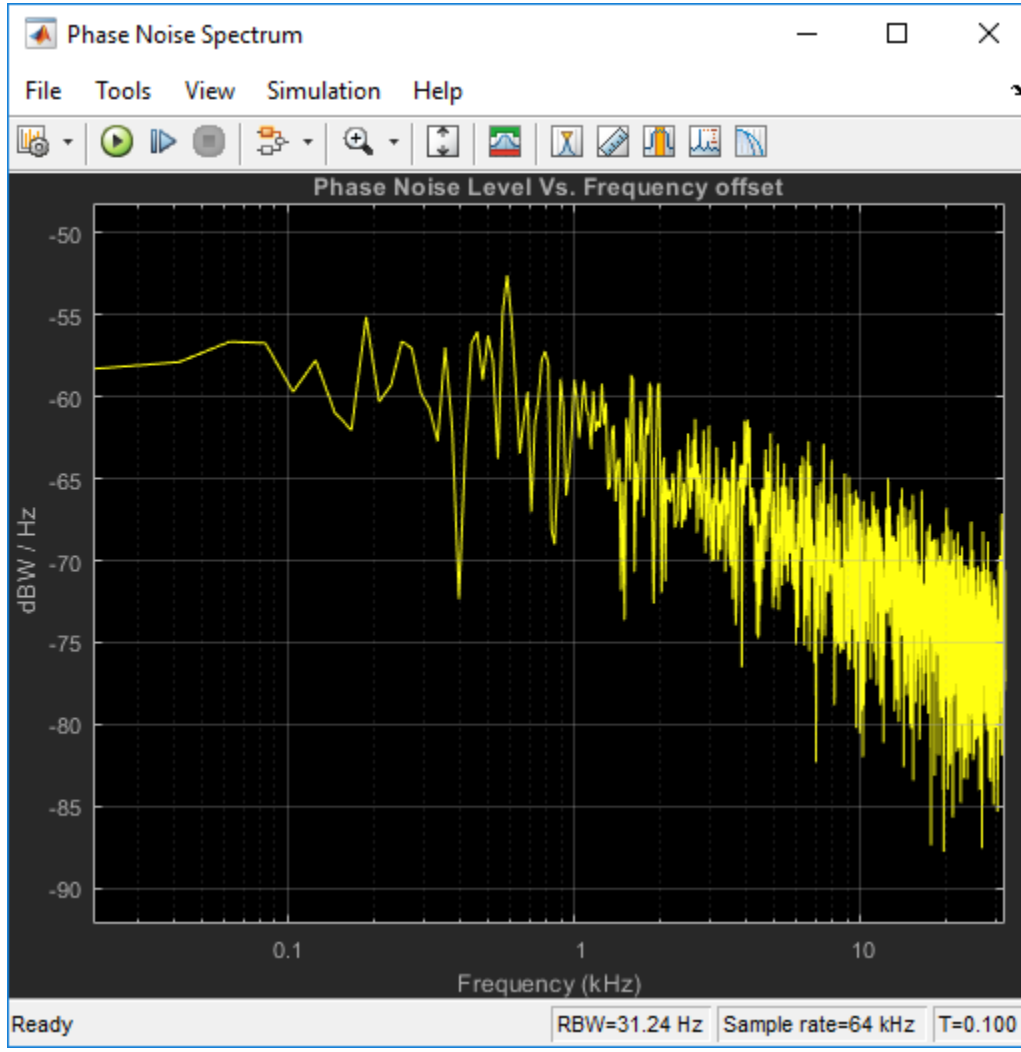
The Phase Noise spectrum scope shows a single-sided power density spectrum measuring the phase noise level at the LO1 source versus frequency offset shown in logarithmic scale.

The IF1 spectrum scope shows a power spectrum centered at the first intermediate frequency, measured between the first and second stages.



The scope shows that the LO phase noise has been transferred to the image. The RF signal on the carrier f_{IF1} is not visible in the figure because its power level is below the phase noise power of the downconverted image signal.

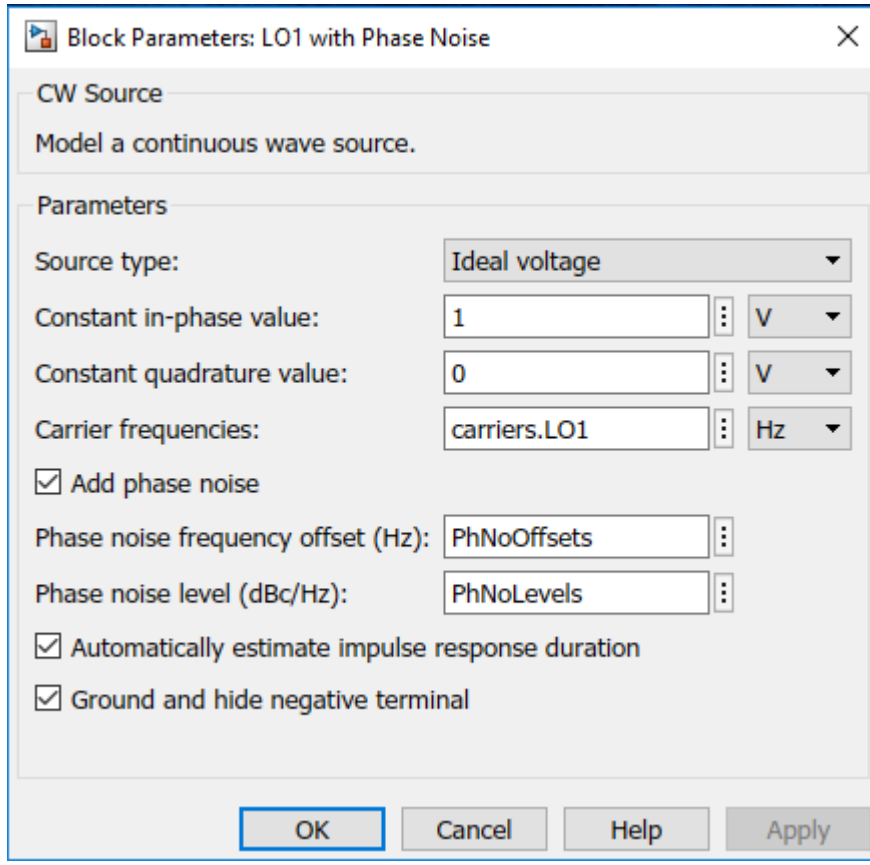
The Output spectrum scope shows the downconverted RF with the images removed.



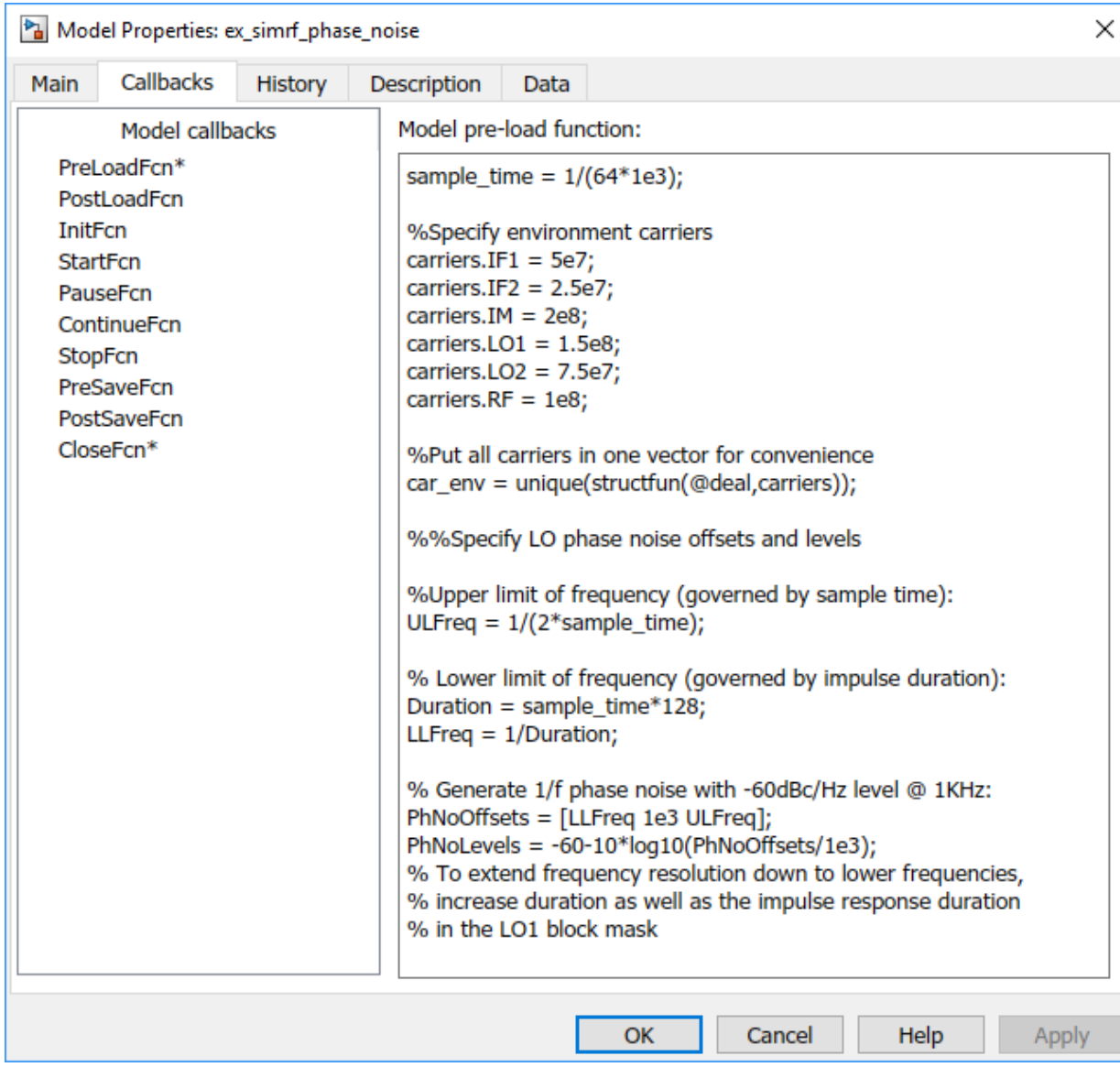
The LO phase noise has been transferred to the receiver output.

Shaping LO Noise Skirt

As seen in the phase noise scope, the added phase noise is pink (1/f) and is specified within the CW source LO1. Specifically, the Add phase noise checkbox is checked in the blocks parameters dialog:



The phase noise frequency offset and phase noise level variables PhNoOffsets and PhNoLevels are defined in the models PreLoadFcn callback, accessible through File > Model Properties > Model Properties:



The upper limit of the offset frequency is governed by the sample time and is limited to the envelope bandwidth of the simulation. The lower limit of the offset frequency is governed by the duration of the impulse response generating the phase noise frequency

profile. Increasing the duration length in time steps from 128 to 256 will double the frequency resolution and allow simulation of the 1/f profile down to 250Hz (as opposed to the current lower limit of 500Hz).

3

Intermodulation Distortion

Model a Direct Conversion Receiver

In this section...

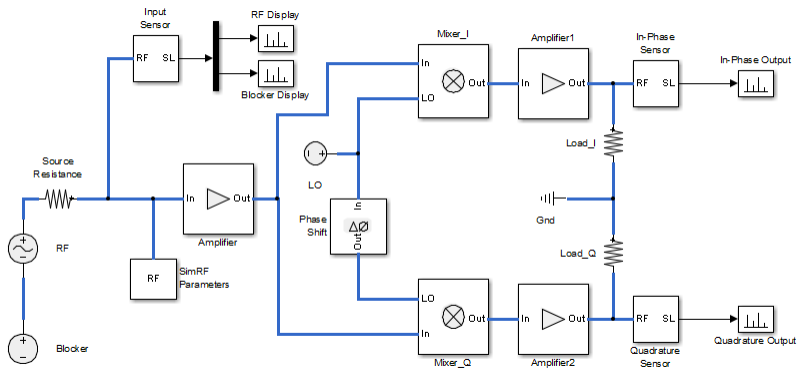
Direct-conversion receivers are sensitive to second-order intermodulation products because they transfer the RF signal directly to baseband.

Create a Direct Conversion Receiver Model

The model `ex_simrf_dc` models a direct-conversion receiver within the RF Blockset environment. The RF system consists of a low-noise amplification (LNA) stage, a direct-conversion stage, and a final amplification stage.

To open this model, at MATLAB command line, enter:

```
addpath(fullfile(docroot,'toolbox','simrf','examples'))
ex_simrf_dc
```



To run the model:

1. Open the model by clicking the link or by typing the model name at the command prompt.

2 Select **Simulation** > **Run**.

Set Up the RF Blockset Environment

The model runs according to the following environment settings:

- In the Configuration dialog box, the **Fundamental tones** parameter specifies the carriers in the RF Blockset environment:
  - $f_{RF} = f_{LO}$, the carrier of the RF and the local oscillator.
  - f_{BL}, the blocker carrier

  The RF Blockset environment always simulates the 0- Hz carrier, regardless of whether the RF Blockset Parameters block specifies it.
- In the Solver Configuration dialog box, the **Use local solver** box is selected. This setting causes the RF Blockset environment to simulate with a local solver with the following settings:
  - **Solver type** is `Trapezoidal rule`.
  - **Sample time** is `sample_time`, defined as `1.25e-4` in the model initialization function.

  Since the model uses a local solver, the global solver settings do not affect the simulation within the RF Blockset environment. For more information on global and local solvers, see Choosing Simulink and Simscape Solvers (Simscape).

To maximize performance, the **Fundamental tones** and **Harmonic order** parameters specify the simulation frequencies explicitly in the Configuration block:

- $f_{RF} = f_{LO}$, the carrier of the RF and the local oscillator, appears as a fundamental tone.
- f_{BL}, the blocker carrier, appears as a fundamental tone.
- A carrier of 0 Hz, representing the passband signal, is included in the set of first-order harmonics of both fundamental tones. Therefore, setting **Harmonic order** to `1` is sufficient to ensure that this frequency appears in the simulation frequencies. This minimal value for the harmonic order ensures a minimum of simulation frequencies.

Solver conditions and noise settings are also specified for the Configuration block:

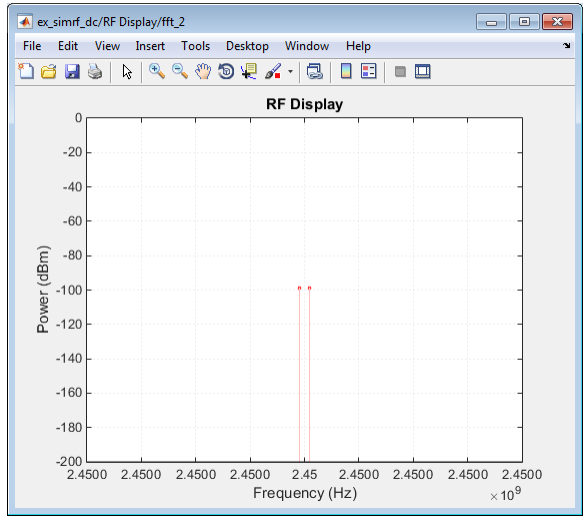
- The **Solver type** is set to `auto`. For more information on choosing solvers, see the reference page for the Configuration block or see Choosing Simulink and Simscape Solvers (Simscape).

- The **Sample time** parameter is set to `sample_time`, which is equal to `1/(mod_freq*64)`. This setting ensures a simulation bandwidth 64 times greater than the envelope signals in the system.
- The **Simulate noise** box is checked, so the environment includes noise parameters during simulation.

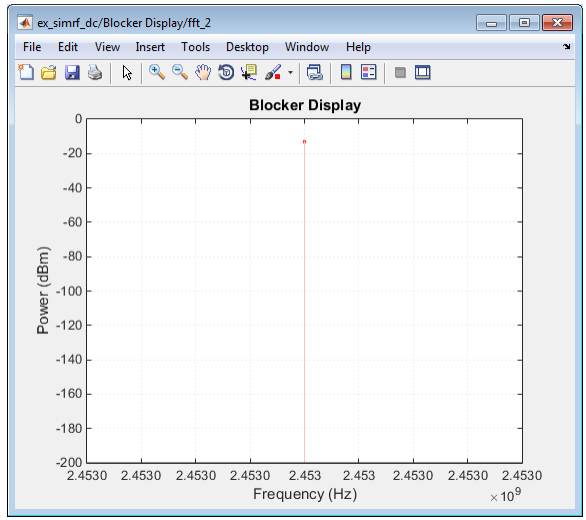
View Simulation Output

The model uses subsystems with a MATLAB Coder implementation of a fast Fourier transform (FFT) to generate four plots:

- The RF Display plot shows the power level of the RF signal.

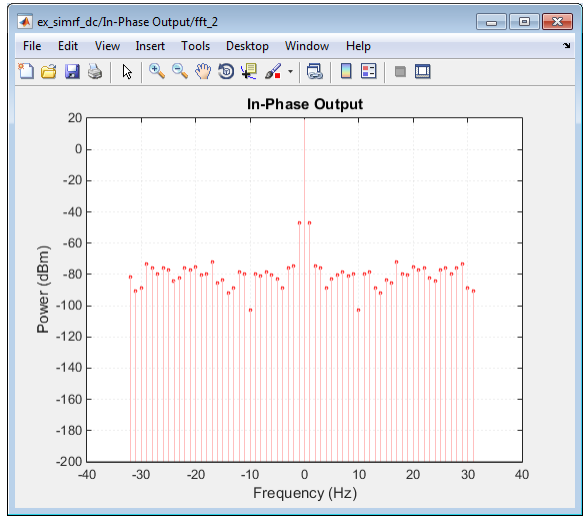


  The power level of the RF is about 100 dBm.
- The Blocker Display plot shows the power spectrum centered at the carrier f_{BL}.



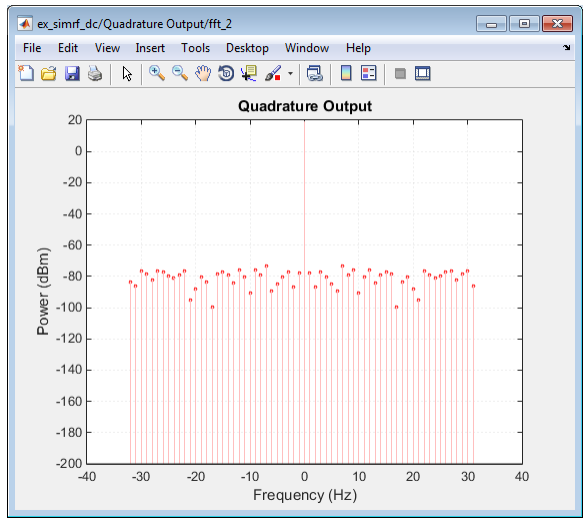
The power level of the blocker is about 90 dB higher than the signal power of the RF.

- The In-Phase Output plot shows the power spectrum of the in-phase signal at baseband.



In the figure, DC power is a direct result of the blocker and the IP2 in the mixers.

- The Quadrature Output plot shows the power spectrum of the quadrature signal at baseband.



If you have DSP System Toolbox software installed, you can replace the MATLAB Coder subsystems with a Spectrum Analyzer block.

Modeling IMD in System-Level Components

The **IP2** and **IP3** parameters specify the second- and third-order intercept points of Amplifier and Mixer blocks:

- The amplifiers have infinite **IP2** and **IP3**, so the amplifiers are linear.
- **IP2** of the mixer is 15 dB

Amplifier and Mixer components have specified gains and noise figures:

- The gain and noise figure in the LNA stage are 25 dB and 6 dB, respectively.
- The gain and noise figure in the mixing stage are 10 dB and 10 dB. The **Input impedance (ohms)** parameters of the two mixers are both `100`, which sum in parallel to a resistance of 50 Ω to match the output impedance of the LNA.
- The gain and noise figure in the final amplification stage are 20 dB and 15 dB, respectively.

To calculate RF system noise figure, use the Friis equation:

$$F_{sys} = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2} + \ldots + \frac{F_n - 1}{G_1 G_2 \ldots G_{n-1}}$$

where F_i and G_i are the noise factor and gain of the ith stage.

Examining DC Impairments

In addition to intermodulation distortion from IP2, direct-conversion receivers are subject to additional DC impairments. For example, coupling between mixer input and local oscillator (LO) ports causes self-mixing of the LO. For more information, see the featured example “Executable Specification of a Direct Conversion Receiver”

Intermodulation Distortion in Amplifiers and Mixers

Intermodulation distortion or IMD is a measure of linearity of amplifiers, mixers, gain blocks, and other RF components. IMD is the result of two or more signals interacting in a non-linear device such as an amplifier or a mixer to produce additional unwanted signals.

Intermodulation Distortion in Amplifiers

Intermodulation distortion or IMD is an undesirable quality in power amplifiers.

Noise in RF Systems

In this section...

Noise in an RF system is generated internally by active components in the system or introduced externally like channel interference or antenna.

White and Colored Noise

White noise: Noise with a flat frequency spectrum is called `white noise`. White noise has equal power across all frequencies of the system band width.

Colored noise: Noise with power that varies according to frequencies in an RF system bandwidth is called `colored noise`.

To simulate white or colored noise in RF Blockset, use the Noise block.

Thermal Noise

Thermal noise is the most common noise introduced in an RF system. This noise is generated internally by active components in the system or externally due to channel interference or antenna. Thermal noise is also known as Johnson or Nyquist noise. The equation for thermal noise is:

$$P_N = k_B T B$$

- k_B is Boltzmann's constant, equal to 1.38065×10^{-23} J/K.
- T is the noise temperature, specified as 293.15 K in this example.
- R_s is the noise source impedance, specified as 50 Ω in this example to agree with the resistance value of the Resistor block labeled `R1`.
- B is the bandwidth.

At room temperature, the thermal noise generated by system with a band width of 1 Hz is `-174 dBm`.

To generate thermal noise in a RF Blockset system use Resistor and Configuration block. You can also generate thermal noise using the S-Parameters block if the S-parameter is passive.

Thermal noise floor in RF Blockset is defined by the equation:

$$P_{noise} = 4k_B T R_s \Delta f$$

where:

- k_B is Boltzmann's constant, equal to 1.38065×10^{-23} J/K.
- T is the noise temperature, specified as 293.15 K in this example.
- R_s is the noise source impedance, specified as 50 Ω in this example to agree with the resistance value of the Resistor block labeled `R1`.
- Δf is the noise bandwidth.

Phase Noise

Phase noise is a short-term fluctuation in the phase of an oscillator signal. This noise introduces uncertainty in the detection of digitally modulated signals. Phase noise is defined as the ratio of power ins one-phase modulation sideband to the total signal power per unit bandwidth. It is expressed in decibels relative to the carrier power per hertz of bandwidth (dBc/Hz). To know how to model LO phase noise in an oscillator see, “Model LO Phase Noise” on page 2-10.

Noise Figure

Noise figure value determines the degradation of signal to noise ratio of a signal as it goes through the network. Noise figure is defined by the equation:

$$N_f = \frac{\frac{Signal}{Noise}\text{at the input}}{\frac{Signal}{Noise}\text{at the output}}$$

Excessive noise figure in the system causes the noise to overwhelm the signal, making the signal unrecoverable. Noise figure is a function of frequency but it is independent of the bandwidth of the system. Noise figure is expressed in dB.

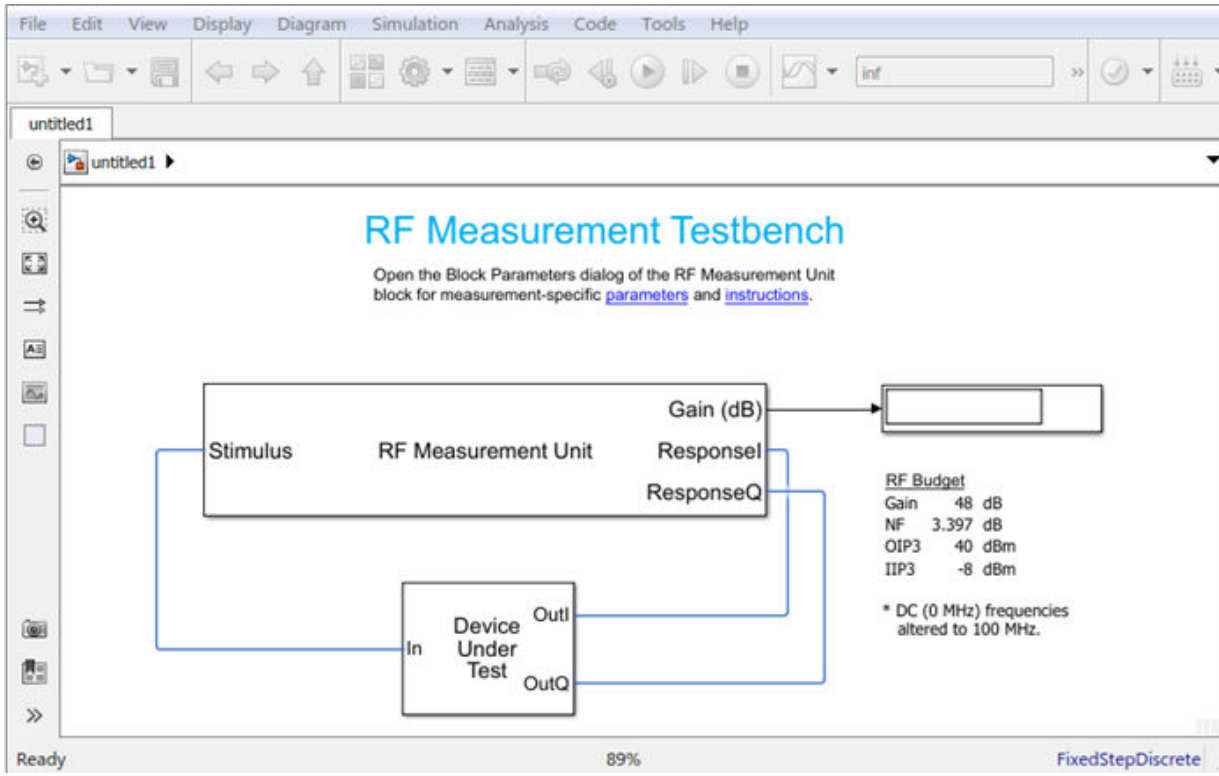
In RF Blockset, you can specify noise figure for an RF system using the Amplifier or Mixer blocks.

Testbenches

Use RF Measurement Testbench for RF-to-IQ Converter

In this section...

Use the RF Measurement Testbench to measure various quantities of an RF-to-IQ converter. Measurable quantities include cumulative gain, noise figure, and nonlinearity (IP3) values. To open the testbench and measure the quantities, use the **RF Budget Analyzer** app to create an RF-to-IQ converter and then click `Export` > `Measurement testbench`.

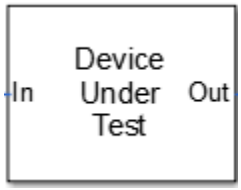


The testbench has two subsystems:

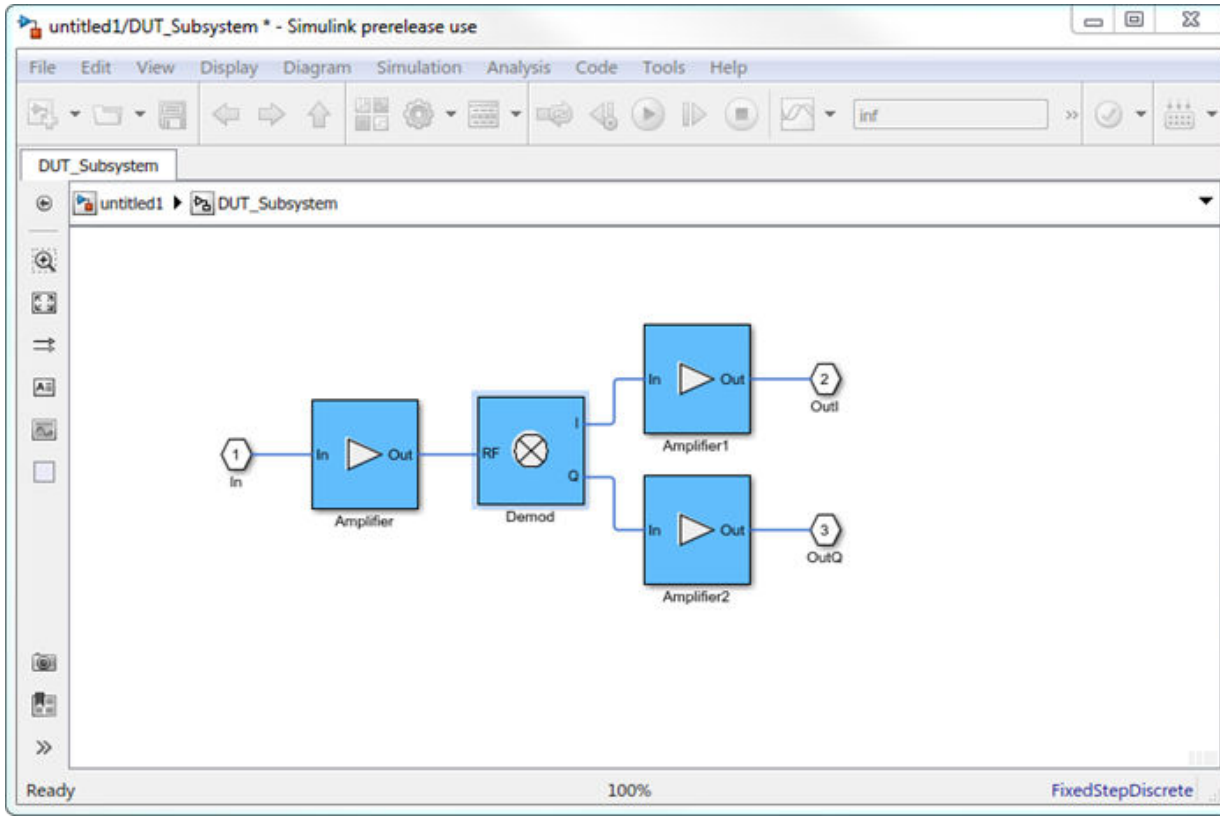
- `RF Measurement Unit`
- `Device Under Test`

The testbench display shows the measured output values of the gain, NF (noise figure), IP3 (third-order intercept), and other quantities etc.

Device Under Test

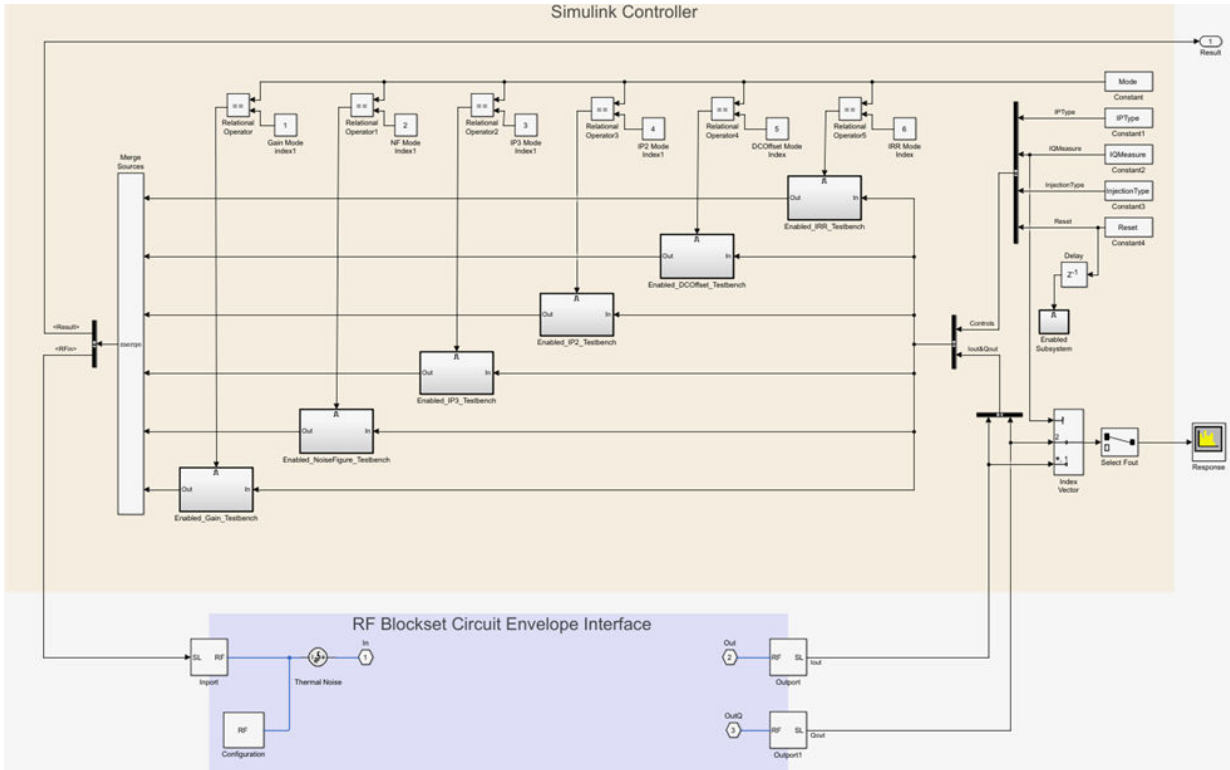


The Device Under Test subsystem contains the RF system exported from the app.

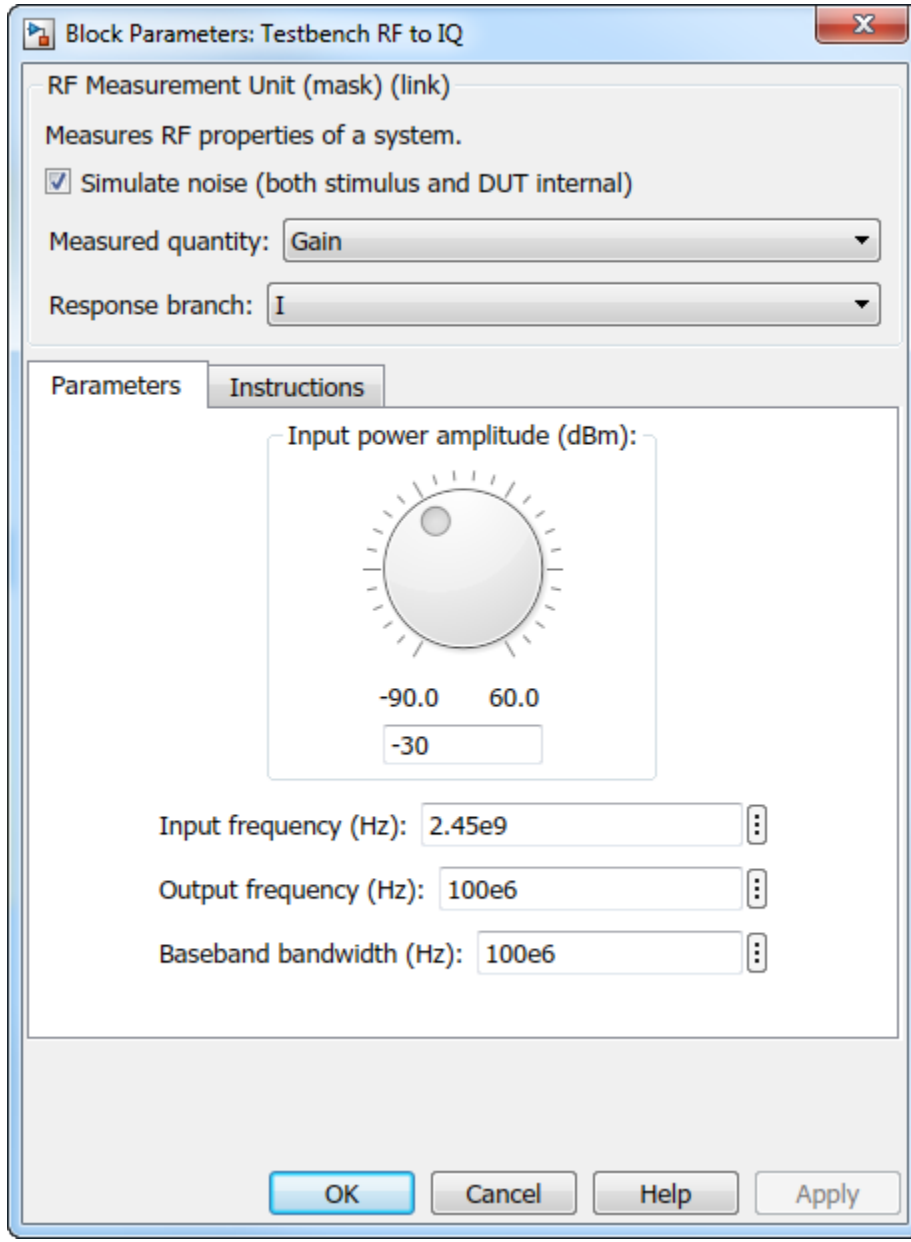


RF Measurement Unit

The RF Measurement Unit subsystem consists of a Simulink Controller and RF Blockset Circuit Envelope interface. The RF Blockset interface is used as input and output from the DUT.


Simulink Controller
Result
Mode
Constant
Relational Operator
Gain Mode Index1
Relational Operator1
NF Mode Index1
Relational Operator2
IP3 Mode Index1
Relational Operator3
IP2 Mode Index1
Relational Operator4
DCOffset Mode Index
Relational Operator5
IRR Mode Index
IPType
IPType
Constant1
IQMeasure
IQMeasure
Constant2
InjectionType
InjectionType
Constant3
Reset
Reset
Constant4
Delay
Merge Sources
Enabled_IRR_Testbench
Enabled_DCOffset_Testbench
Enabled_IP2_Testbench
Enabled_IP3_Testbench
Enabled_NoiseFigure_Testbench
Enabled_Gain_Testbench
Controls
Iout&Qout
Enabled Subsystem
Index Vector
Select Fout
Response
RF Blockset Circuit Envelope Interface
Inport
Thermal Noise
Configuration
Outport
Outport1

RF Measurement Unit Parameters



- **Simulate noise (both stimulus and DUT)** — Select this check box to enable noise modeling in the stimulus signal entering the DUT and inside the DUT.
- **Measured quantity** — Choose the quantity you want to measure:

  - `Gain` – Measure the transducer gain of the converter, assuming a load of 50 ohm. If you choose only `I` or only `Q` from **Response branch**, you see only half the value of the measured gain.
  - `NF` – Measure the noise figure value at the output of the converter.
  - `IP3` – Measure the output or input third-order intercept (IP3).
  - `IP2` – Measure the output or input second-order intercept (IP2).
  - `DC Offset` – Measure the DC level interference centered on the desired signal due to LO leakage mixing with input signal.
  - `Image Rejection Ratio` – Measure the image rejection ratio required to cancel the effect of images at the RF input signal.

  By default, the testbench measures the `Gain`. The contents in the **Instructions** tab changes according to the **Measured quantity** value.
- **IP Type** — Choose the type of intercept points (IP) to measure: `Output referred` or `Input referred`.

  By default, the testbench measures `Output referred`. This option is available when you set the **Measured quantity** to `IP2` or `IP3`.
- **Injection Type** – Choose the local oscillator (LO) injection for image rejection ratio: `Low-side` or `High-side`.

  By default, the testbench measures the `Low-side`. This option is available when you set the **Measured quantity** to `Image Rejection Ratio`.
- **Response branch** — Choose the output branch you want to measure from:

  - `I only(Q=0)` – Output signal measured at the in-phase branch.
  - `Q only(I=0)` – Output signal measured at the quadrature branch.

  This option is not available when you set the **Measured quantity** to `Image Rejection Ratio`.

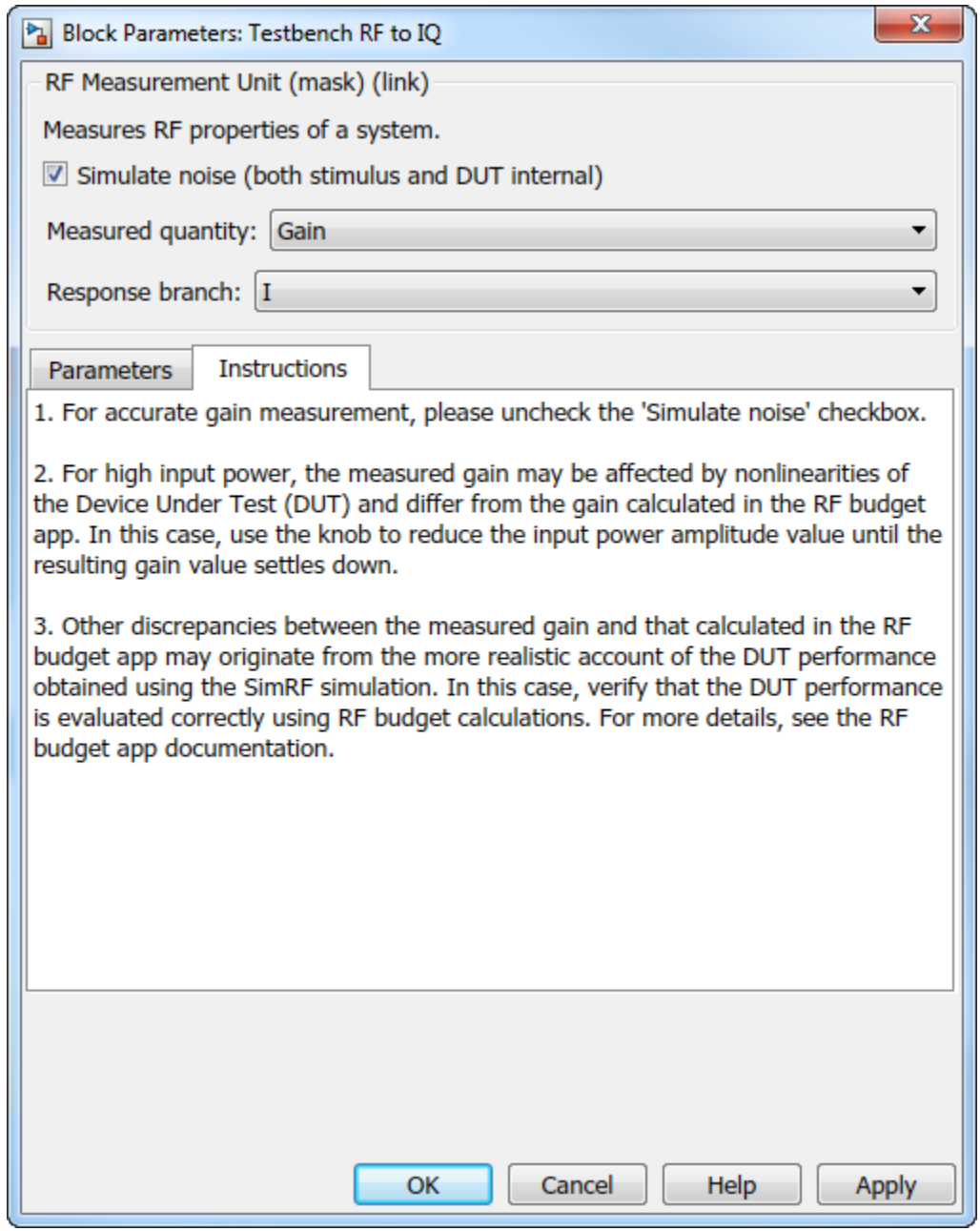
Parameters Tab

- **Input power amplitude (dBm)** – Available input power to the DUT. You can change the input power by manually specifying a value or by turning the knob. When

measuring DC Offset, this input field is **Input RMS voltage (dBmV)**, because the Offset is measured in voltage units. The specified voltage represents the voltage falling on the input ports of the DUT.

- **Input frequency (Hz)** – Carrier frequency fed at the RF input of the DUT.
- **Output frequency (Hz)** – Output frequency to measure the I and Q outputs of the DUT. By default, this frequency is one bandwidth above DC, to allow meaningful measurement.
- **Baseband bandwidth (Hz)** – Bandwidth of the input signal.
- **Ratio of test tone frequency to baseband bandwidth** – Position of the test tones used for IP3 measurements. By default, the value is `1/8`.

Instructions Tab



Instructions for Gain Measurement

- Clear **Simulate noise (both stimulus and DUT)** for accurate gain measurement. Select the check box to account for the noise.
- Change the **Input power amplitude (dBm)** or turn the knob to reduce the input power amplitude. For high input power, nonlinearities in the DUT can affect the gain measurements.

Instructions for NF Measurement

- The testbench measures the spot NF calculated. This calculation assumes a frequency-independent system within a given bandwidth. To simulate a frequency-independent system and calculate the correct NF value, reduce the baseband bandwidth until this condition is fulfilled. In common RF systems, the bandwidth is reduced below 1 kHz for NF testing.
- Change **Input power amplitude (dBm)** or turn the knob to reduce or increase the input power amplitude. For high input power, nonlinearities in the DUT can affect the NF measurements. For low input power, the signal is too close or below the noise floor of the system. As a result, the NF fails to converge.

Instructions for IP3 and IP2 Measurement

- Clear **Simulate noise (both stimulus and DUT)** for accurate IP3 and IP2 measurement.
- Change **Input power amplitude (dBm)** or turn the knob to reduce the input power amplitude. For high input power, higher-order nonlinearities in the DUT can affect the OIP3 and IIP3 measurements.

Instructions for DC Offset Measurement

- Clear **Simulate noise (both stimulus and DUT)** for accurate DC offset measurement.
- Correct calculation of the DC offset assumes a frequency-independent system in the frequencies surrounding the test tones. Reduce the frequency separation between the test tones or reduce the baseband bandwidth until this condition is fulfilled. In common RF systems, the bandwidth is reduced below `1 KHz` for DC offset testing.
- . Change **Input RMS voltage amplitude (dBmV)** or turn the knob to reduce the input RMS voltage amplitude. For high input RMS voltage, higher-order nonlinearities in the DUT can affect the DC offset measurements

Instructions for Image Rejection Ratio

- Clear **Simulate noise (both stimulus and DUT)** for accurate OIP3 and IIP3 measurement.
- Correct calculation of the image rejection ratio (IRR) assumes a frequency-independent system in the frequencies surrounding the test tones. Reduce the frequency separation between the test tones or reduce the baseband bandwidth until this condition is fulfilled. In common RF systems, the bandwidth is reduced below 1 KHz for IRR testing.
- . Change **Input power amplitude (dBm)** or turn the knob to reduce the input power amplitude. For high input power, higher-order nonlinearities in the DUT can affect the image rejection ratio measurement.

For all measurements using the testbench, you cannot correct result discrepancies using the **RF Budget Analyzer** app. The RF Blockset testbench provides true RF circuit simulation that incorporates RF phenomena including saturation and interaction between multiple tones and harmonics in nonlinear devices. These RF phenomena are not yet incorporated in **RF Budget Analyzer**, leading to some differences in the values between the testbench and the app.

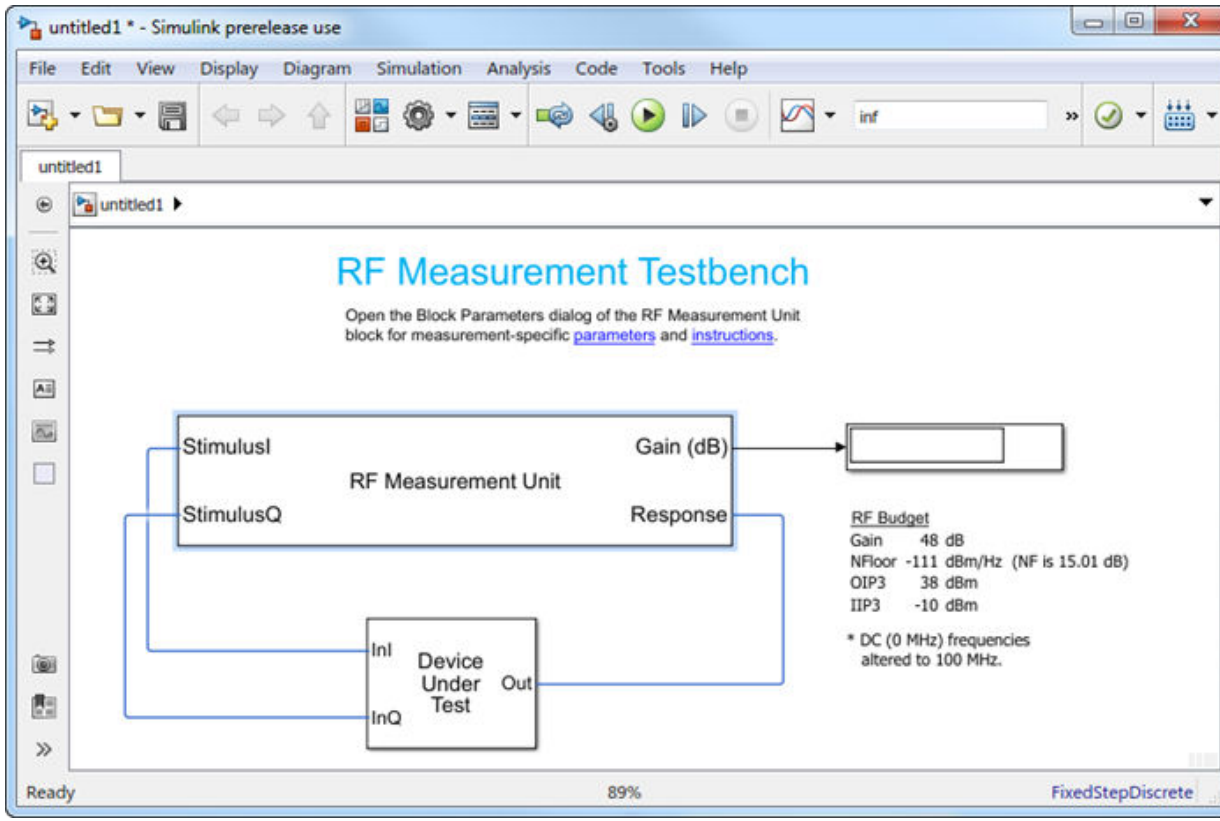
See Also

RF Budget Analyzer

Using RF Measurement Testbench for IQ-to-RF Converter

In this section...

Use the RF Measurement Testbench to measure various quantities of an IQ-to-RF converter system. Measurable quantities include cumulative gain, noise figure, and nonlinearity (IP3) values. To open the testbench and measure the quantities, use the **RF Budget Analyzer** app to create an RF system and then click `Export` > `Measurement testbench`.

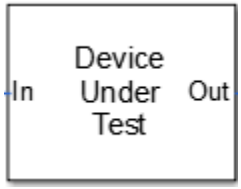


The testbench has two subsystems:

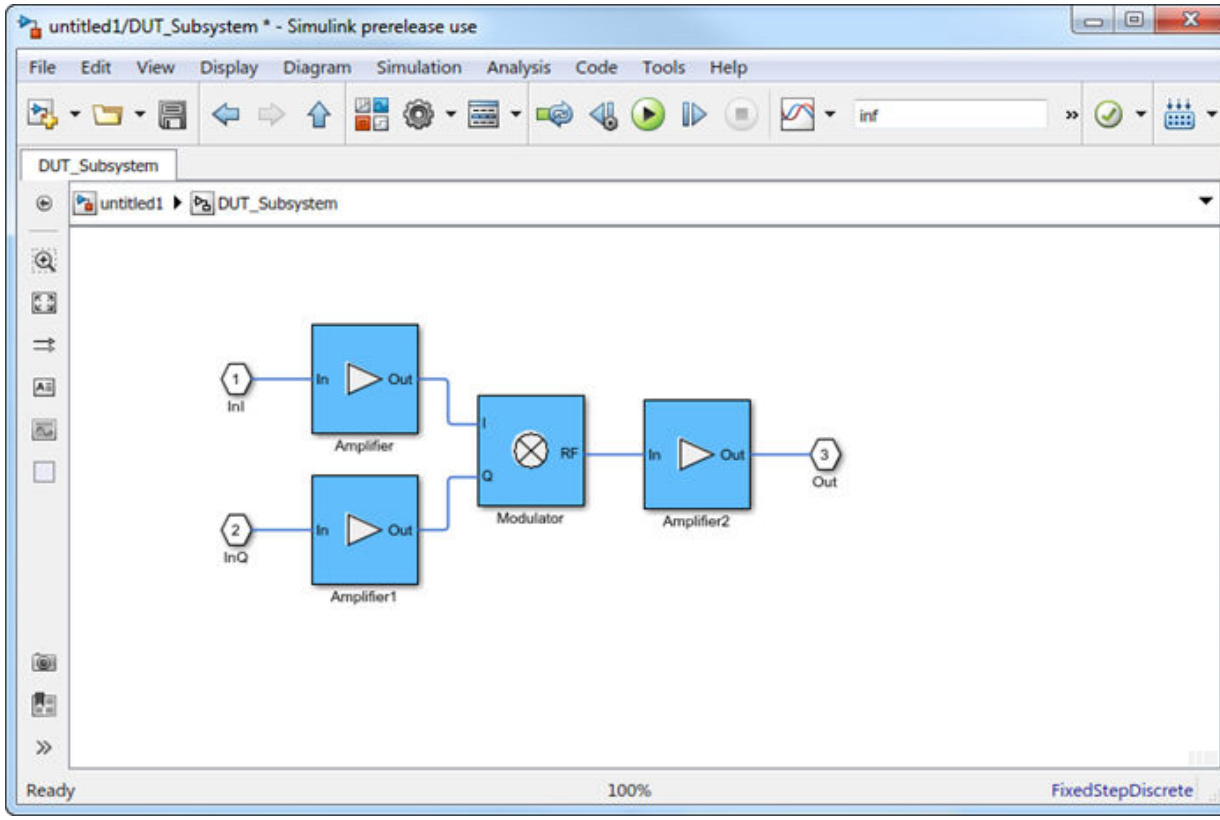
- `RF Measurement Unit`
- `Device Under Test`

The testbench display shows the measured output values of the gain, NF (noise figure), IP3 (third-order intercept), and other quantities.

Device Under Test

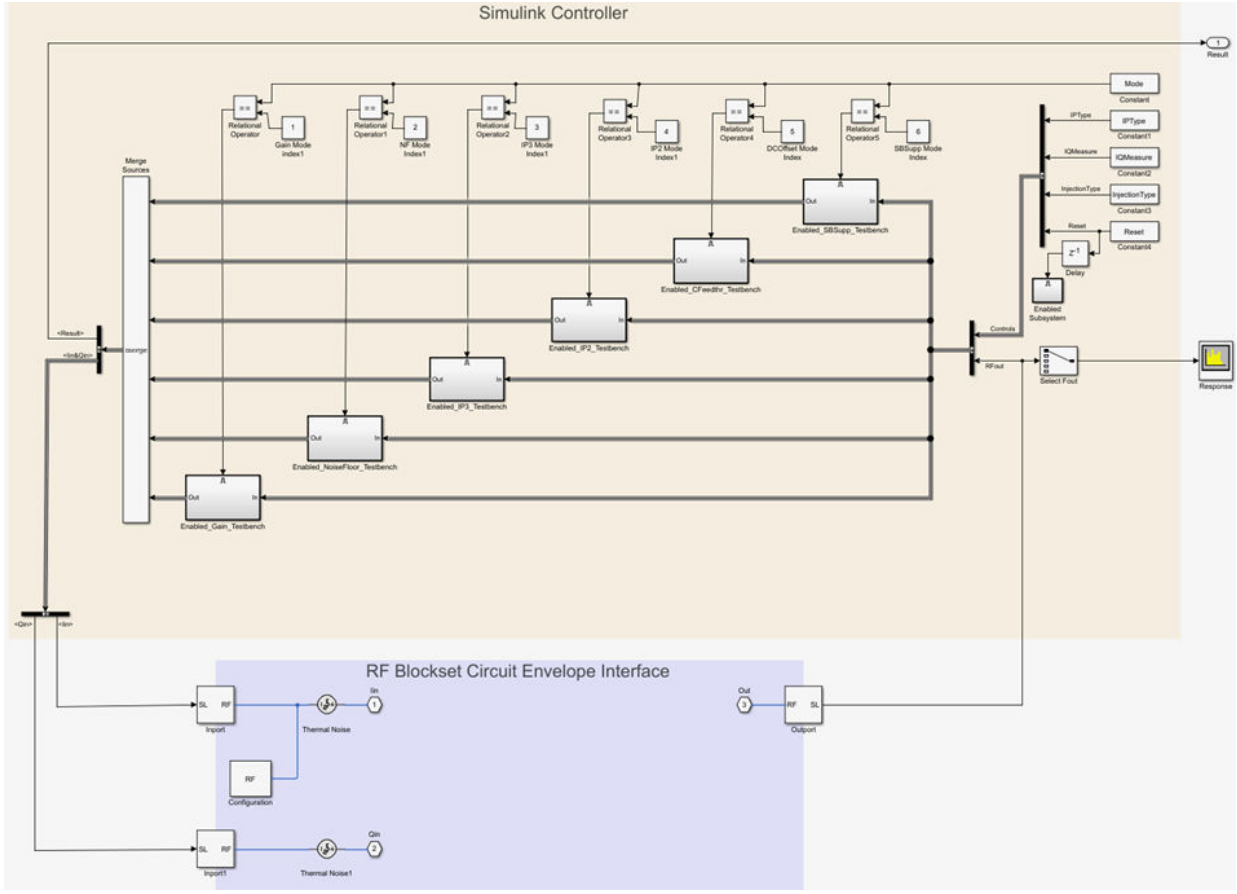


The Device Under Test subsystem contains the RF system exported from the app.

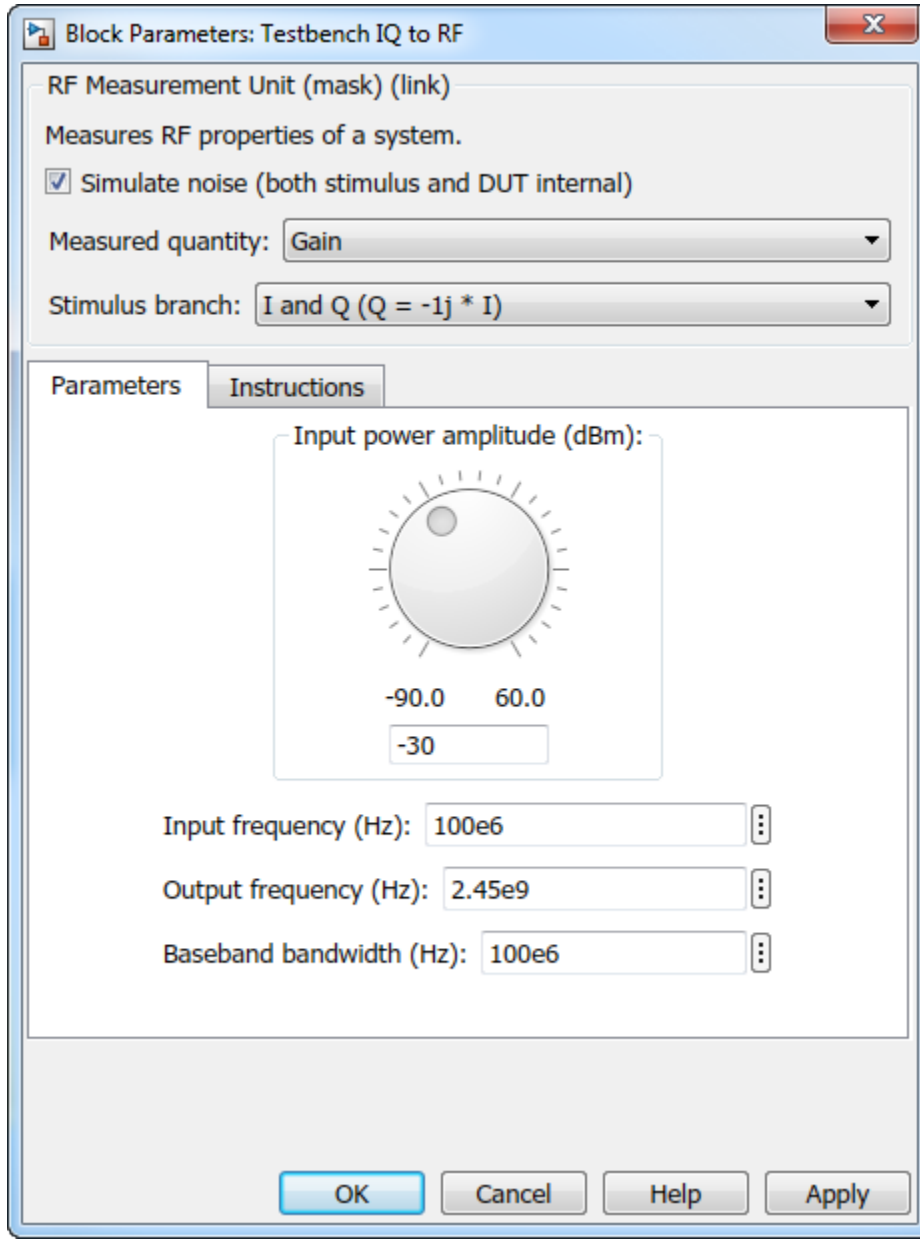


RF Measurement Unit

The RF Measurement Unit subsystem consists of a Simulink Controller and RF Blockset Circuit Envelope interface. The RF Blockset interface is used as input and output from the DUT.


Simulink Controller
Result
Mode
Constant
Relational Operator
Gain Mode Index1
Relational Operator1
NF Mode Index1
Relational Operator2
IP3 Mode Index1
Relational Operator3
IP2 Mode Index1
Relational Operator4
DCOffset Mode Index
Relational Operator5
SBSupp Mode Index
IPType
Constant1
IQMeasure
Constant2
InjectionType
Constant3
Reset
Constant4
Delay
Enabled Subsystem
Merge Sources
Enabled_SBSupp_Testbench
Enabled_CFeedthr_Testbench
Enabled_IP2_Testbench
Enabled_IP3_Testbench
Enabled_NoiseFloor_Testbench
Enabled_Gain_Testbench
Controls
RFout
Select Fout
Response
RF Blockset Circuit Envelope Interface
Inport
Thermal Noise
Configuration
Inport1
Thermal Noise1
Outport

RF Measurement Unit Parameters



- **Simulate noise (both stimulus and DUT)** — Select this check box to enable noise modeling in the stimulus signal entering the DUT and inside the DUT.
- **Measured quantity** — Choose the quantity you want to measure:

  - `Gain` – Measure the transducer gain of the converter. If you choose only `I` or only `Q` from **Stimulus branch**, you only see half the value of the measured gain.
  - `Noise Floor` – Measure the noise floor value of the converter.
  - `IP3` – Measure the output or input third-order intercept (IP3).
  - `IP2` – Measure the output or input second-order intercept (IP2).
  - `Carrier Feedthrough` – Measure the leakage of carrier tone into the RF spectrum due to imbalances in the in-phase and quadrature phase inputs.
  - `Sideband Suppression` – Measure the sideband suppression required for the ideal cancellation of image signals around the RF output signal.

  By default, the testbench measures `Gain`. The contents in the **Instructions** tab changes according to the **Measured quantity** value.
- **IP Type** – Choose the type of intercept points (IP) to measure: `Output referred` or `Input referred`,

  By default, the testbench measures `Output referred`. This option is available when you set the **Measured quantity** to `IP2` or `IP3`.
- **Injection Type** – Choose the local oscillator (LO) injection for sideband suppression: `Low-side` or `High-side`,

  By default, the testbench measures `Low-side`. This option is available when you set the **Measured quantity** to `Sideband Suppression`.
- **Stimulus branch** — Choose the branch you want to use as input stimulus for the measurement:

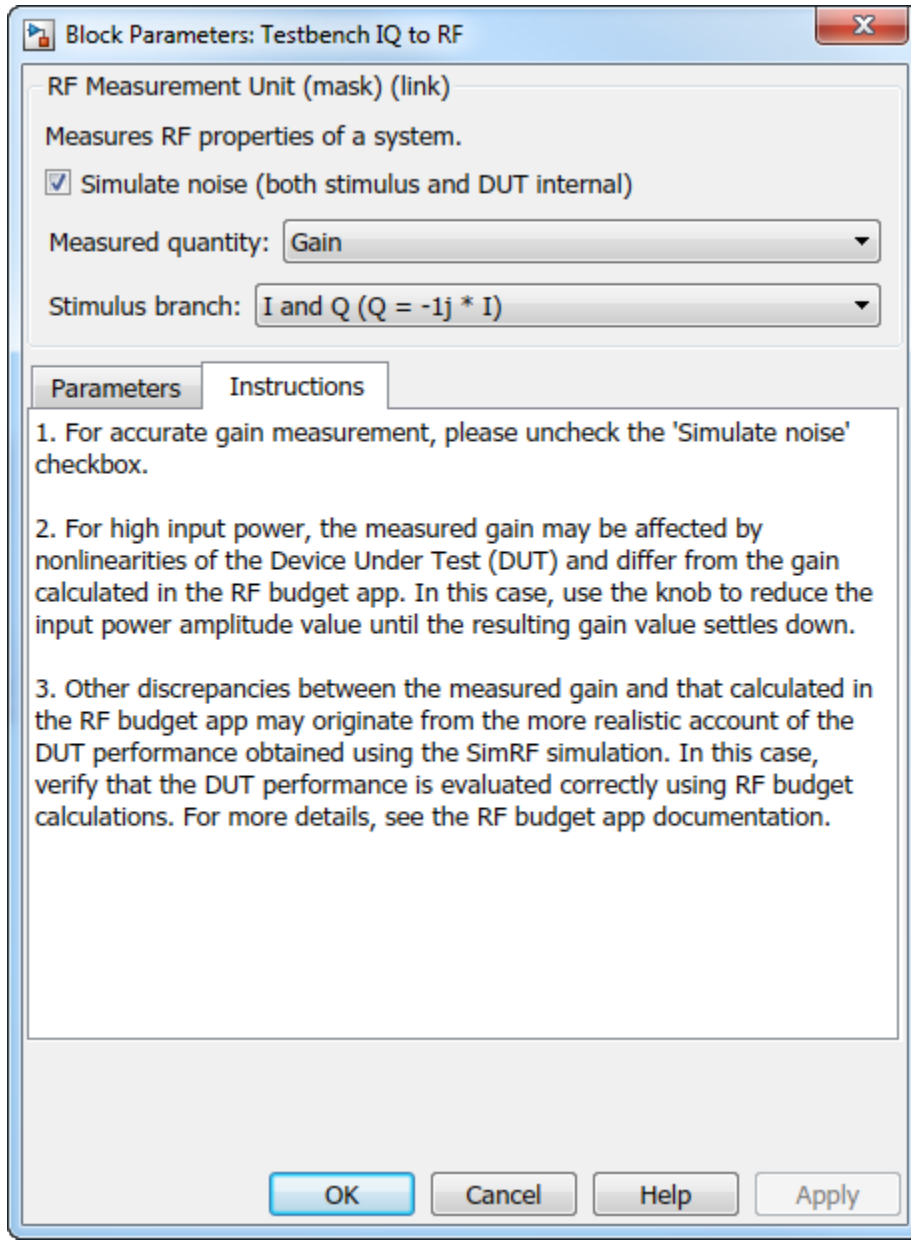
  - `I and Q (Q=-j*I)` — Signal measured is based on a combination of input signals. Quadrature input is same as the in-phase input but is –90 degrees out of phase.
  - `I and Q (Q=j*I)` — Signal measured is based on a combination of input signals. Quadrature input is same as the in-phase input but is 90 degrees out of phase.
  - `I only(Q=0)` — Signal measured is only the output of the in-phase input signal. Gain measured using this input is only one quarter of the total output signal gain.
  - `Q only(I=0)` — Signal verified is only the output of the quadrature input signal. Gain measured using this input is only one quarter of the total output signal gain.

Parameters Tab

- **Input power amplitude (dBm)** — Available input power to the DUT. You can change the input power by manually specifying or by turning the knob. When measuring **Carrier Feedthrough**, this input field is **Input RMS voltage (dBmV)**, because the feedthrough is measured in voltage units. The specified voltage represents the voltage falling on the input ports of the DUT.
- **Input frequency (Hz)** — Carrier frequency fed into the I and Q inputs of the DUT. By default, this frequency is by default one bandwidth above DC.
- **Output frequency (Hz)** — Output frequency to measure the DUT.
- **Baseband bandwidth (Hz)** — Bandwidth of the input signal.
- **Ratio of test tone frequency to baseband bandwidth** — Position of the test tones used for IP3 measurements. By default, the value is `1/8`.

  This option is only available when you set **Measured quantity** to `IP2`, `IP3`, and `Carrier Feedthrough`.

Instructions Tab



Instructions for Gain Measurement

- Clear **Simulate noise (both stimulus and DUT)** for accurate gain measurement. Select the check box for account for noise.
- Change the **Input power amplitude (dBm)** or turn the knob to reduce the input power amplitude. For high input power, nonlinearities in the DUT can affect the gain measurements.

Instructions for Noise Floor Measurement

- The testbench measures the spot noise floor calculated. This calculation assumes a frequency-independent system within a given bandwidth. To simulate a frequency-independent system and calculate the correct noise floor value, reduce the baseband bandwidth until this condition is fulfilled. In common RF systems, the bandwidth is reduced below 1 kHz for noise floor testing.
- Change **Input power amplitude (dBm)** or turn the knob to reduce or increase the input power amplitude. For high input power, nonlinearities in the DUT can affect the noise floor measurements.

Instructions for IP3 and IP2 Measurement

- Clear **Simulate noise (both stimulus and DUT)** for accurate IP3 and IP2 measurement.
- Change **Input power amplitude (dBm)** or turn the knob to reduce the input power amplitude. For high input power, higher-order nonlinearities in the DUT can affect the IP3 and IP2 measurements.

Instructions for Carrier Feedthrough Measurement

- Clear **Simulate noise (both stimulus and DUT)** for accurate IP3 and IP2 measurement.
- Change **Input RMS voltage amplitude (dBmV)** or turn the knob to reduce the input RMS voltage amplitude. For high input RMS voltage, higher-order nonlinearities in the DUT can affect the carrier feedthrough measurements
- Correct calculation of the carrier feedthrough assumes a frequency-independent system in the frequencies surrounding the test tones. Reduce the frequency separation between the test tones or reduce the baseband bandwidth until this condition is fulfilled. In common RF systems, the bandwidth is reduced below `1 KHz` for carrier feedthrough testing.

Instructions for Sideband Suppression Measurement

- Clear **Simulate noise (both stimulus and DUT)** for accurate IP3 and IP2 measurement.
- Change **Input power amplitude (dBm)** or turn the knob to reduce the input power amplitude. For high input power, higher-order nonlinearities in the DUT can affect the sideband suppression measurement.

For all measurements using the testbench, you cannot correct result discrepancies using the **RF Budget Analyzer** app. The RF Blockset testbench provides true RF circuit simulation that incorporates RF phenomena including saturation and interaction between multiple tones and harmonics in nonlinear devices. These RF phenomena are not incorporated in **RF Budget Analyzer**, leading to some differences in the values between the testbench and the app.

See Also

RF Budget Analyzer

RF Blockset Models

5

Analog Devices Transceiver Models

AD9361 Models

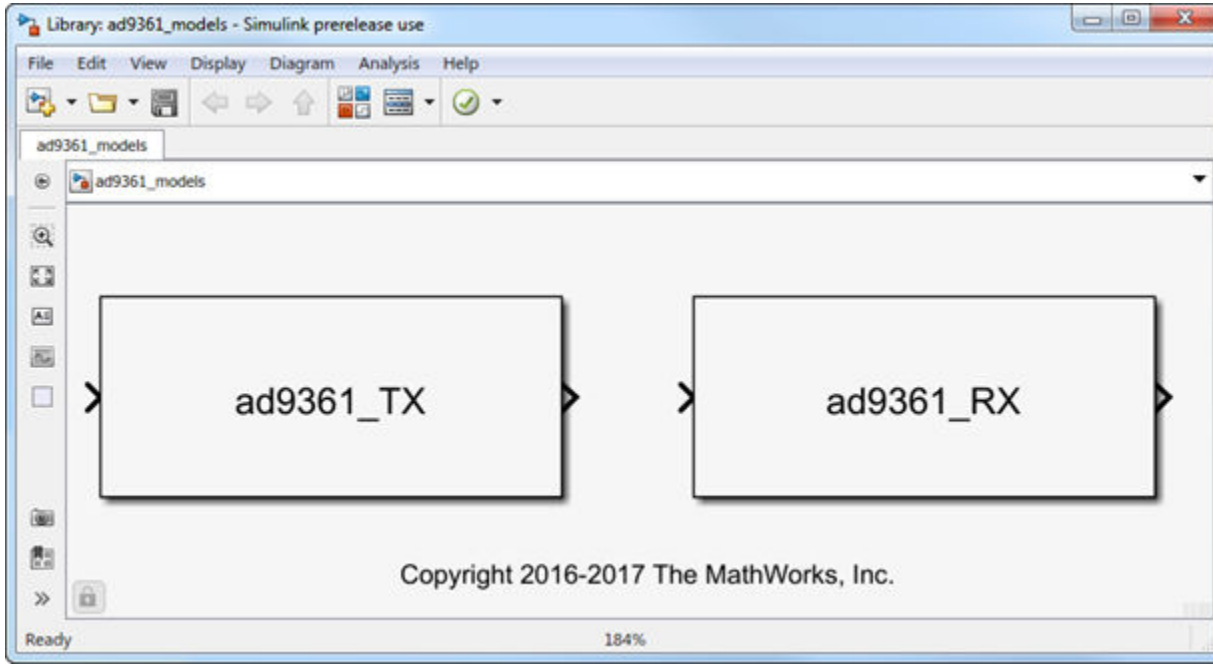
In this section...

You can use the AD9361 models to simulate Analog Devices® AD9361 RF transmitter or receiver designs. These models also helps to see the impact of RF imperfections on your transmitted or received signal.

Install Analog Devices RF Transceivers using, `simrfSupportPackages`. You can open models using the Simulink Library Browser and opening RF Blockset Models for Analog Devices RF Transceivers, or by typing the following in the MATLAB command prompt:

```
open ad9361_models
```

Choose the model you want from the library:



Note You require these additional licenses to run this model:

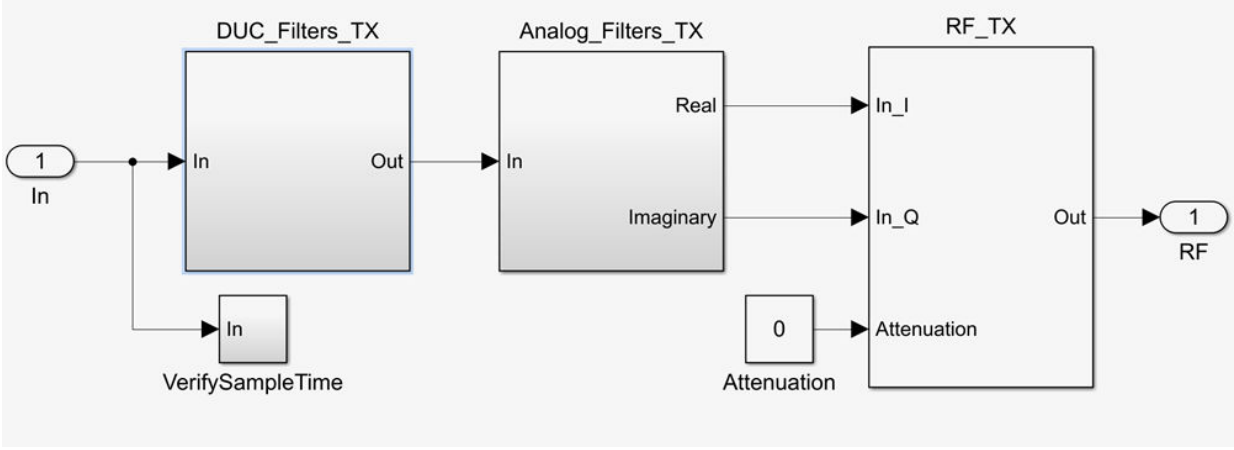
- Communications System Toolbox™
- Stateflow®
- Fixed-Point Designer™

Complete documentation on how to use the models is available with the software download.

```
open(ad93xx_getdoc)
```

AD9361_TX Analog Devices Transmitter

Model Description



The transmitter model consists of three stages:

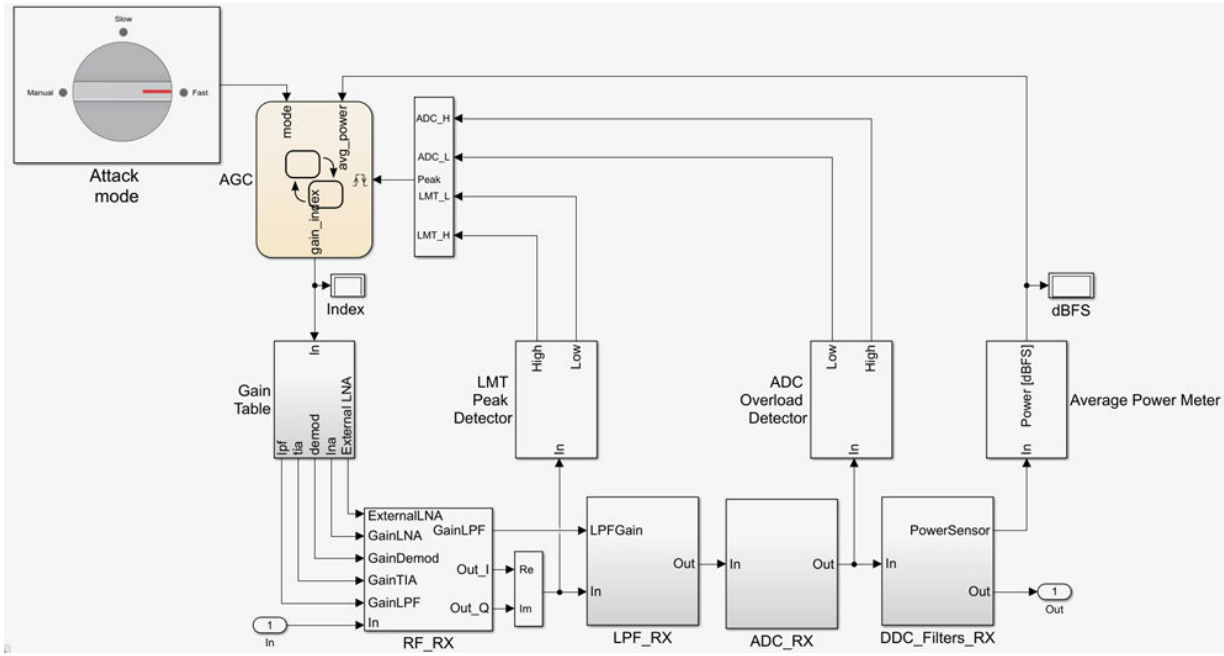
- Digital up-conversion filters (DUC_Filters_TX))
- Analog filters (Analog_Filters_TX)
- RF front end (RF_TX)

You can use the transmitter model to simulate the following behaviors:

- Tunable attenuation
- Oscillator phase noise
- Carrier-dependant noise floor
- Attenuation and carrier-dependant non-linearities like output-referred third-order intercept (OIP3)
- Attenuation dependant gain imbalance
- Attenuation dependant local oscillator (LO) carrier leak

AD9361_RX Analog Devices Receiver

Model Description



The receiver model consists of:

- RF receiver (RF_RX)
- Analog filters (Analog_Filters_RX)

- Analog to Digital Converter (ADC_RX)
- Digital down-conversion filters (DDC)
- Receiver signal strength indicators
- Automatic gain control (AGC) state machine
- Gain table

You can use the receiver model to simulate the following behaviors:

- Tunable low-noise amplifier, mixer, and trans-impedance amplifier (LMT) gains
- Carrier-dependent noise floor
- Gain and carrier-frequency dependant non-linearities like output-referred third-order intercept (OIP3)
- Gain and carrier-frequency dependant non-linearities like output–referred second–order intercept (OIP2)
- Gain dependant gain imbalance
- Gain dependant local oscillator (LO) carrier leak

AD9361 Testbenches

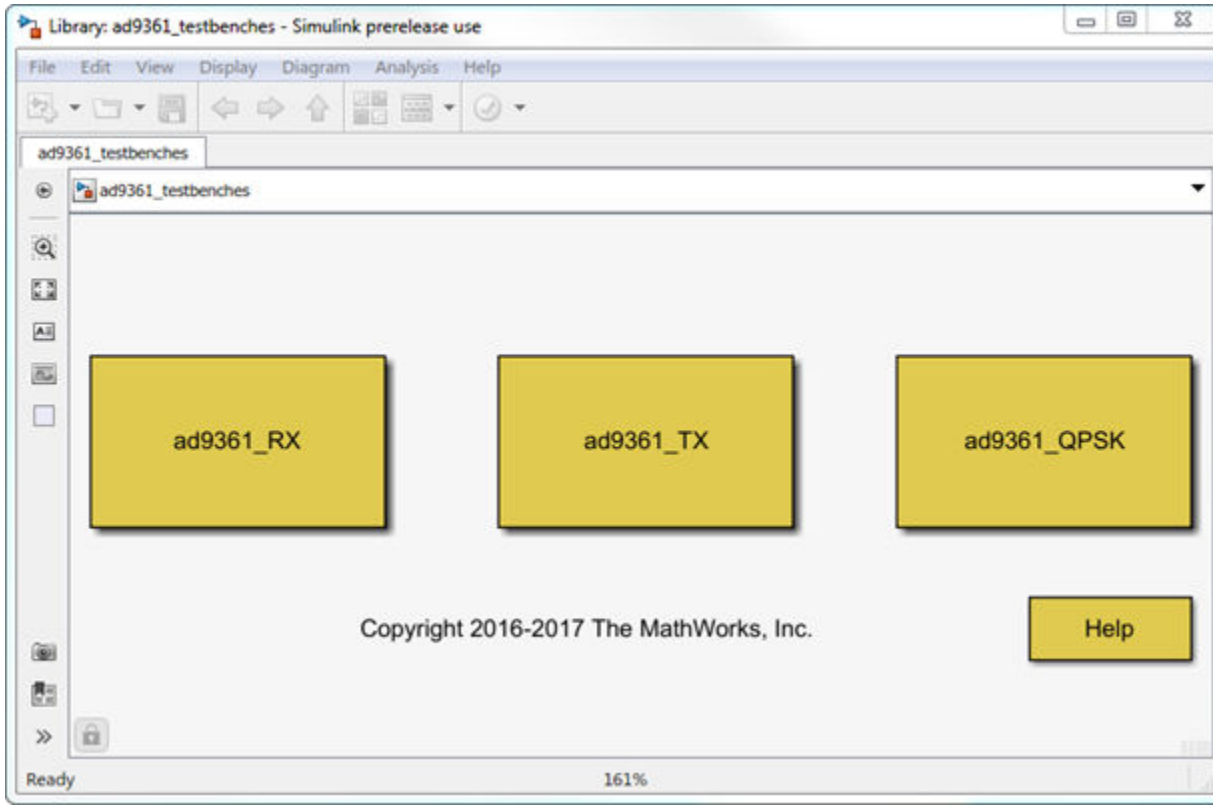
In this section...

You can use the AD9361 testbench models to analyze the functioning and values of Analog Devices AD9361 RF transmitter, receiver, or end-to-end designs.

Install Analog Devices RF Transceivers using, `simrfSupportPackages`. You can open the testbench models using the Simulink library browser and opening RF Blockset Models for Analog Devices RF Transceivers, or by typing the following in the MATLAB command prompt:

```
open ad9361_testbenches
```

Click to open the AD9361 testbench model from the library:



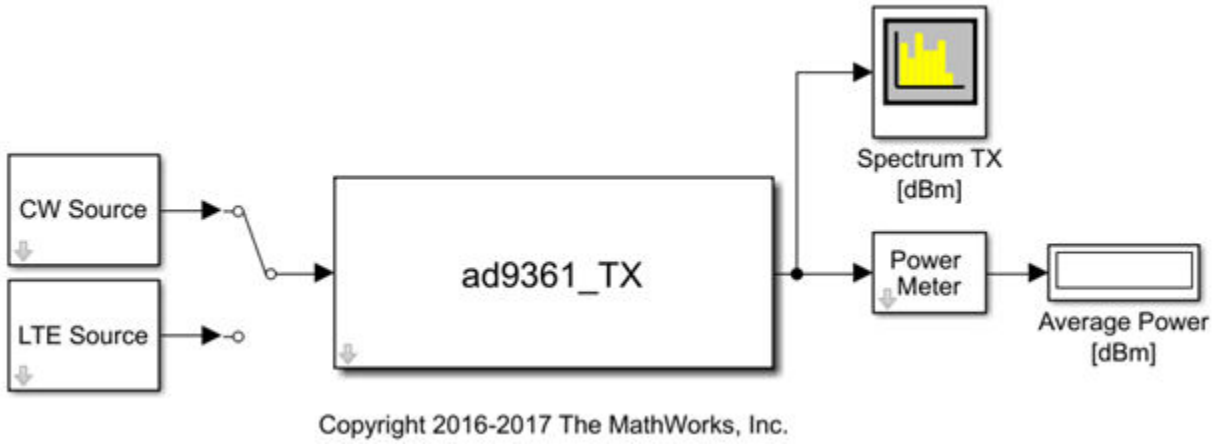
Note You require these additional licenses to run this model:

- Communications System Toolbox
- Stateflow
- Fixed-Point Designer

Complete documentation on how to use the models is available with the software download.

```
open(ad93xx_getdoc)
```

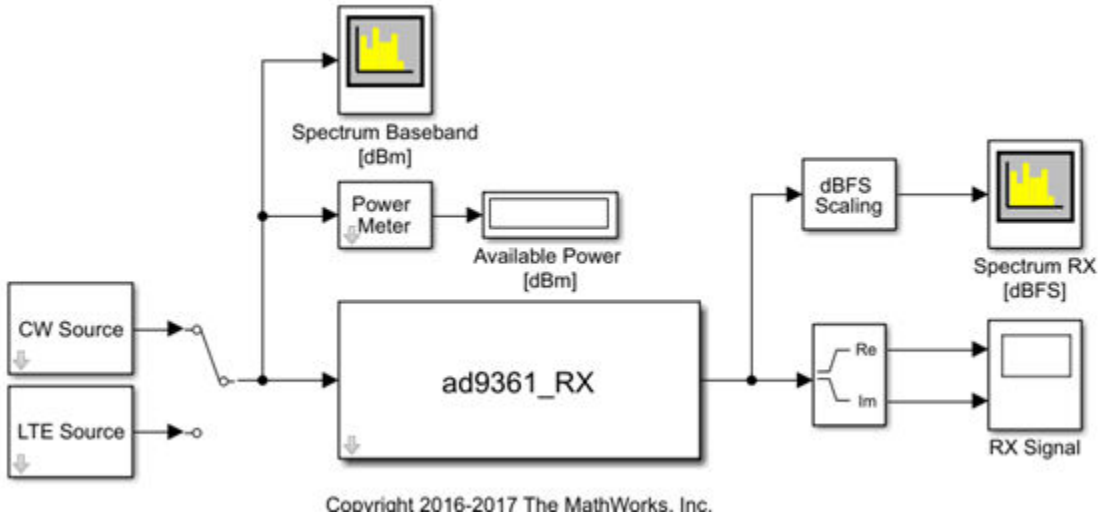
AD9361_TX Analog Devices Transmitter Testbench



The transmitter testbench consists of:

- CW and LTE Signal sources
- AD9361 transmitter which is the device under test
- Spectrum analyzer and power meter for signal visualization

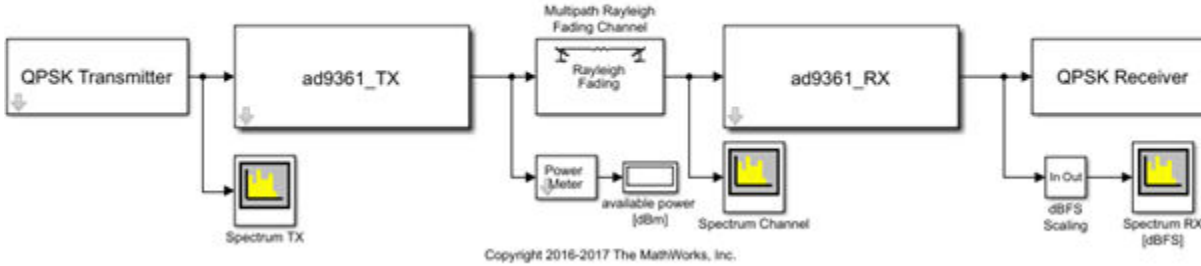
AD9361_RX Analog Devices Receiver Testbench



The receiver testbench consists of:

- CW and LTE Signal sources
- AD9361 receiver which is the device under test
- Spectrum analyzer and power meter for signal visualization

AD9361_QPSK Analog Devices Testbench



The QPSK testbench consists of:

- QPSK signal generator
- AD9361 transmitter operating at 2 GHz, with default LTE 5-MHz filter configuration
- Multi-path Rayleigh fading channel
- Multi-path Rayleigh fading channel
- AD9361 receiver operating at 2 GHz, with default LTE 5-MHz filter configuration
- QPSK baseband signal demodulator decoder.

AD9371 Models

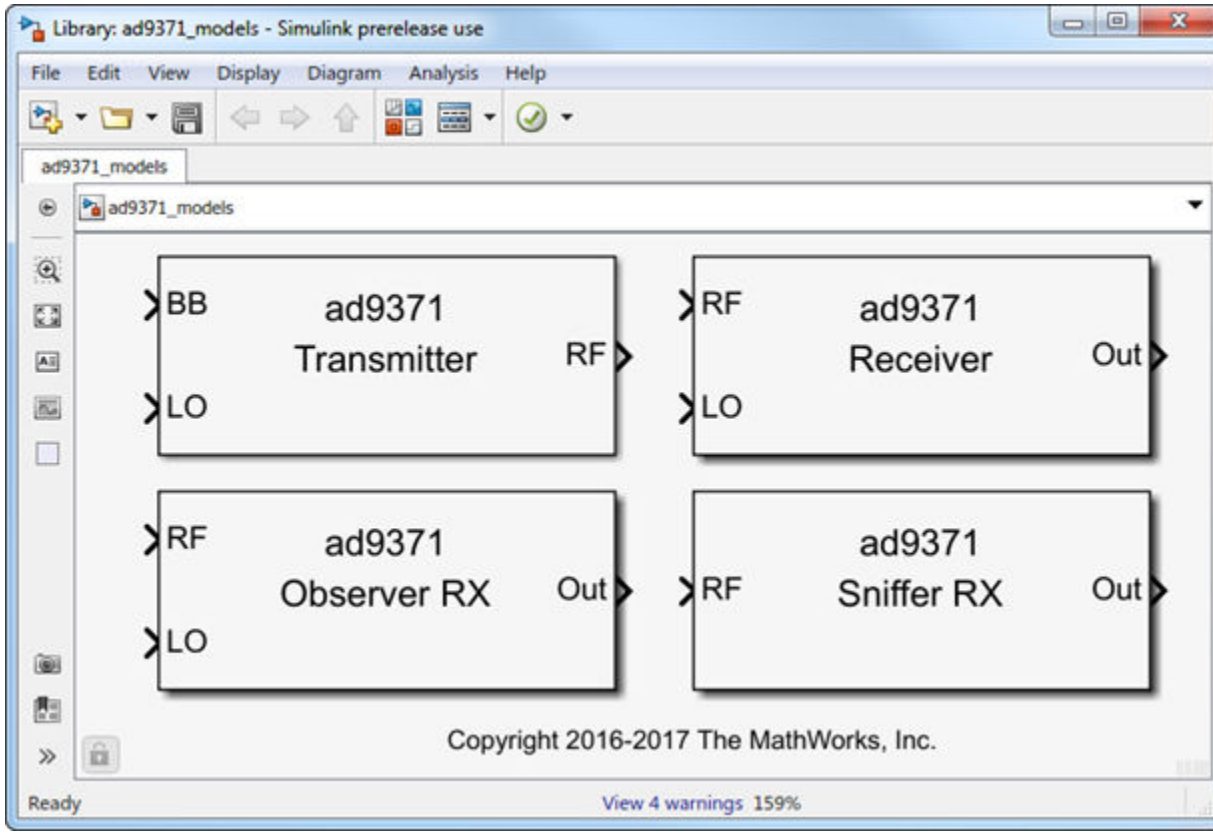
In this section...

You can use the AD9371 models to simulate Analog Devices AD9371 RF transmitter, receiver, observer, and sniffer designs. These models also helps to see the impact of RF imperfections on your transmitted or received signal.

Install Analog Devices RF Transceivers using, `simrfSupportPackages`. You can open models using the Simulink Library Browser and opening RF Blockset Models for Analog Devices RF Transceivers, or by typing the following in the MATLAB command prompt:

```
open ad9371_models
```

Choose the model you want from the library:



Note You require these additional licenses to run this model:

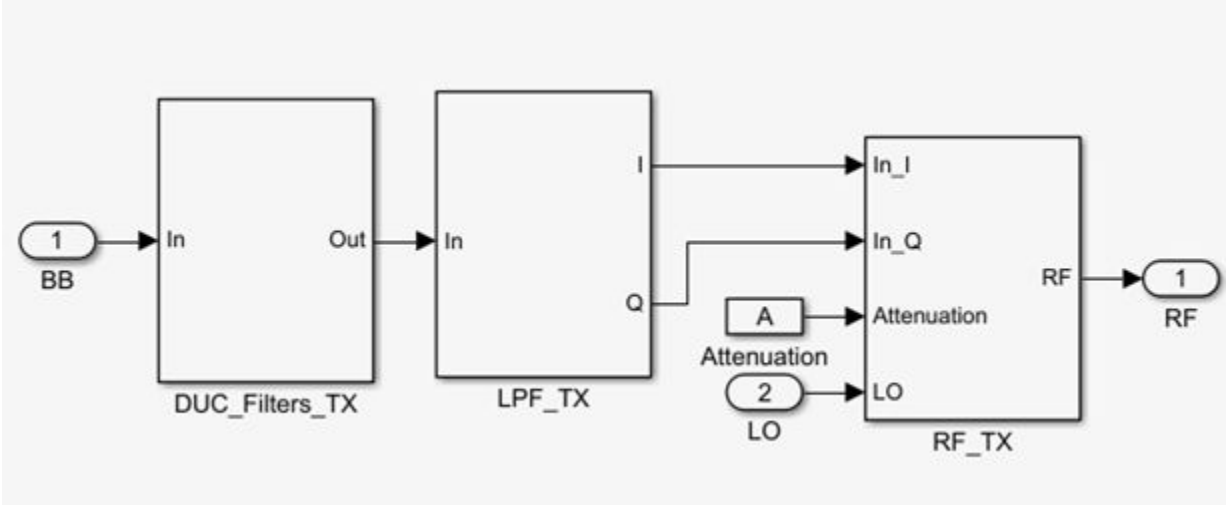
- Communications System Toolbox
- Stateflow
- Fixed-Point Designer

Complete documentation on how to use the models is available with the software download.

```
open(ad93xx_getdoc)
```

AD9371_TX Analog Devices Transmitter

Model Description



The transmitter model consists of three stages:

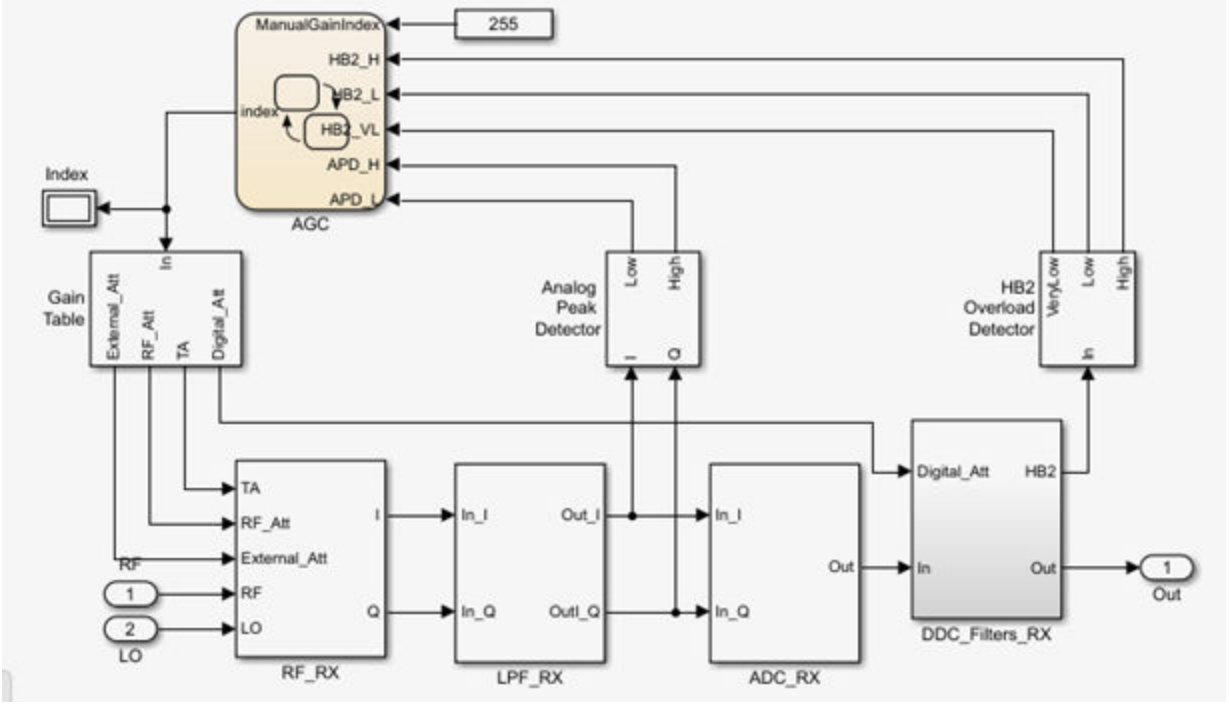
- Digital up-conversion filters (DUC_Filters_TX))
- Analog filters (Analog_Filters_TX)
- RF front end (RF_TX)

You can use the transmitter model to simulate the following behaviors:

- Tunable attenuation
- Attenuation and carrier-frequency dependant noise floor
- Attenuation and carrier-frequency dependant output-referred third-order intercept (OIP3) and output-referred second-order intercept (OIP2)
- Attenuation and carrier-frequency dependant gain imbalance (to model finite image rejection)
- Attenuation and carrier-frequency dependant local oscillator (LO) carrier leakage

AD9371_RX Analog Devices Receiver

Model Description



The receiver model consists of three stages:

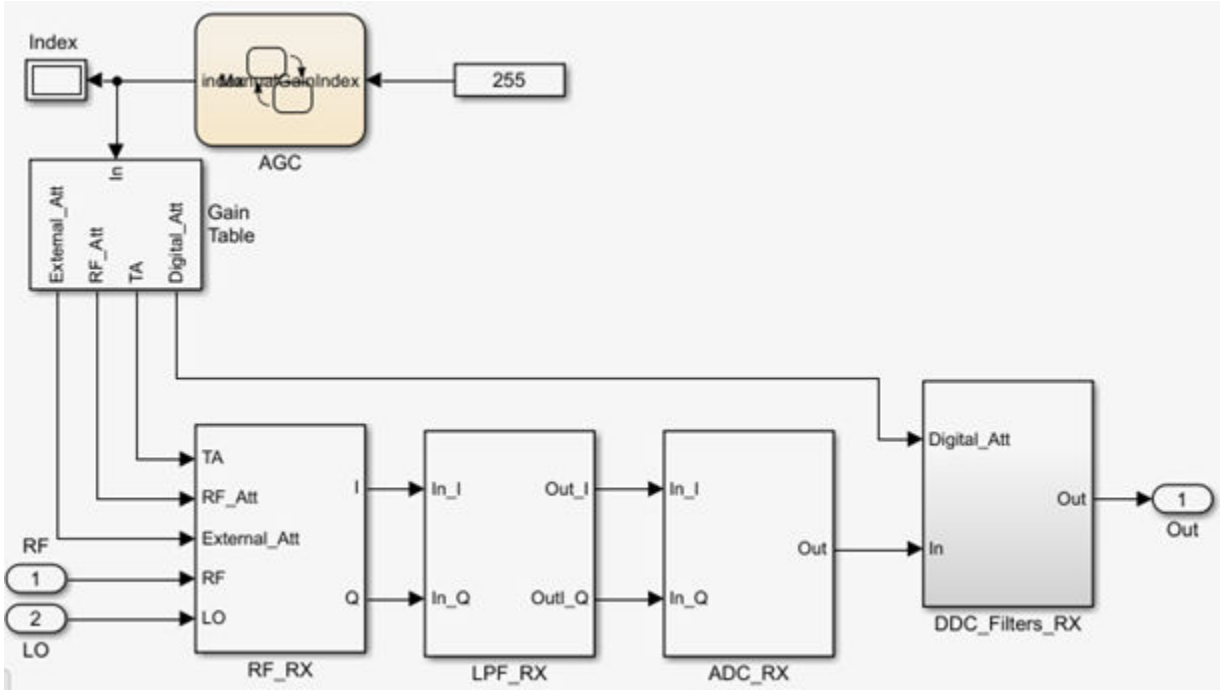
- RF receiver (RF_RX)
- Analog filters (LPF_RX)
- Analog to Digital converter (ADC_RX)
- Digital down-conversion filters (DDC_Filters_RX)
- Receiver signal strength indicator (RSSI): two power meters to detect the strength of the received signal at different section of the chain (Analog Peak Detector, and HB2 Overload Detector)
- Automatic gain control state machine (AGC)
- Programmable gain table (Gain Table)

You can use the receiver model simulate the following behaviors:

- Tunable internal attenuation
- Attenuation and carrier-frequency dependant noise figure
- Attenuation and carrier-frequency dependant output-referred third-order intercept (OIP3) and output-referred second-order intercept (OIP2)
- Attenuation and carrier-frequency dependant gain imbalance (to model finite image rejection)
- Attenuation and carrier-frequency dependant local oscillator (LO) carrier leakage
- Analog filters provide continuous time signal
- ADC models a high-sampling rate third order delta-sigma modulator.
- Digital down conversion digital filters converts the highly sampled signal at the output of the ADC to a lower baseband rate.
- Received signal strength indicator measures power at two stages, at the RF receiver output and after the first half-band filter
- AGC changes the index of the gain table according to the flags of threshold crossing reported by the RSSI.
- Default gain table is read from the MATLAB file, `DefaultGainTables`. You can customize this file.

AD9371_ORX Analog Devices Observer Receiver

Model Description



The observer receiver model consists of three stages:

- RF receiver (RF_RX)
- Analog filters (LPF_RX)
- Analog to Digital converter (ADC_RX)
- Digital down-conversion filters (DDC_Filters_RX)
- Automatic gain control state machine (AGC) operating in manual control mode
- Programmable gain table (Gain Table)

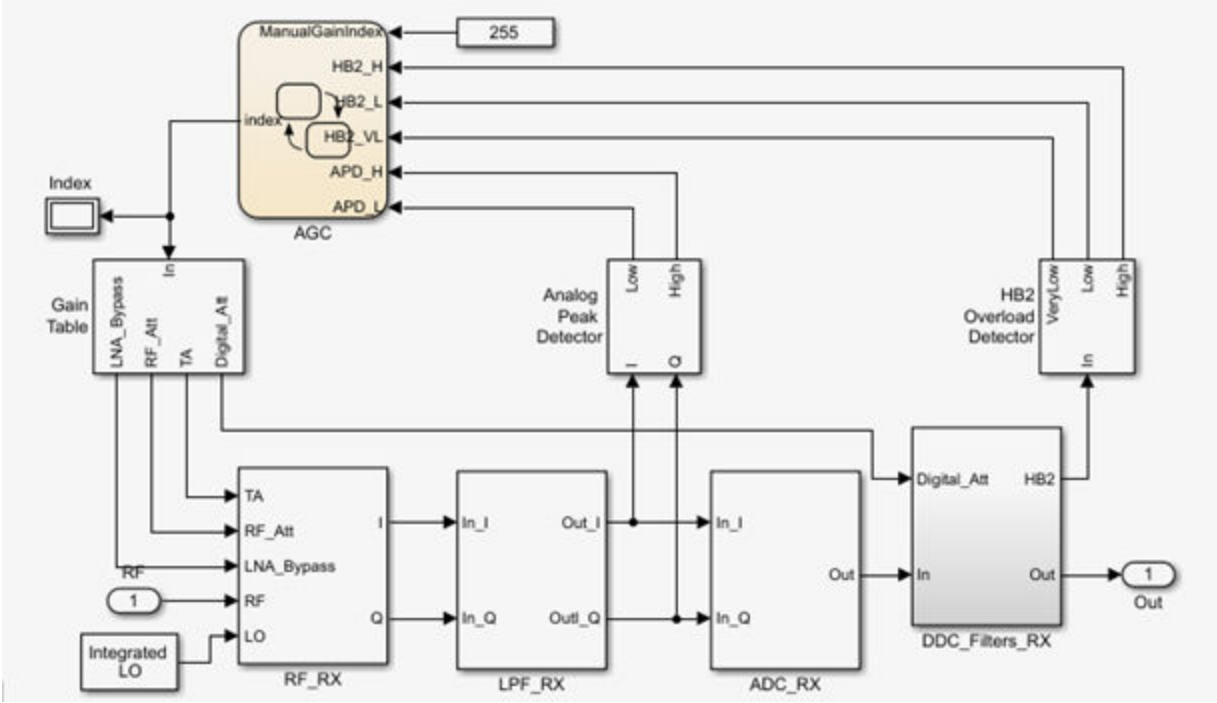
You can use the observer receiver model to simulate the following behaviors:

- Tunable internal attenuation

- Attenuation and carrier-frequency dependant noise figure
- Attenuation and carrier-frequency dependant output-referred third-order intercept (OIP3) and output-referred second-order intercept (OIP2)
- Attenuation and carrier-frequency dependant gain imbalance (to model finite image rejection)
- Attenuation and carrier-frequency dependant local oscillator (LO) carrier leakage
- Analog filters provide continuous time signal.
- ADC models a high-sampling rate third-order delta-sigma modulator.
- Digital down conversion digital filters converts the highly sampled signal at the output of the ADC to a lower baseband rate.
- Received signal strength indicator measures power at two stages, at the RF receiver output and after the first half-band filter
- Default gain table is read from the MATLAB file, `DefaultGainTables`. You can customize this file.

AD9371_SNF Analog Devices Sniffer Receiver

Model Description



The sniffer receiver model consists of three stages:

- RF receiver (RF_RX)
- Analog filters (LPF_RX)
- Analog to Digital converter (ADC_RX)
- Digital down-conversion filters (DDC_Filters_RX)
- Receiver signal strength indicator (RSSI): two power meters to detect the strength of the received signal at different section of the chain (Analog Peak Detector, and HB2 Overload Detector)
- Automatic gain control state machine (AGC) operating in manual control mode
- Programmable gain table (Gain Table)

You can use the AD9371 sniffer receiver model to simulate the following behaviors:

- Tunable internal attenuation
- Carrier-frequency dependant noise figure
- Attenuation and carrier-frequency dependant output-referred third-order intercept (OIP3) and output-referred second-order intercept (OIP2)
- Attenuation and carrier-frequency dependant gain imbalance (to model finite image rejection)
- Attenuation and carrier-frequency dependant local oscillator (LO) carrier leakage
- Analog filters provide continuous time signal.
- ADC models a high-sampling rate third-order delta-sigma modulator.
- Digital down conversion digital filters converts the highly sampled signal at the output of the ADC to a lower baseband rate.
- Received signal strength indicator measures power at two stages, at the RF receiver output and after the first half-band filter
- AGC changes the index of the gain table according to the flags of threshold crossing reported by the RSSI.
- Default gain table is read from the MATLAB file, `DefaultGainTables`. You can customize this file.

AD9371 Testbenches

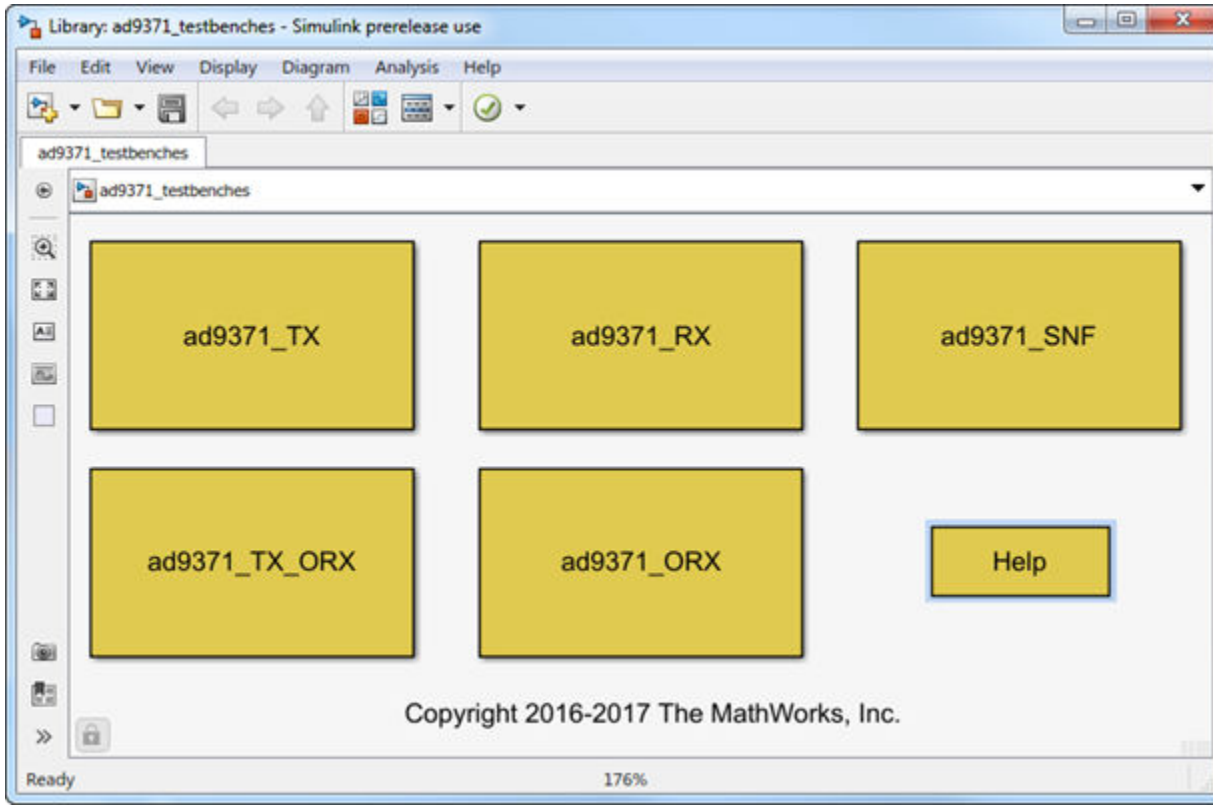
In this section...

You can use the AD9371 testbench models to analyze the functioning and values of Analog Devices AD9371 RF transmitter, receiver, observer, sniffer, or end-to-end designs.

Install Analog Devices RF Transceivers using, `simrfSupportPackages`. You can open the testbench models using the Simulink library browser and opening RF Blockset Models for Analog Devices RF Transceivers, or by typing the following in the MATLAB command prompt:

```
open ad9371_testbenches
```

Click to open the AD9371 testbench model from the library:



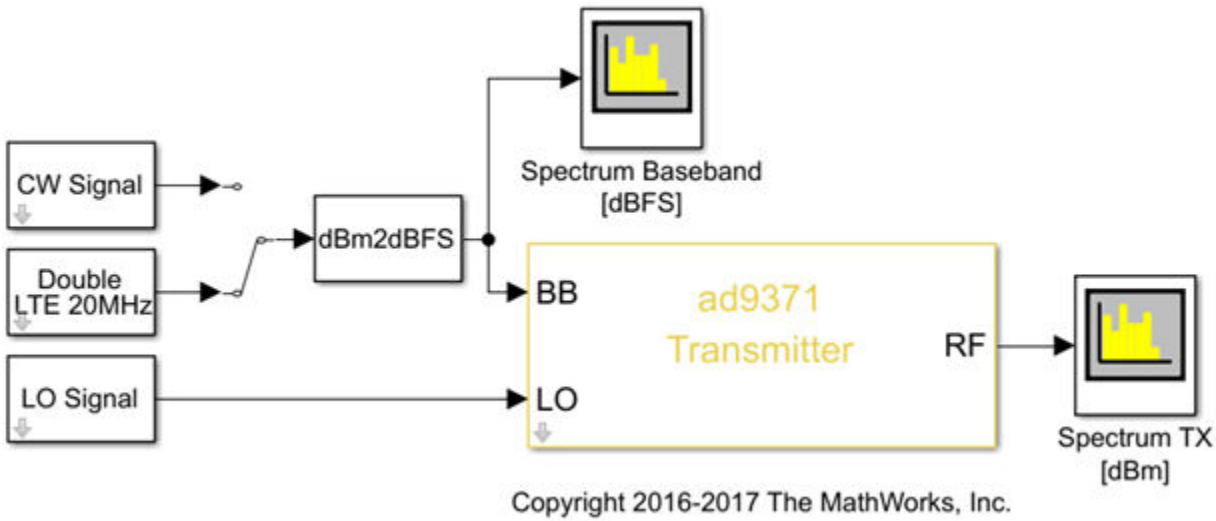
Note You require these additional licenses to run this model:

- Communications System Toolbox
- Stateflow
- Fixed-Point Designer

Complete documentation on how to use the models is available with the software download.

```
open(ad93xx_getdoc)
```

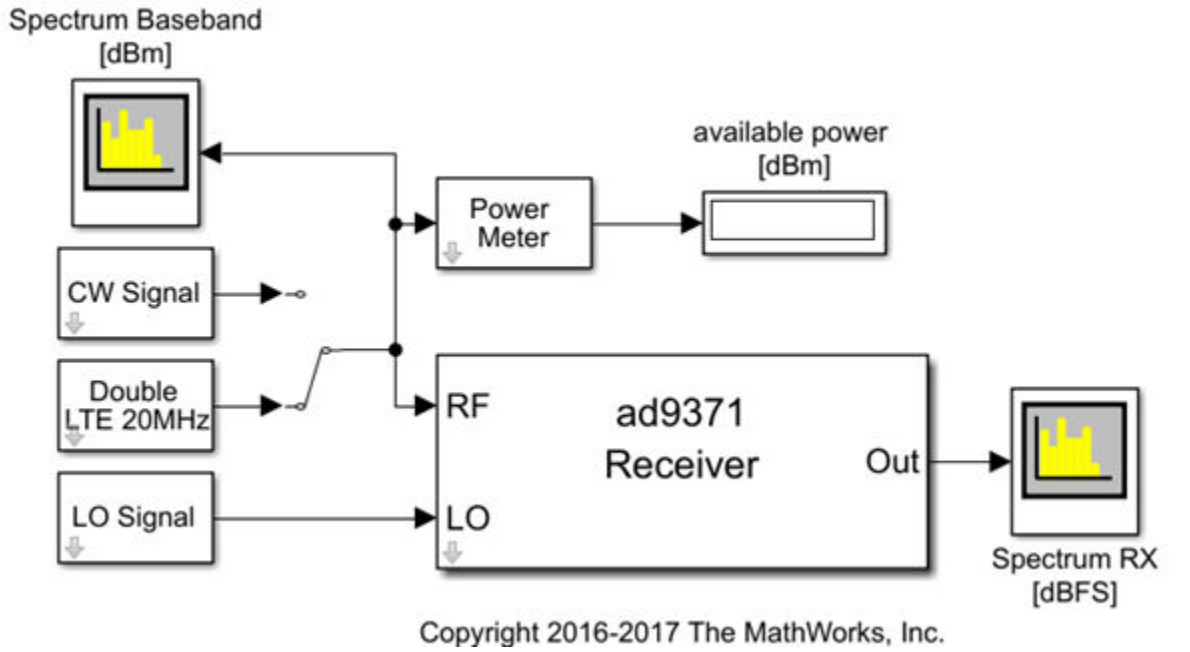
AD9371_TX Analog Devices Transmitter Testbench



The transmitter testbench consists of:

- CW and LTE Signal sources
- External local oscillator signal source
- AD9371 transmitter which is the device under test
- Spectrum analyzer and power meter for signal visualization

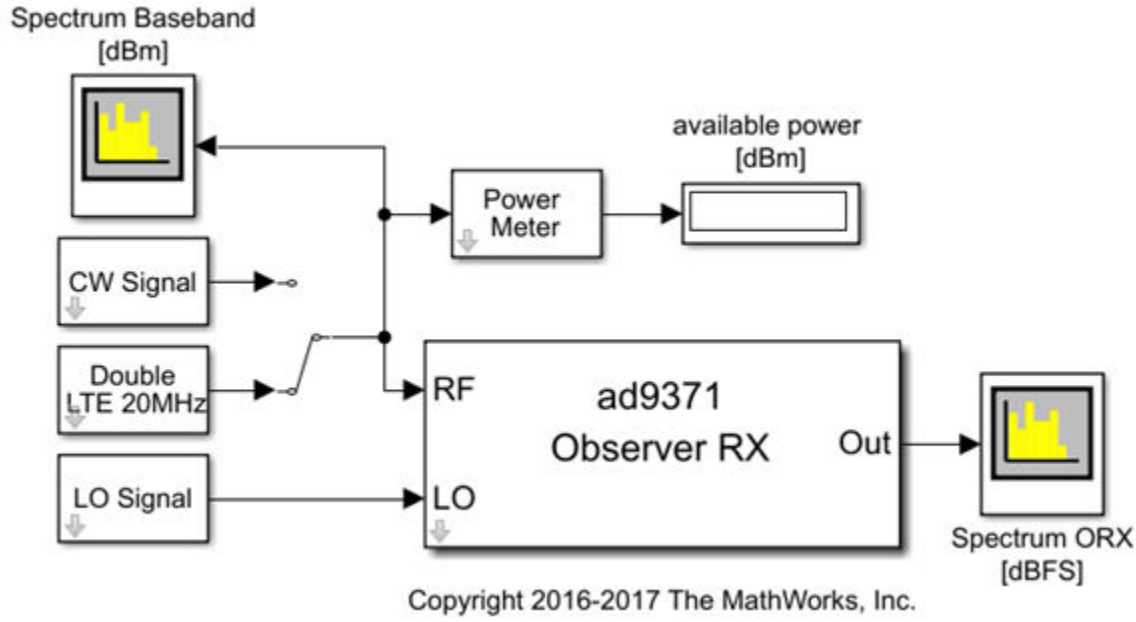
AD9371_RX Analog Devices Receiver Testbench



The receiver testbench consists of:

- CW and LTE 20 MHz Signal sources
- External local oscillator signal source
- AD9371 receiver which is the device under test
- Spectrum analyzer and power meter for signal visualization

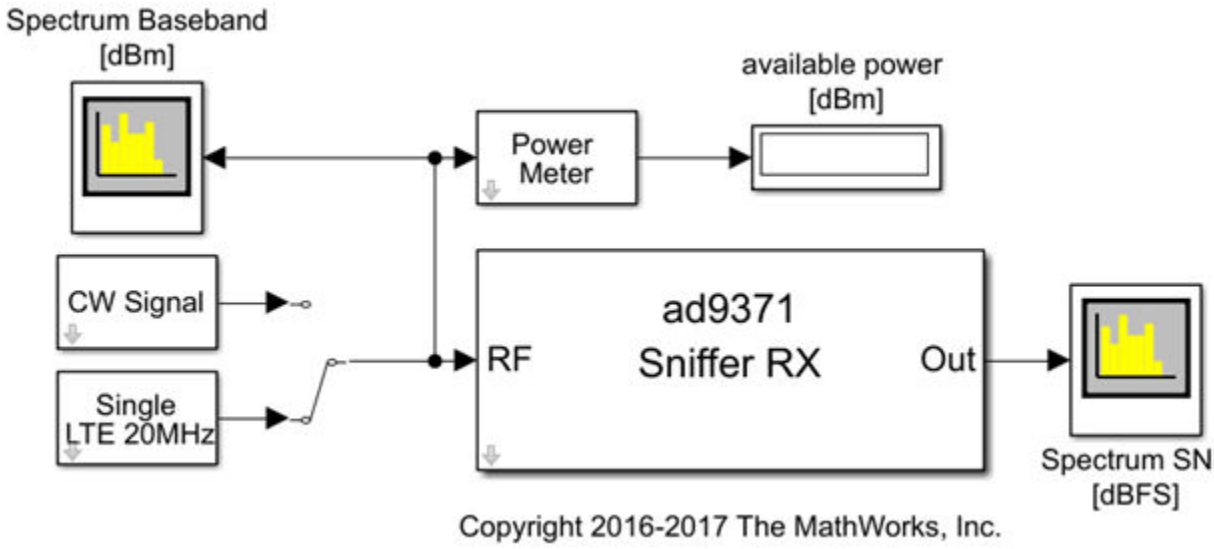
AD9371_ORX Analog Devices Observer Receiver Testbench



The observer receiver testbench consists of:

- CW and LTE 20 MHz Signal sources
- External local oscillator signal source
- AD9371 receiver which is the device under test
- Spectrum analyzer and power meter for signal visualization

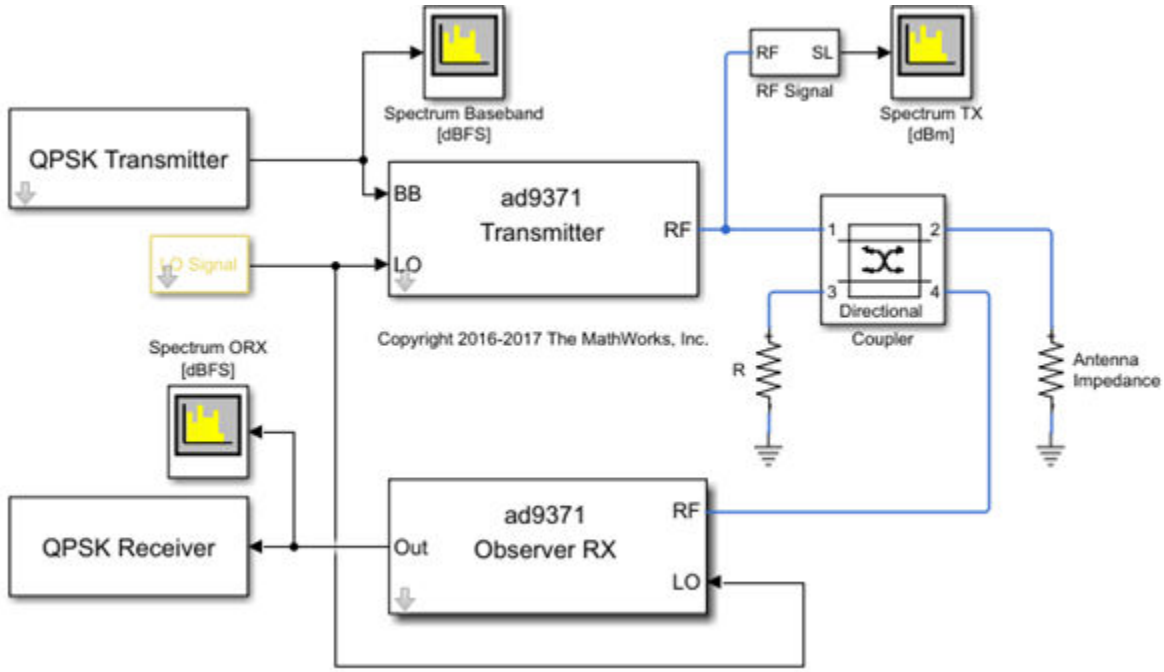
AD9371_SNF Analog Devices Sniffer Receiver Testbench



The sniffer receiver testbench consists of:

- CW and LTE 20 MHz Signal sources
- AD9371 sniffer receiver which is the device under test
- Spectrum analyzer and power meter for signal visualization

AD9371_TX_ORX Analog Devices Transmitter-Observer Testbench



The transmitter-observer testbench consists of:

- CW and LTE 20 MHz Signal sources
- External local oscillator signal source driving both transmitter and observer
- AD9371 transmitter and observer connected via directional coupler
- Spectrum analyzer and power meter for signal visualization

Equivalent Baseband

Model an RF System

Model RF Components

In this section...

Add RF Blocks to a Model

You can include blocks from the RF Blockset Equivalent Baseband Physical and Mathematical libraries in a Simulink model. For more information on the libraries and the available RF blocks, see “RF Blockset Equivalent Baseband Libraries”.

This section contains the following topics:

- “Input Signal Requirements” on page 6-2
- “How to Add RF Blocks to a Model” on page 6-2

Input Signal Requirements

Most RF Blockset Equivalent Baseband blocks only support complex single-channel signals. The signals can be either sample-based or frame-based. The following blocks have this requirement:

- All Physical blocks
- Mathematical Amplifier and Mixer blocks

You can model the effect of these components on a multichannel signal as follows:

1. Use a Simulink Demux block to split the multichannel signal into single-channel signals.
2. Create duplicate RF models, with one model for each channel, and pass each single-channel signal into a separate model.
3. Use a Simulink Mux block multiplex the signals at the output of the RF models.

How to Add RF Blocks to a Model

To add RF blocks to a Simulink model:

1. Type `rflib` at the MATLAB prompt to open the RF Blockset Equivalent Baseband library.
2. Navigate to the desired library or sublibrary.
3. Drag instances of RF Blockset Equivalent Baseband blocks into the model window using the mouse.

Note You can also access RF Blockset Equivalent Baseband blocks and other Simulink blocks from the Simulink Library Browser window. Open this window by typing `simulink` at the MATLAB prompt. Add blocks to the model by dragging them from this window and dropping them into the model window.

Connect Model Blocks

You follow the same procedure for connecting RF Blockset Equivalent Baseband blocks as for connecting Simulink blocks: you click a port and drag the mouse to draw a line to another port on a different block.

You can only connect blocks that use the same type of signal. Physical library blocks use different types of signals than Mathematical library blocks, and are represented graphically by a different port style. Therefore, you can freely connect pairs of Mathematical modeling blocks. You can also freely connect pairs of Physical modeling blocks. However, you cannot directly connect Physical blocks to Mathematical blocks. Instead, you must use the Input Port and Output Port blocks to bridge them.

For more information on the Physical and Mathematical libraries, including how to open the libraries and a description of the available blocks, see “Overview of RF Blockset Equivalent Baseband Libraries”.

This section contains the following topics:

Connect Mathematical Blocks

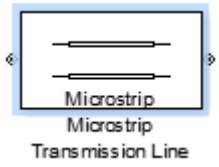
The Mathematical library blocks use the same input and output ports as standard Simulink blocks. These ports show the direction of the signal at the port, as shown in the following diagram.

Similar to standard Simulink blocks, you draw lines between the ports of the Mathematical modeling blocks, called *signal lines*, to represent signals that are inputs to and outputs from the mathematical functions represented by the blocks. Therefore, you can connect Simulink, DSP System Toolbox, and RF Blockset Equivalent Baseband mathematical blocks by drawing signal lines between their ports.

You can connect a port to multiple ports by branching the signal line, or you can leave a port unconnected.

Connect Physical Blocks

The Physical library blocks have specialized *connector ports*. These ports only represent physical connections; they do not imply signal direction.



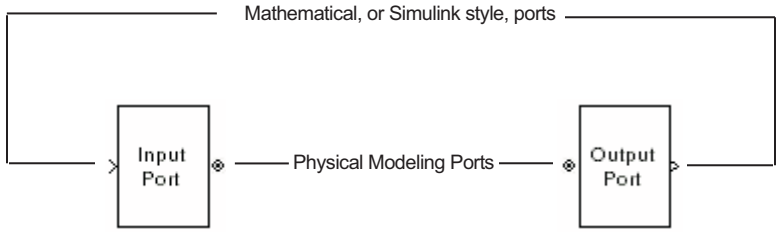
The lines you draw between the physical modeling blocks, called *connection lines*, represent physical connections among the block components. Connection lines appear as solid black when connected and as dashed red lines when either end is unconnected.

You can draw connection lines only between the connector ports of physical modeling blocks. You cannot branch these connection lines. You cannot leave connector ports unconnected.

Bridge Physical and Mathematical Blocks

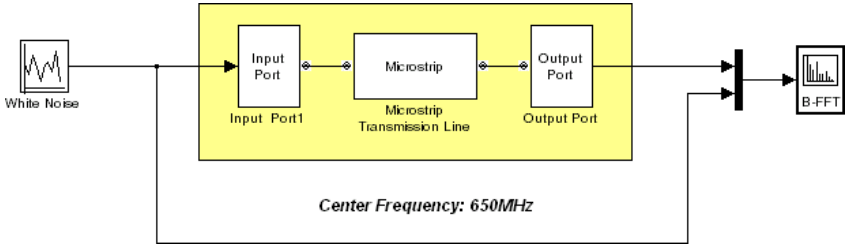
The blockset provides the Input Port and Output Port blocks to connect the physical and mathematical parts of the model. These blocks convert mathematical signals to and from the physical modeling environment.

The Input Port and Output Port blocks have one of each kind of connector port: a standard Simulink style input port and a physical modeling port. These ports are shown in the following figure:



The Input Port and Output Port blocks must bound a physical subsystem to connect it to the mathematical part of a model.

For example, a simple RF model of a coaxial transmission line might resemble the following figure.



The Microstrip Transmission Line block uses an Input Port block to get its white noise input from a Random Source block, and an Output Port block to pass its output to a Spectrum Scope block. The Random Source and Spectrum Scope blocks are from DSP System Toolbox library.

For information on how RF Blockset Equivalent Baseband software converts mathematical signals to and from the physical modeling environment, see “Convert to and from Simulink Signals” on page A-31.

Specify or Import Component Data

In this section...

Specify Parameter Values

There are two ways to set block parameter values:

- Using the GUI — Enter information in the block dialog boxes, which open when you double-click a block in the Simulink window.
- Using commands — Use the Simulink `set_param` and `get_param` commands to set and get parameter values of the blocks, respectively. For more information on these commands, see the `set_param` and `get_param` reference pages.

Supported File Types for Importing Data

The blockset also lets you import the following types of data files:

- Industry-standard file formats — Touchstone S2P, Y2P, Z2P, and H2P formats specify the network parameters and noise information for measured and simulated data.

  For more information on Touchstone files, see `https://ibis.org/connector/touchstone_spec11.pdf`.
- Agilent® P2D file format — Specifies amplifier and mixer large-signal, power-dependent network parameters, noise data, and intermodulation tables for several operating conditions, such as temperature and bias values.

  The P2D file format lets you import system-level verification models of amplifiers and mixers.

- Agilent S2D file format — Specifies amplifier and mixer network parameters with gain compression, power-dependent S_{21} parameters, noise data, and intermodulation tables for several operating conditions.

  The S2D file format lets you import system-level verification models of amplifiers and mixers.
- MathWorkso amplifier (AMP) file format — Specifies amplifier network parameters, power data, noise data, and third-order intercept point

  For more information about `.amp` files, see “AMP File Data Sections” (RF Toolbox).
- MATLAB circuits — RF Toolbox™ circuit objects in the MATLAB workspace specify network parameters, noise data, and third-order intercept point information of circuits with different topologies.

  For more information about RF circuit objects, see “RF Circuit Objects” (RF Toolbox).

Import Data Files into RF Blocks

The blockset lets you import industry-standard data files, Agilent P2D and S2D files, and MathWorks AMP files into specific blocks to simulate the behavior of measured components in the Simulink modeling environment.

This section contains the following topics:

- “Blocks Used to Import Data” on page 6-7
- “How to Import Data Files” on page 6-7

Blocks Used to Import Data

Three blocks in the Physical library accept data from a file. The following table lists the blocks and any corresponding data format that each supports.

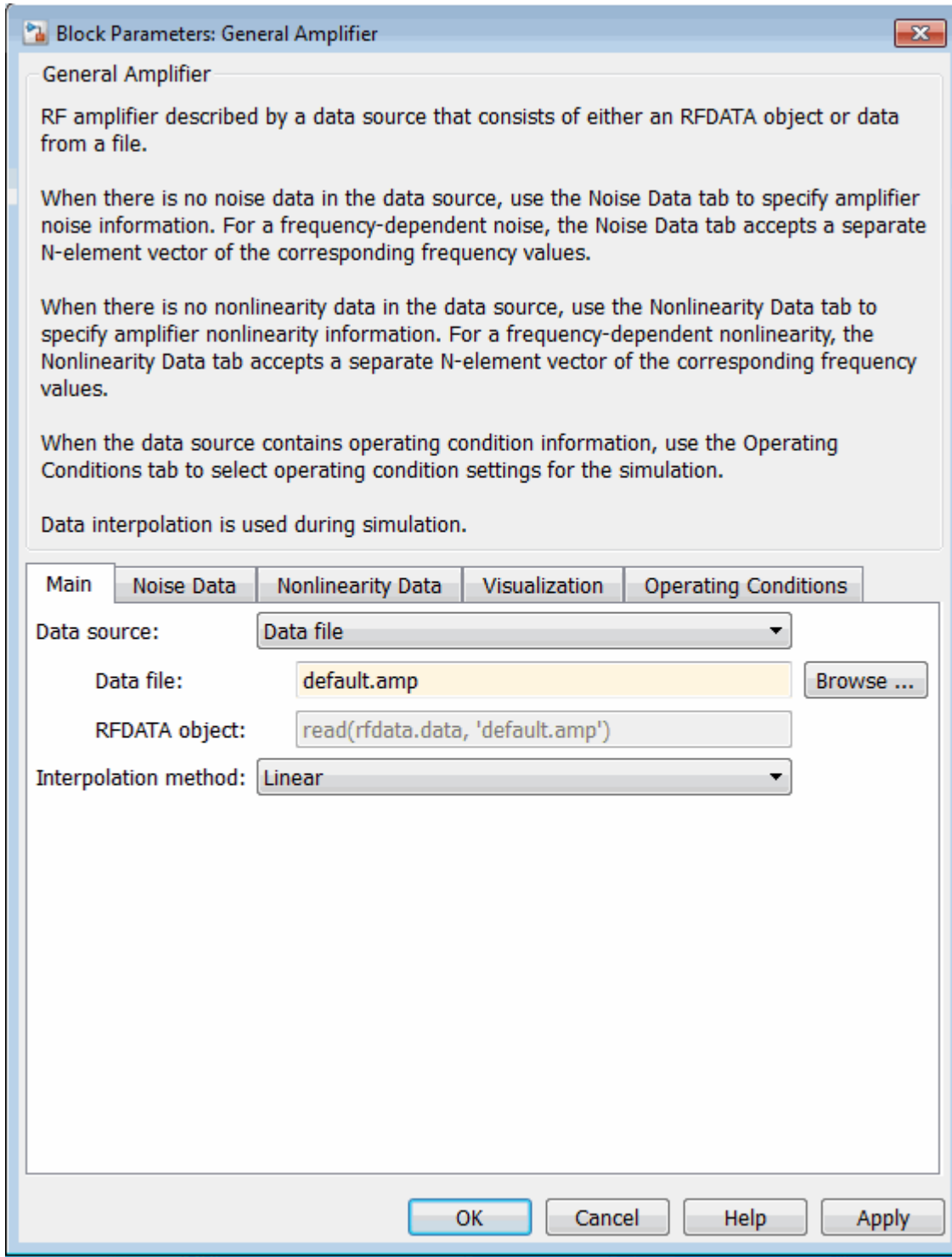
Block	Description	Supported Format(s)
General Amplifier	Generic amplifier	Touchstone, AMP, P2D, S2D
General Mixer	Generic mixer	Touchstone, AMP, P2D, S2D
General Passive Network	Generic passive component	Touchstone

How to Import Data Files

To import a data file:

1. Choose the block that best represents your component from the list of blocks that accept file data shown in “Blocks Used to Import Data” on page 6-7.
2. Open the Physical library, and navigate to the sublibrary that contains the block.
3. Click and drag the block into your Simulink model.
4. In the block dialog box, enter the name of your data file for the **Data file** parameter. The file name must include the extension. If the file is not in your MATLAB path, specify the full path to the file or use the **Browse** button to find the file.

Note The **Data file** parameter is only enabled when the **Data source** parameter is set to `Data file`. This is the default setting and it means the block data comes from a file.


Block Parameters: General Amplifier
General Amplifier
RF amplifier described by a data source that consists of either an RFDATA object or data from a file.
When there is no noise data in the data source, use the Noise Data tab to specify amplifier noise information. For a frequency-dependent noise, the Noise Data tab accepts a separate N-element vector of the corresponding frequency values.
When there is no nonlinearity data in the data source, use the Nonlinearity Data tab to specify amplifier nonlinearity information. For a frequency-dependent nonlinearity, the Nonlinearity Data tab accepts a separate N-element vector of the corresponding frequency values.
When the data source contains operating condition information, use the Operating Conditions tab to select operating condition settings for the simulation.
Data interpolation is used during simulation.
Main
Noise Data
Nonlinearity Data
Visualization
Operating Conditions
Data source:
Data file
Data file:
default.amp
Browse ...
RFDATA object:
read(rfdata.data, 'default.amp')
Interpolation method:
Linear
OK
Cancel
Help
Apply

The following section shows an example of this procedure.

Example — Import a Touchstone Data File into an RF Model

In this example, you model the frequency response of a passive component using data from a Touchstone file, `defaultbandpass.s2p`.

You use a model from one of the RF Blockset Equivalent Baseband examples to perform the following tasks:

- “Import Data into a General Passive Network Block” on page 6-10
- “Validate the Passive Component” on page 6-12

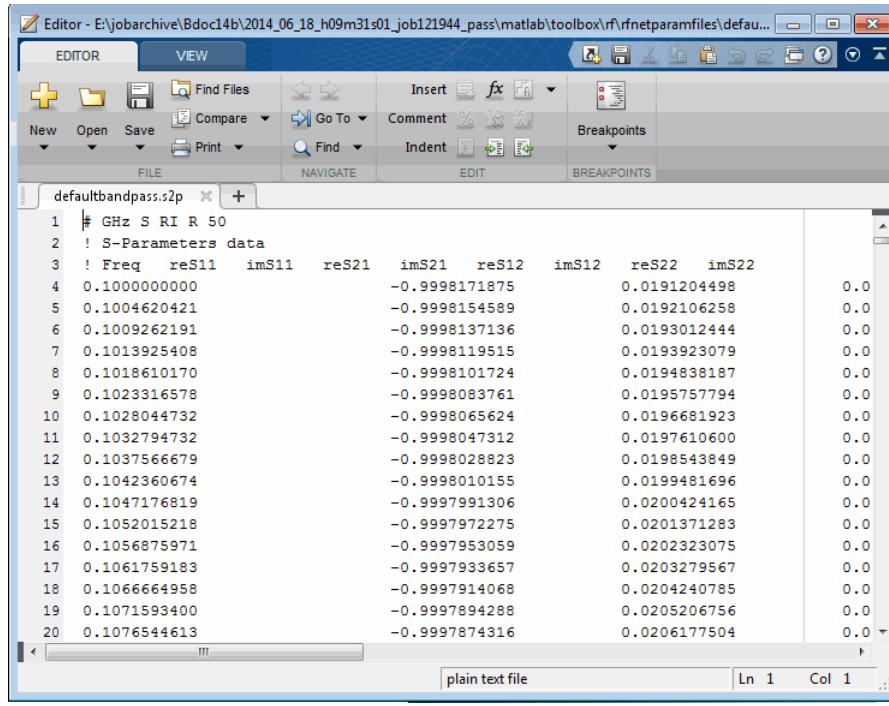
Import Data into a General Passive Network Block

In this part of the example, you inspect the `defaultbandpass.s2p` file and import data into the RF model using the General Passive Network block.

1. Type the following at the MATLAB prompt to open the `defaultbandpass.s2p` file:

   ```
   edit defaultbandpass.s2p
   ```

   The following figure shows a portion of the `.s2p` file.



The option line

```
# GHz S RI R 50
```

specifies the following information about the contents of the data file:

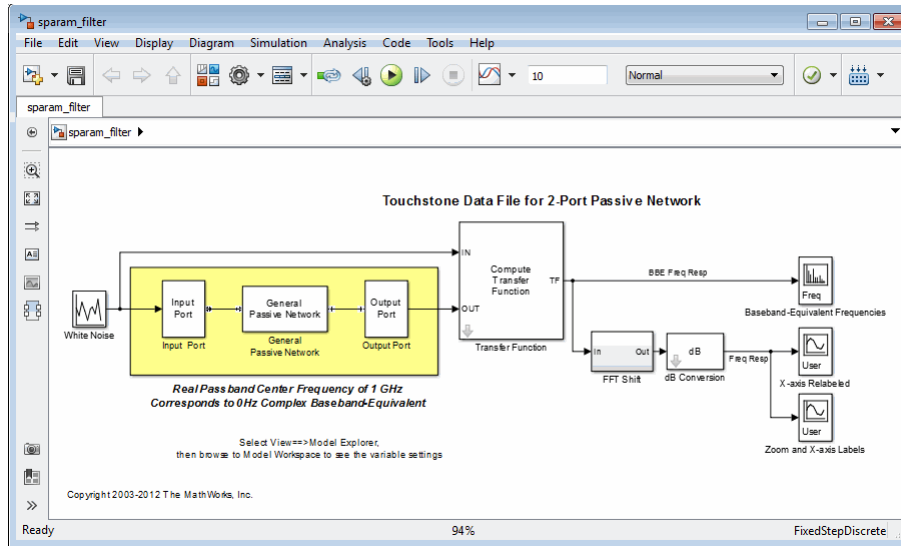
- GHz — Frequency units.
- S — Network parameters are S-parameters.
- RI — Network parameters are specified as the real and imaginary parts.
- R 50 — Reference impedance is 50 ohms.

For more information about the Touchstone specification, including the option line, see `https://ibis.org/connector/touchstone_spec11.pdf`.

2 At the MATLAB prompt, type

```
sparam_filter
```

This command opens the RF Blockset Equivalent Baseband example called "Touchstone Data File for 2-Port Bandpass Filter," as shown in the following figure.



3 Double-click the General Passive Network block to display its parameters.

The **Data source** parameter is set to `Data file`, so the **Data file** parameter specifies the data file to import. The **Data file** parameter is set to `defaultbandpass.s2p`. The block uses this data with the other block parameters during simulation.

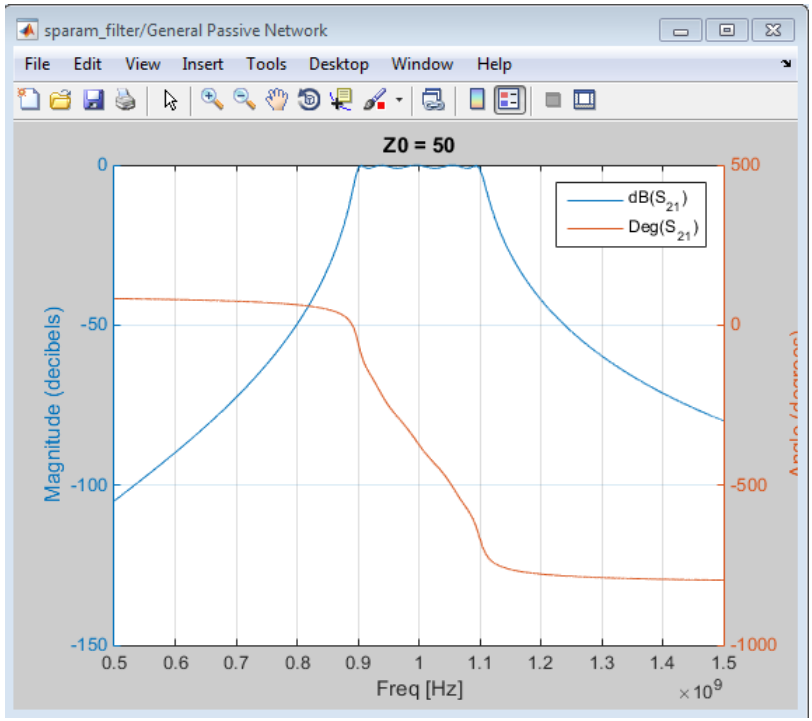
Note When the imported file contains data that is measured at frequencies other than the modeling frequencies, use the **Interpolation method** parameter to specify how the block determines the data values at the modeling frequencies. For more information, see "Determine Modeling Frequencies" on page A-3 and "Map Network Parameters to Modeling Frequencies" on page A-5.

Validate the Passive Component

In this part of the example, you plot the network parameters of the General Passive Network block to validate the data you imported in "Import Data into a General Passive Network Block" on page 6-10.

1. Open the General Passive Network block dialog box, and select the **Visualization** tab.
2. Set the **Source of frequency data** parameter to `User-specified`.
3. Set the **Frequency data (Hz)** parameter to `[0.5e9:0.1e6:1.5e9]`.
4. Click **Plot**.

These actions create a plot of the magnitude and phase of S_{21} as a function of frequency.



S_{21} versus Frequency for the Imported Data

Import Circuits from the MATLAB Workspace

You can only connect Physical library blocks in cascade. However, the blockset works with RF Toolbox software to let you include additional circuit topologies in an RF model. To model circuit topologies that contain other types of connections, you must define a circuit in the MATLAB workspace and import it into an RF model.

To import a circuit from the MATLAB workspace:

1. Define the circuit object in the MATLAB workspace using the RF Toolbox functions.

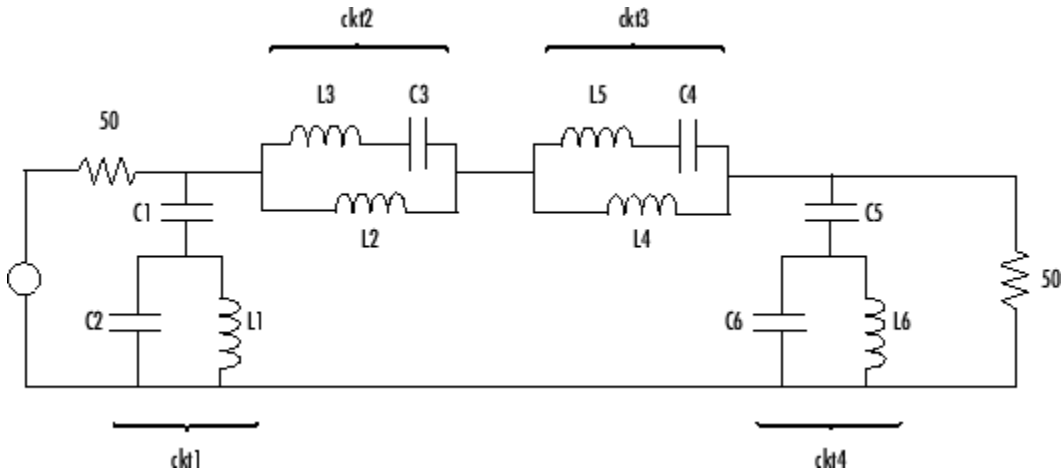
   For more information about RF circuit objects, see the RF Toolbox documentation for "RF Circuit Objects" (RF Toolbox).
2. Add a General Circuit Element block to your RF model from the Black Box Elements sublibrary of the Physical library. For information on how to open this library, see "Open RF Blockset Equivalent Baseband Libraries".
3. Enter the circuit object name in the **RFCKT object** parameter in the General Circuit Element block dialog box.

This procedure is illustrated by example in the following section.

Example — Import a Bandstop Filter into an RF Model

In this example, you simulate the frequency response of a filter that you model using circuit objects from the MATLAB workspace.

The filter in this example is the 50-ohm bandstop filter shown in the following figure.



Bandstop Filter Diagram

You represent the filter using four circuit objects that correspond to the four parts of the filter, `ckt1`, `ckt2`, `ckt3`, and `ckt4` in the diagram. You use an input signal with random,

complex input values that have a Gaussian distribution to stimulate the filter. The scope block displays the output signal.

This example illustrates how to perform the following tasks:

- “Create Circuit Objects in the MATLAB Workspace” on page 6-15
- “Build the Model” on page 6-16
- “Specify and Import Component Data” on page 6-17
- “Run the Simulation and Plot the Results” on page 6-19

Create Circuit Objects in the MATLAB Workspace

In this part of the example, you define MATLAB variables to represent the physical properties of the filter shown in the previous figure, “Bandstop Filter Diagram” on page 6-14, and use functions from RF Toolbox software to create RF circuit objects that model the filter components.

1 Type the following at the MATLAB prompt to define the filter's capacitance and inductance values in the MATLAB workspace:

```
C1 = 1.734e-12;
C2 = 4.394e-12;
C3 = 7.079e-12;
C4 = 7.532e-12;
C5 = 1.734e-12;
C6 = 4.394e-12;
L1 = 25.70e-9;
L2 = 3.760e-9;
L3 = 17.97e-9;
L4 = 3.775e-9;
L5 = 17.63e-9;
L6 = 25.70e-9;
```

2 Type the following at the MATLAB prompt to create RF circuit objects that model the components labeled `ckt1`, `ckt2`, `ckt3`, and `ckt4` in the circuit diagram:

```
ckt1 = ...
    rfckt.series('Ckts',{rfckt.shuntrlc('C',C1),...
    rfckt.shuntrlc('L',L1,'C',C2)});
ckt2 = ...
    rfckt.parallel('Ckts',{rfckt.seriesrlc('L',L2),...
```

```
    rfckt.seriesrlc('L',L3,'C',C3)});
ckt3 = ...
    rfckt.parallel('Ckts',{rfckt.seriesrlc('L',L4),...
    rfckt.seriesrlc('L',L5,'C',C4)});
ckt4 = ...
    rfckt.series('Ckts',{rfckt.shuntrlc('C',C5),...
    rfckt.shuntrlc('L',L6,'C',C6)});
```

For more information about the RF Toolbox objects used in this example, see the `rfckt.series` , `rfckt.parallel`, `rfckt.shuntrlc`, and `rfckt.seriesrlc`, object reference pages in the RF Toolbox documentation.

Build the Model

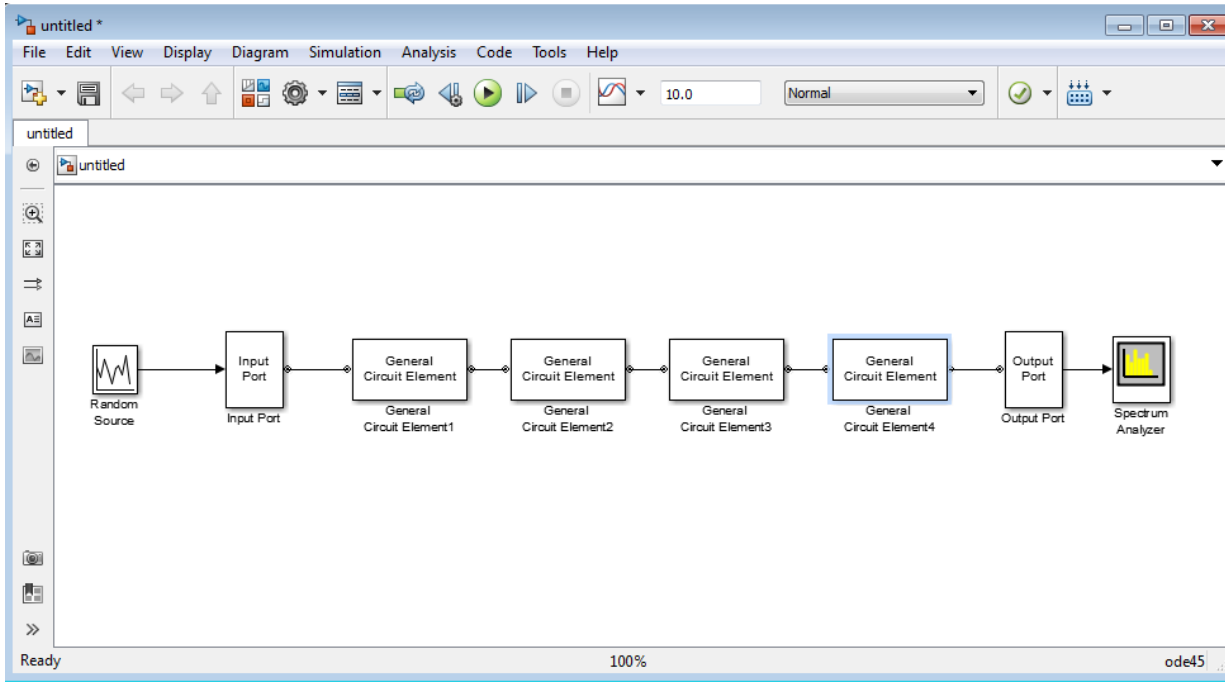
In this portion of the example, you create a Simulink model. For more information about adding and connecting components, see “Model RF Components” on page 6-2.

1. Create a new model.
2. Add to the model the blocks shown in the following table. The Library column of the table specifies the hierarchical path to each block.

Block	Library	Quantity
Random Source	**DSP System Toolbox > Sources**	1
Input Port	**RF Blockset > Equivalent Baseband > Input/Output Ports**	1
General Circuit Element	**RF Blockset > Equivalent Baseband > Black Box Elements**	4
Output Port	**RF Blockset > Equivalent Baseband > Input/Output Ports**	1
Spectrum Analyzer	**DSP System Toolbox > Sinks**	1

3. Connect the blocks as shown in the following figure.

   Change the names of your General Circuit Element blocks to match those in the figure by double-clicking the text below the block and typing a new name.



Specify and Import Component Data

In this portion of the example, you specify block parameters. To open the parameter dialog box for each block, double-click the block.

1. In the Random Source block dialog box:

   - Set the **Source type** parameter to `Gaussian`.
   - Set the **Sample time** parameter to `1/100e6`.
   - Set the **Samples per frame** parameter to `256`.
   - Set the **Complexity** parameter to `Complex`.

   Selecting these settings creates an input signal with random, complex input values that have a Gaussian distribution.

2. In the Input Port block dialog box:

   - Set the **Treat input Simulink signal as** parameter to `Incident power wave`.

- Set the **Finite impulse response filter length** parameter to `256`.
- Set the **Center frequency (Hz)** parameter to `400e6`.
- Set the **Sample time** parameter to `1/100e6`.
- Clear the **Add noise** check box.

Selecting these settings defines the physical characteristics and modeling bandwidth of the filter.

3. Set the parameters of the General Circuit Element blocks as follows:

   - In the General Circuit Element1 block dialog box, set the **RFCKT object** parameter to `ckt1`.
   - In the General Circuit Element2 block dialog box, set the **RFCKT object** parameter to `ckt2`.
   - In the General Circuit Element3 block dialog box, set the **RFCKT object** parameter to `ckt3`.
   - In the General Circuit Element4 block dialog box, set the **RFCKT object** parameter to `ckt4`.

   Selecting these settings imports the circuit objects that model the filter components into the model.

4. In the Output Port block dialog box, set the **Load impedance** parameter to `50`.

5. Set the Spectrum Analyzer block parameters as follows:

   - In the **View** tab, under `Spectrum Settings` , set the **Averages** under `Trace options` to `100`.

     This parameter establishes the number of spectra that the scope averages to produce the displayed signal. You use a value of 100 because the input signal is random and you want to display the average filter response over a large number of input values.
   - In the **View** tab, under `Spectrum Settings` , set the **Units** under `Trace options` to `dBm/Hertz`.
   - In the **View** tab, under `Configuration Properties`, set the **Minimum Y-limit** parameter to `-75` and the **Maximum Y-limit** parameter to `-45`.

     These values set the range of *x*- and *y*-values on the display such that the entire signal is visible when you run the simulation.

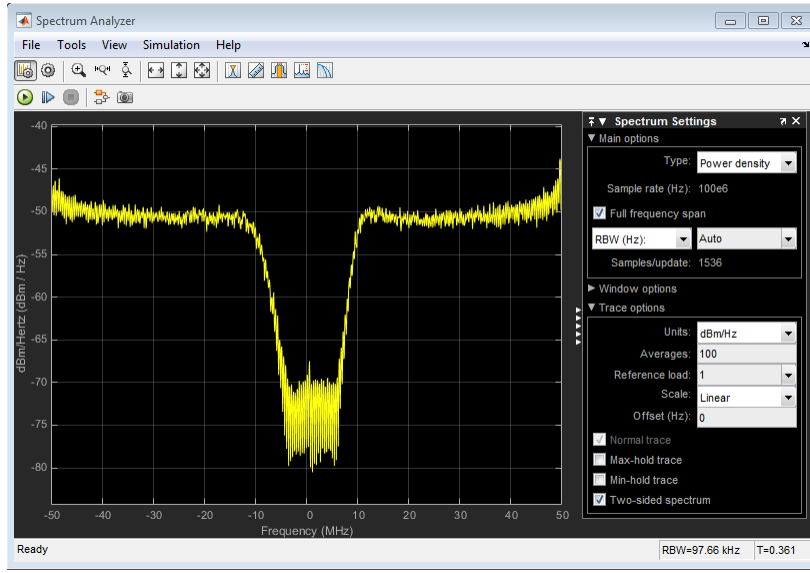
- Set the **Y-axis label** parameter to `dBm/Hertz`.

Run the Simulation and Plot the Results

In this part of the example, you run the simulation and examine the frequency response of the filter.

Select **Simulation** > **Start** in the model window to start the simulation.

The Spectrum Scope window appears automatically and displays the following plot, which shows the frequency response of the filter.



Frequency Response of Bandstop Filter

The Spectrum Scope block displays the frequency response at the shifted (baseband-equivalent) frequencies, not at the selected passband frequencies. You can relabel the *x*-axis of the Spectrum Scope window to display the passband signal by entering the **Center frequency** parameter value of `400e6` (from the Input Port block) for the **Frequency display offset (Hz)** parameter in the **Axis Properties** tab of the Spectrum Scope block. For more information on complex-baseband modeling, see “Create Complex Baseband-Equivalent Model” on page A-13.

References

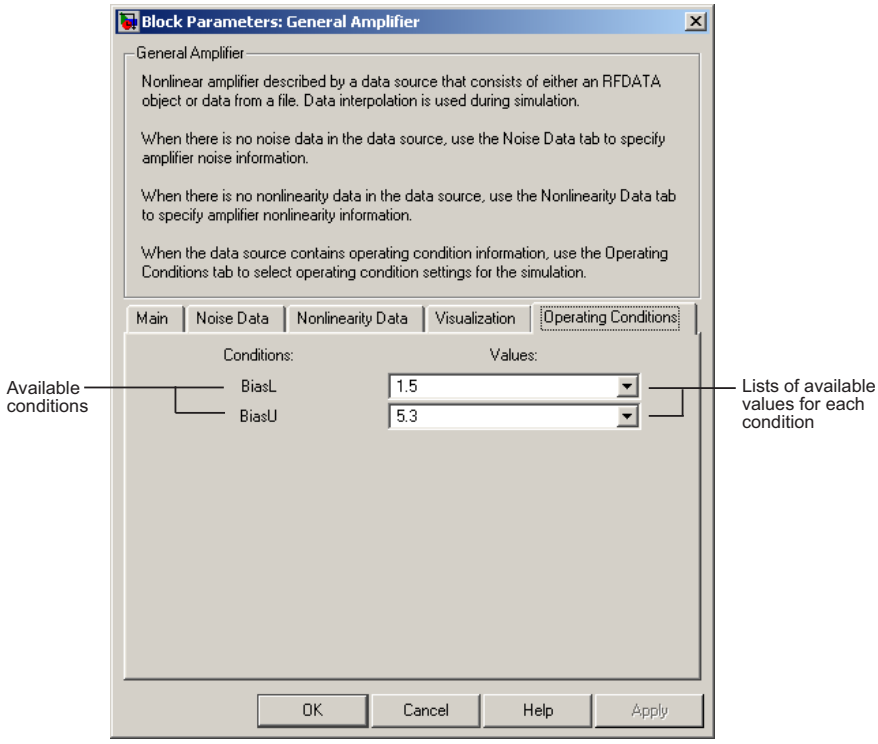
Geffe, P.R., “Novel designs for elliptic bandstop filters,” *RF Design*, February 1999.

Specify Operating Conditions

Agilent P2D and S2D files contain simulation results at one or more operating conditions. Operating conditions define the independent parameter settings that are used when creating the file data. The specified conditions differ from file to file.

When you import component data from a `.p2d` or `.s2d` file into a General Amplifier or General Mixer block, the block contains parameter values for several operating conditions. The available conditions depend on the data in the file. By default, the blockset defines the object behavior using the property values that correspond to the operating conditions that appear first in the file. To use other property values, you must select a different operating condition in the block dialog box.

If the block contains data at multiple operating conditions, the **Operating Conditions** tab contains two columns. The **Conditions** column shows the available conditions, and the **Values** column contains a drop-down list of the available values for the corresponding condition.



Block Dialog Box Showing Operating Conditions

To specify the operating condition values for a simulation:

1. Double-click the block to open the block dialog box.
2. Select the **Operating Conditions** tab.
3. In the **Conditions** column, find the condition to specify. Select the corresponding pull-down list in the **Values** column, and choose the desired operating condition value.

Repeat the preceding step as needed to specify the desired operating condition values.

Model Nonlinearity

In this section...

Amplifier and Mixer Nonlinearity Specifications

You define nonlinearity for the physical amplifier and mixer blocks at one or more frequency points through one of the following specifications:

- Power data, consisting of output power as a function of input power, imported into the block.
- Third-order intercept data, with or without power parameters, in the block dialog box. The available power parameters are gain compression power (defined as the ratio of output power to input power at small input power) and output saturation power.

The following table summarizes the nonlinearity specification options for each type of physical amplifier and mixer block.

Block	Nonlinearity Specification
General Amplifier	You can choose either of the following specifications: Power data (using a P2D, S2D, or AMP data file) or Third-order intercept data or one or more power parameters, in the block dialog box.
S-Parameters Amplifier Y-Parameters Amplifier Z-Parameters Amplifier	Third-order intercept data or one or more power parameters, in the block dialog box.
General Mixer	You can choose either of the following specifications: Power data (using a P2D, S2D, or AMP data file) or Third-order intercept data or one or more power parameters, in the block dialog box.

Block	Nonlinearity Specification
S-Parameters Mixer Y-Parameters Mixer Z-Parameters Mixer	Third-order intercept data or one or more power parameters, in the block dialog box.

Add Nonlinearity to Your System

To simulate the nonlinearity of an amplifier or mixer, you must specify or import nonlinearity data at one or more frequency points into the block.

The method you use to add nonlinearity data to a block depends on whether you specify the data manually or import the data into a block.

The following table provides instructions for adding nonlinearity data.

Nonlinearity Specification	Instructions
IP3	In the **Nonlinearity Data** tab of the block dialog box: • Set the **IP3 type** parameter to `IIP3` or `OIP3`. • Enter input third-order intercept values at one or more frequency points in the **IP3 (dBm)** parameter. • Enter corresponding frequency values in the **Frequency (Hz)** parameter.

Nonlinearity Specification	Instructions
Power parameters	Enter the gain compression power in the **1 dB gain compression power (dBm)** parameter or the saturation power in the **Output saturation power (dBm)** parameter. If you choose a scalar value for the **Frequency (Hz)** parameter, then you must also use scalar values for the power parameters. If you choose a vector value for the **Frequency (Hz)** parameter, then you can use either scalar or vector values for the power parameters.
Power data (from a file)	Import file data that includes power information into the **Data file** or **RFCKT object** parameter of the General Amplifier or General Mixer block.

Note If you import file data with no power information into a General Amplifier or General Mixer block, the **Nonlinearity Data** tab lets you add nonlinearity data manually in the block dialog box.

For information on how the blockset simulates nonlinearity data of an amplifier or mixer, see the block reference page.

Model Noise

In this section...

Amplifier and Mixer Noise Specifications

You only need to specify noise information for the physical amplifier and mixer blocks that generate noise other than resistor noise. For the other blocks, the blockset calculates the noise automatically based on the resistor values.

You define noise for the physical amplifier and mixer blocks through one of the following specifications:

- Spot noise data in the data source.
- Spot noise data in the block dialog box.
- Spot noise data (`rfdata.noise`) object in the block dialog box.
- Frequency-independent noise figure, noise factor, or noise temperature value in the block dialog box.
- Frequency-dependent noise figure data (`rfdata.nf`) object in the block dialog box.

The following table summarizes the noise specification options for each type of physical amplifier and mixer block.

Block	Noise Specification
General Amplifier	Spot noise data (using a Touchstone, P2D, S2D, or AMP data file) **OR** Spot noise data, noise figure value, noise factor value, noise temperature value, `rfdata.noise`, or `rfdata.nf` object in the block dialog box

Block	Noise Specification
S-Parameters Amplifier Y-Parameters Amplifier Z-Parameters Amplifier	Spot noise data, noise figure value, noise factor value, noise temperature value, `rfdata.noise`, or `rfdata.nf` object in the block dialog box
General Mixer	Spot noise data (using a Touchstone, P2D, S2D, or AMP data file) **OR** Spot noise data, noise figure value, noise factor value, noise temperature value, `rfdata.noise`, or `rfdata.nf` object in the block dialog box
S-Parameters Mixer Y-Parameters Mixer Z-Parameters Mixer	Spot noise data, noise figure value, noise factor value, noise temperature value, `rfdata.noise`, or `rfdata.nf` object in the block dialog box

Add Noise to Your System

To simulate the noise of a physical subsystem, you perform the following tasks:

- “Specify or Import Noise Data” on page 6-27
- “Add Noise to the Simulation” on page 6-29

Specify or Import Noise Data

The method you use to add noise data to a block depends on whether you are specifying noise data manually or importing spot-noise data.

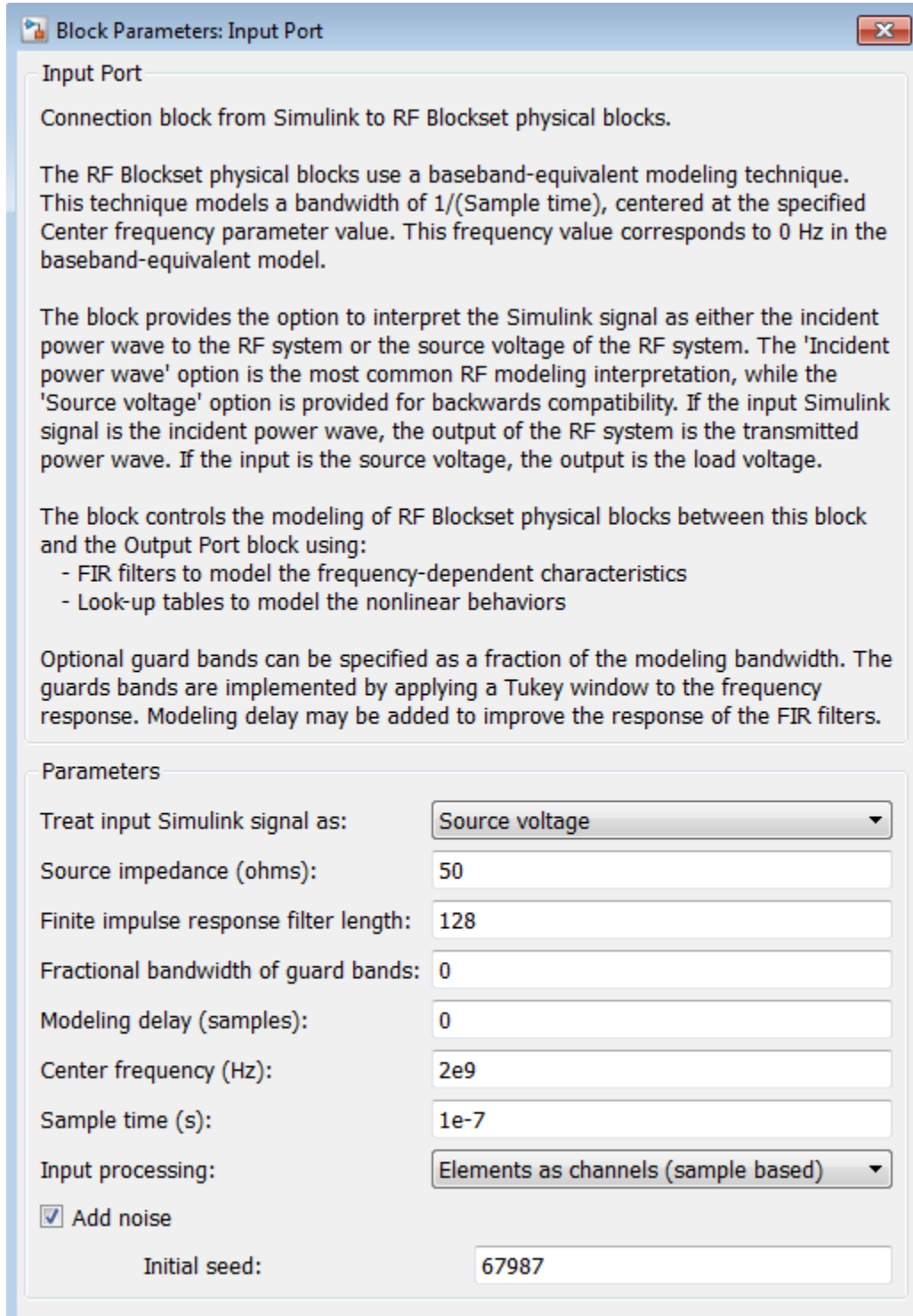
The following table provides instructions for adding noise data.

Noise Specification	Instructions
Frequency-independent noise figure	In the **Noise Data** tab of the block dialog box, set the **Noise type** parameter to `Noise figure`, and enter the noise figure value in the **Noise figure (dB)** parameter.
Frequency-dependent noise figure	In the **Noise Data** tab of the block dialog box, set the **Noise type** parameter to `Noise figure`, and enter the name of the `rfdata.nf` object in the **Noise figure (dB)** parameter.
Noise factor	In the **Noise Data** tab of the block dialog box, set the **Noise type** parameter to `Noise factor`, and enter the noise factor value in the **Noise factor** parameter.
Noise temperature	In the **Noise Data** tab of the block dialog box, set the **Noise type** parameter to `Noise temperature`, and enter the noise temperature value in the **Noise temperature (K)** parameter.
Spot noise data (in a block dialog box)	In the **Noise Data** tab of the block dialog box, set the **Noise type** parameter to `Spot noise data`. Enter the spot noise information in the **Minimum noise figure (dB)**, **Optimal reflection coefficient**, and **Equivalent normalized noise resistance** parameters.
Spot noise data (from a data object)	In the **Noise Data** tab of the block dialog box, set the **Noise type** parameter to `Noise figure` and enter the name of the `rfdata.noise` object in the **Noise figure (dB)** parameter.
Spot noise data (from a file)	Import file data that includes noise information into the **Data file** or **RFCKT object** parameter of the General Amplifier or General Mixer block.

Note If you import file data with no noise information into a General Amplifier or General Mixer block, the **Noise Data** tab lets you add noise data manually in the block dialog box.

Add Noise to the Simulation

To include noise in the simulation, you must select the **Add noise** check box on the Input Port block dialog box. This check box is selected by default.


Block Parameters: Input Port
Input Port
Connection block from Simulink to RF Blockset physical blocks.
The RF Blockset physical blocks use a baseband-equivalent modeling technique. This technique models a bandwidth of 1/(Sample time), centered at the specified Center frequency parameter value. This frequency value corresponds to 0 Hz in the baseband-equivalent model.
The block provides the option to interpret the Simulink signal as either the incident power wave to the RF system or the source voltage of the RF system. The 'Incident power wave' option is the most common RF modeling interpretation, while the 'Source voltage' option is provided for backwards compatibility. If the input Simulink signal is the incident power wave, the output of the RF system is the transmitted power wave. If the input is the source voltage, the output is the load voltage.
The block controls the modeling of RF Blockset physical blocks between this block and the Output Port block using:
- FIR filters to model the frequency-dependent characteristics
- Look-up tables to model the nonlinear behaviors
Optional guard bands can be specified as a fraction of the modeling bandwidth. The guards bands are implemented by applying a Tukey window to the frequency response. Modeling delay may be added to improve the response of the FIR filters.
Parameters
Treat input Simulink signal as:
Source voltage
Source impedance (ohms):
50
Finite impulse response filter length:
128
Fractional bandwidth of guard bands:
0
Modeling delay (samples):
0
Center frequency (Hz):
2e9
Sample time (s):
1e-7
Input processing:
Elements as channels (sample based)
Add noise
Initial seed:
67987

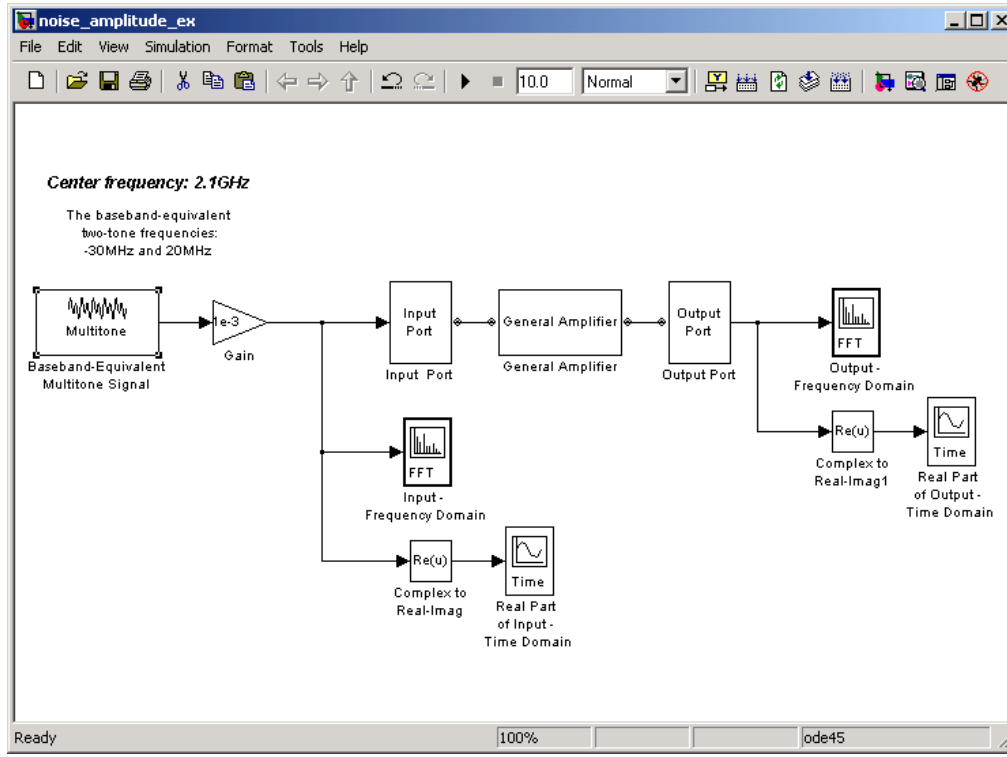
For information on how the blockset simulates noise, see “Model Noise in an RF System” on page A-7.

Plot Noise

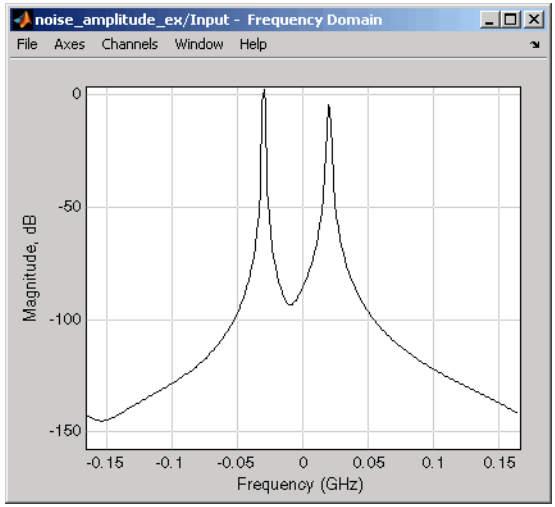
RF Blockset Equivalent Baseband software models communications systems. The noise in these systems has a very small amplitude, typically from 1e-6 to 1e-12 Watts. In contrast, the default signal power of a Communications System Toolbox modulator block is 1 Watt at a nominal 1 ohm. Therefore, the signal-to-noise ratio in an RF system simulation is large, making it difficult to view the noise that the RF system adds to your signal.

To display the noise on a plot, you might need to attenuate the signal amplitude to a value within a couple orders of magnitude of the noise.

For example, suppose you have the following model that contains a multitone test signal source.

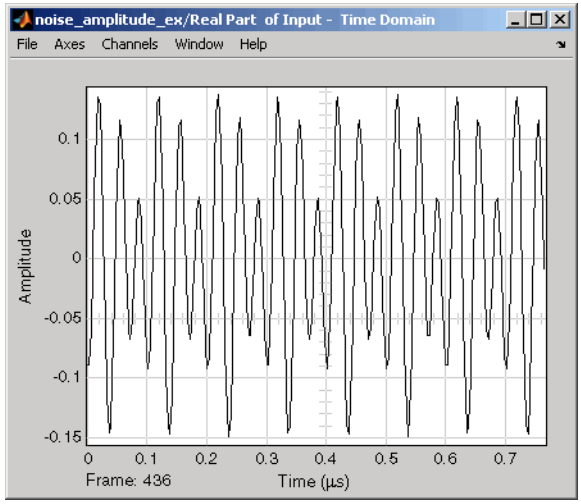


When you simulate this model, Simulink brings up several windows showing the input and output for the physical subsystem. The Input - Frequency Domain window shown in the following figure displays the input signal in the frequency domain.



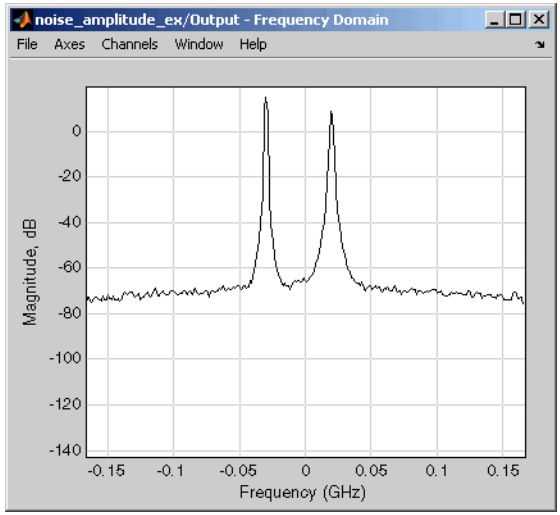
Input Signal Spectrum

The Real Part of Input - Time Domain window displays the real part of the complex-valued input signal in the time domain.



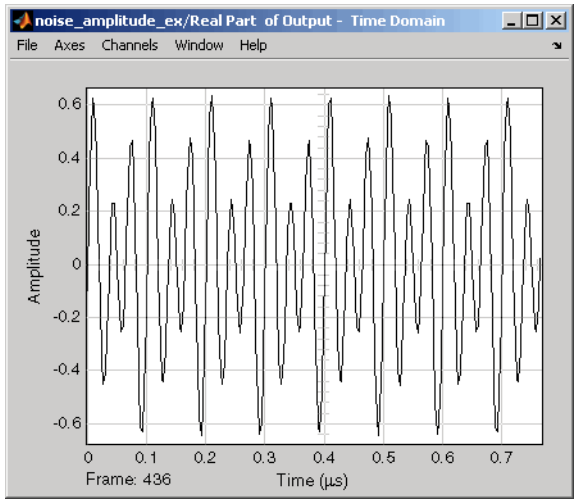
Real Part of Input Signal

In the model, the physical subsystem adds noise to the input signal. The Output - Frequency Domain window shows the noisy output signal in the frequency domain.



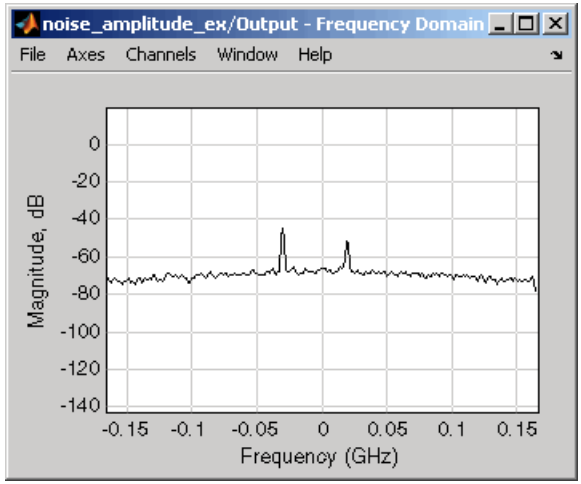
Output Signal Spectrum

The amplitude of the signal is large compared to the amplitude of the noise, so the noise is not visible in the Real Part of Output - Time Domain window that shows the real part of the time-domain output signal. Therefore, you must attenuate the amplitude of the input signal to display the noise of the time-domain output signal.



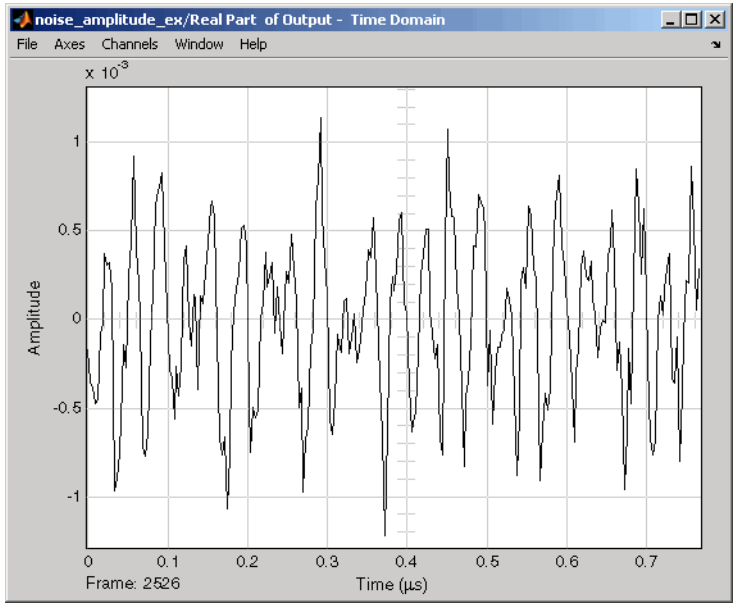
Real Part of Output Signal

Attenuate the amplitude of the input signal by setting the **Gain** parameter to `1e-3`. This is equivalent to attenuating the input signal by 60 dB. When you run the model again, the two signal peaks are not as pronounced in the Output - Frequency Domain window.



Output Signal Spectrum for Attenuated Input

You can now view the noise that the RF system adds to your signal in the Real Part of Output - Time Domain window.



Real Part of Output Signal Showing Noise

Plot Model Data

Create Plots

In this section...

Available Data for Plotting

RF Blockset Equivalent Baseband software lets you validate the behavior of individual RF components and physical subsystems in your model by plotting the following data:

- Large- and small-signal S-parameters
- Noise figure, noise factor and noise temperature
- Output third-order intercept point
- Power data
- Phase noise
- Voltage standing-wave ratio
- Transfer function
- Group delay
- Reflection coefficients

Note When you plot information about a physical block, the blockset plots the actual frequency response of the block, as specified in the block dialog box. The blockset does not plot the frequency response of the complex-baseband model that it uses to simulate the block, in which the frequency response is centered at zero.

Validate Individual Blocks and Subsystems

You can plot model data for an individual physical block or for a physical subsystem. A *subsystem* is a collection of one or more physical blocks bracketed by an Input Port block

and an Output Port block. To understand the behavior of specific subsystems, plot the data of the corresponding Output Port block after you run a simulation.

To validate the behavior of individual RF components in the model, plot the data of the corresponding physical blocks. You can plot data for individual blocks from each of these components either before or after you run a simulation.

You create a plot by selecting options in the block dialog box, as shown in “Create and Modify Subsystem Plots” on page 7-28. To learn about the available plots, see “Types of Plots” on page 7-3. For more information about creating plots, see “How to Create a Plot” on page 7-14.

Types of Plots

RF Blockset Equivalent Baseband software provides a variety of plots for analyzing the behavior of RF components and subsystems. The following table summarizes the available plots and charts and describes each one.

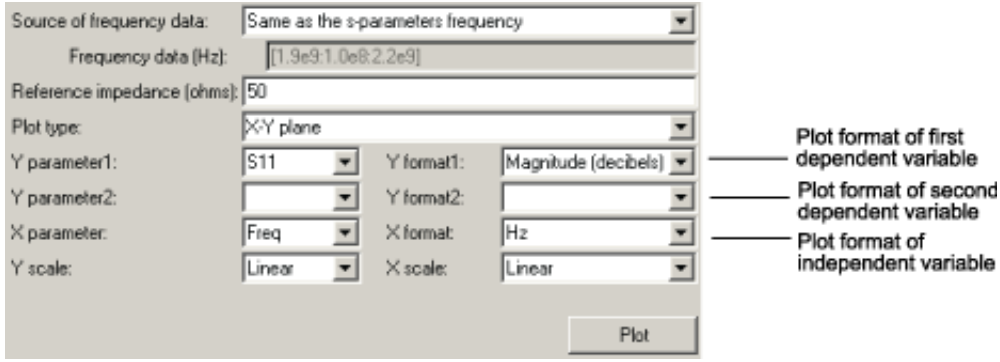
Plot Type	Plot Contents
X-Y Plane (Rectangular) Plot	Parameters as a function of frequency, input power, or operating condition, such as • S-parameters • Noise figure (NF), Noise factor (NFactor), and Noise Temperature (NTemp) • Voltage standing-wave ratio (VSWR) • Output third-order intercept point (OIP3) • Input and output reflection coefficients (GammaIn and GammaOut)

Plot Type	Plot Contents
Link Budget Plot (3-D)	Parameters as a function of frequency for each component in a physical subsystem *where* The curve for a given component represents the cumulative contribution of each RF component up to and including the parameter value of that component. For more information, see “Link Budget” on page 7-10.
Polar Plane Plot	Magnitude and phase of parameters as a function of frequency or operating condition, such as • S-parameters • Input and output reflection coefficients (GammaIn and GammaOut)
Smith® Chart	Real and imaginary parts of S-parameters as a function of frequency or operating condition, used for analyzing the reflections caused by impedance mismatch.
Composite Plot	Multiple plots and charts in one figure.

To learn how to create these plots, see “How to Create a Plot” on page 7-14.

Plot Formats

When you create a plot from a block dialog box, you must specify the format of the data for both the *x*- and *y*-axes.



These plot options define how RF Blockset Equivalent Baseband software displays the data on the plot.

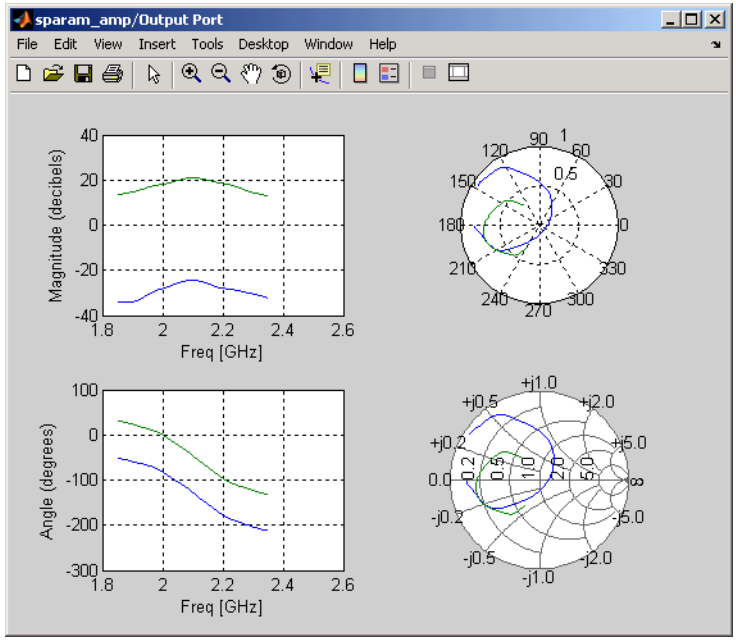
The available formats vary with the data you select to plot. The data you can plot depends on the plot type you select. The plot formats determine whether the blockset converts the data to a new set of units, or performs a calculation on the data. For example, setting the format to `Real` tells the blockset to compute and plot the real part of the parameter.

The following topics describe the available parameters and formats for each plot type:

- “Composite Data” on page 7-5
- “X-Y Plane” on page 7-7
- “Link Budget” on page 7-10
- “Polar Plane Plots and Smith Charts” on page 7-12

Composite Data

The composite data plot automatically generates four separate plots in one figure window, showing the frequency dependence of several parameters. The following figure shows an example of such a plot.



Example — Composite Data Plot

Note For composite data plots, you do not need to specify the parameters or the formats —they are set automatically.

The combination of plots differs based on the type of block and the specified block data. The following table describes the contents of the composite data plot for each specification. The Plot Contents column lists the types of plots as they appear on the composite plot, counterclockwise and starting in the upper-left corner. The blockset plots all data as a function of frequency.

Block	Specified Data	Plot Contents
General Amplifier or General Mixer	Network parameters **OR** Network parameters and noise	• X-Y plot, magnitude of S_{12} and S_{21} in decibels • X-Y plot, phase of S_{12} and S_{21} in degrees • Z Smith Chart, real and imaginary parts of S_{11} and S_{22} • Polar plot, magnitude and phase of S_{11} and S_{22}
	Network parameters and power **OR** Network parameters, noise, and power	• X-Y plot, magnitude of S_{12} and S_{21} in decibels • X-Y plot, output power (P_{out}) in dBm (decibels referenced to one milliwatt) • Z Smith Chart, real and imaginary parts of S_{11} and S_{22} • Polar plot, magnitude and phase of S_{11} and S_{22}
Other Physical block	Network parameters **OR** Network parameters and noise (S-, Y-, and Z-Parameters Amplifiers and Mixers only) **Note** Only the General Amplifier and General Mixer blocks accept power data.	• X-Y plot, magnitude of S_{12} and S_{21} in decibels • X-Y plot, phase of S_{12} and S_{21} in degrees • Z Smith Chart, real and imaginary parts of S_{11} and S_{22} • Polar plot, magnitude and phase of S_{11} and S_{22}

X-Y Plane

You can plot any parameters that are relevant to your block on an X-Y plane plot. For this type of plot, you specify data for both the *x*- and *y*-axes. If you specify two Y parameters, and you specify different formats for the two Y parameters, the blockset plots the second Y parameter on the right *y*-axis.

The following table summarizes the available Y parameters and formats. The parameters and formats are the same for both the left and right *y*-axes.

Note LS11, LS12, LS21, and LS22 are large-signal S-parameters. You can plot these parameters as a function of input power or as a function of frequency.

Y Parameter	Y Format
S11, S12, S21, S22 LS11, LS12, LS21, LS22 (General Amplifier and General Mixer blocks with multiple operating conditions only)	Magnitude (decibels) Magnitude (linear) Angle (degrees) Angle (radians) Real Imaginary
NF	Magnitude (decibels)
NFactor	None This format tells the blockset to plot the noise factor as it is specified to or calculated by the block.
NTemp	Kelvin
OIP3	dBm dBW W mW
VSWRIn, VSWROut	Magnitude (decibels) None This format tells the blockset to plot the voltage standing-wave ratio as it is specified to or calculated by the block.
Pout (General Amplifier and General Mixer blocks with power data only)	dBm dBW W mW
Phase (General Amplifier and General Mixer blocks with power data only)	Angle (degrees) Angle (radians)

Y Parameter	Y Format

`AM/AM` (General Amplifier and General Mixer blocks with power data only)	`Magnitude (decibels)`
	`None`
	This format tells the blockset to plot the AM/AM conversion as it is specified to or calculated by the block.

`AM/PM` (General Amplifier and General Mixer blocks with power data only)	`Angle (degrees)` `Angle (radians)`
`PhaseNoise` (Mixer blocks only)	`dBc/Hz`

`FMIN` (Amplifier and Mixer blocks with spot noise data only)	`Magnitude (decibels)`
	`None`
	This format tells the blockset to plot the minimum noise figure as it is specified to or calculated by the block.

`GammaIn`, `GammaOut`(Output Port block only)	`Magnitude (decibels)`
	`Magnitude (linear)`
	`Angle (degrees)`
	`Angle (radians)`
	`Real`
	`Imaginary`

`GAMMAOPT` (Amplifier and Mixer blocks with spot noise data only)	`Magnitude (decibels)`
	`Magnitude (linear)`
	`Angle (degrees)`
	`Angle (radians)`
	`Real`
	`Imaginary`

`RN` (Amplifier and Mixer blocks with spot noise data only)	`None` This format tells the blockset to plot the noise resistance as it is specified to or calculated by the block.

The available X parameters depend on the Y parameters you select. The following table summarizes the available X parameters for each of the Y parameters in the preceding table.

Y Parameter	X Parameter
`Pout`, `Phase`, `LS11`, `LS12`, `LS21`, `LS22`	`Pin` `Freq`
`S11`, `S12`, `S21`, `S22`, `NF`, `OIP3`, `VSWRIn`, `VSWROut`, `GAMMAIn`, `GAMMAOut`, `FMIN`, `GAMMAOPT`, `RN`	`Freq`
`AM/AM`, `AM/PM`	`AM`

The following table shows the X formats that are available for the X parameters listed in the preceding table.

X Parameter	X Format

`Pin`	`dBm`
	`dBW`
	`W`
	`mW`

`Freq`	`THz`
	`GHz`
	`MHz`
	`KHz`
	`Hz`
	`Auto` (`xformat` is chosen to provide the best scaling for the given `xparameter` values.)

`AM`	`Magnitude (dB)` `Magnitude (linear)`

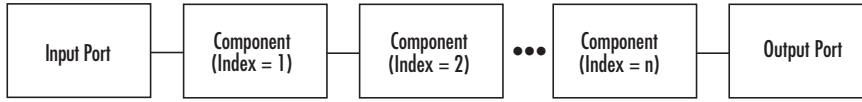
When you import block data from a `.p2d` or `.s2d` file, you can also plot Y parameters as a function of any operating condition from the file that has numeric values, such as bias. You can specify an operating condition as the X parameter only when validating individual blocks, and the format is always `None`. This format tells the blockset to plot the operating condition values as they are specified in the file.

Link Budget

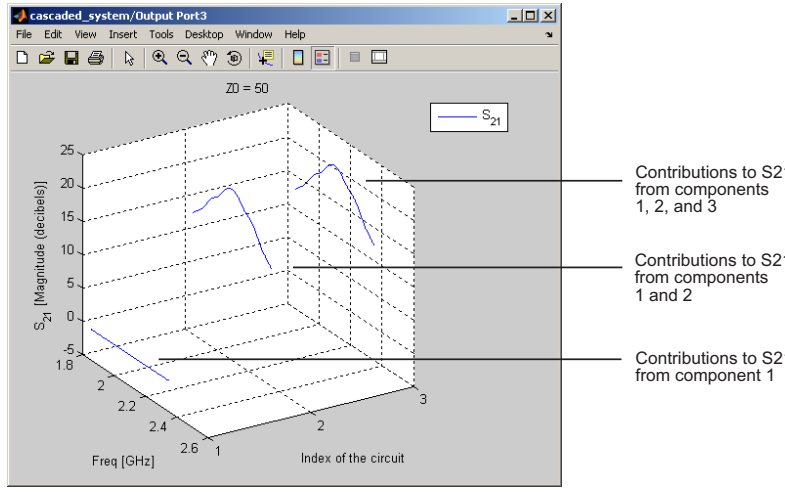
You use the `Link budget` plot to understand the individual contribution of each block to a plotted Y parameter value in a physical subsystem with multiple components between the Input Port and the Output Port blocks.

The link budget plot is a three-dimensional plot that shows one or more curves of parameter values as a function of frequency, ordered by the subsystem circuit index.

The following figure shows how the circuit index is assigned to a component in a physical subsystem based on its sequential position in the subsystem.



A curve on the link budget plot for each circuit index represents the contributions to the parameter value of the RF components up to that index. The following figure shows an example of a link budget plot.



Example — Link Budget Plot

The following table summarizes the Y parameters and formats that are available for a link budget plot.

Y Parameter	Y Format

S11, S12, S21, S22	`Magnitude (decibels)`
	`Magnitude (linear)`
	`Angle (degrees)`
	`Real`
	`Imaginary`

`OIP3`	`dBm`
	`dBW`
	`W`
	`mW`

`NF`	`Magnitude (decibels)` `Magnitude (linear)`
`NFactor`	`None` This format tells the blockset to plot the noise factor as it is specified to the block.
`NTemp`	`Kelvin`

If you specify two Y parameters, the blockset puts both parameters in a single plot. The Y parameters must have the same formats.

For a link budget plot, the X parameter is always `Freq`. The format of the X parameter specifies the units of the *x*-axis.

Polar Plane Plots and Smith Charts

You can use RF Blockset Equivalent Baseband software to generate Polar plots and Smith Charts. When you select these plot types, you do not need to specify the format of any Y parameters—the formats are set automatically. If you specify two Y parameters, the blockset puts both parameters in a single plot.

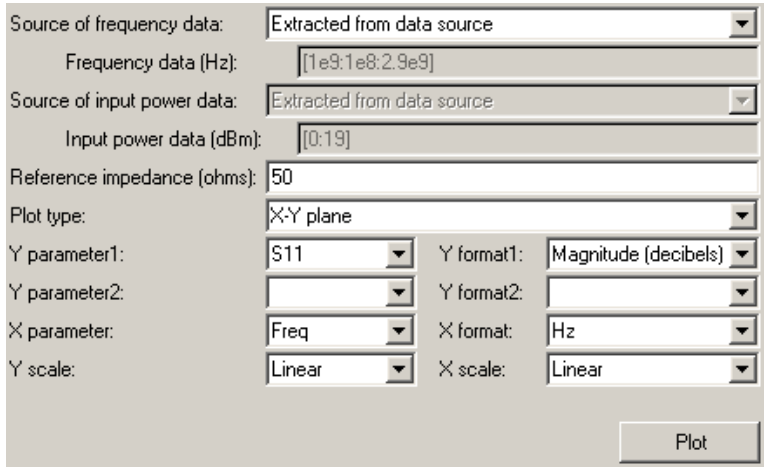
The following table describes the Polar plot and Smith Chart options. It also lists the available Y parameters.

Plot Type	Y Parameter
`Polar plane`	`S11`, `S12`, `S21`, `S22` `LS11`, `LS12`, `LS21`, `LS22` (General Amplifier and General Mixer blocks with data from a P2D file only) `GammaIn`, `GammaOut` (Output Port block only)
`Z Smith chart`	`S11`, `S22` `LS11`, `LS22` (General Amplifier and General Mixer blocks with data from a P2D file only) `GammaIn`, `GammaOut` (Output Port block only)
`Y Smith chart`	`S11`, `S22` `LS11`, `LS22` (General Amplifier and General Mixer blocks with data from a P2D file only) `GammaIn`, `GammaOut` (Output Port block only)
`ZY Smith chart`	`S11`, `S22` `LS11`, `LS22` (General Amplifier and General Mixer blocks with data from a P2D file only) `GammaIn`, `GammaOut` (Output Port block only)

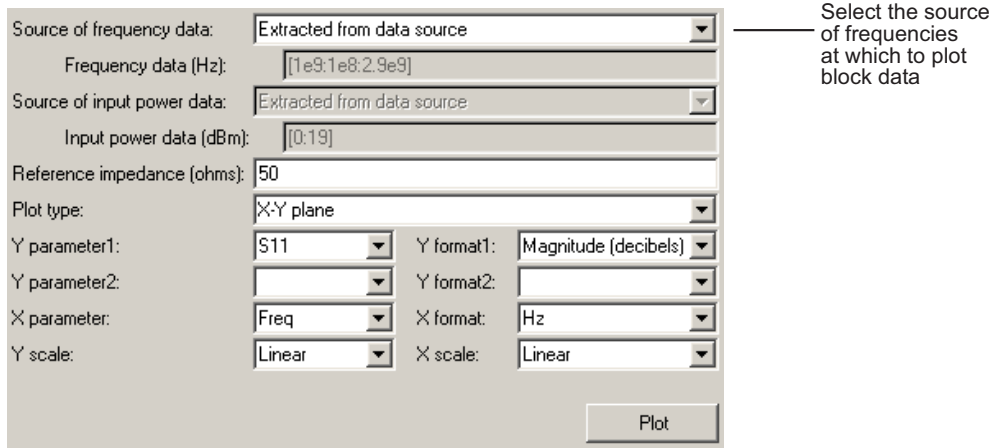
By default, the X parameter is `Freq`. The format of the X parameter specifies the units of the *x*-axis. When you import block data from a `.p2d` or `.s2d` file, you can also plot Y parameters as a function of any operating condition from the file that has numeric values, such as bias. You can specify an operating condition as the X parameter only when validating individual blocks, and the format is always `None`.

How to Create a Plot

1 Double-click the block to open the block dialog box, and select the **Visualization** tab. The following figure shows the contents of the tab.



2 Select the **Source of frequency data**.

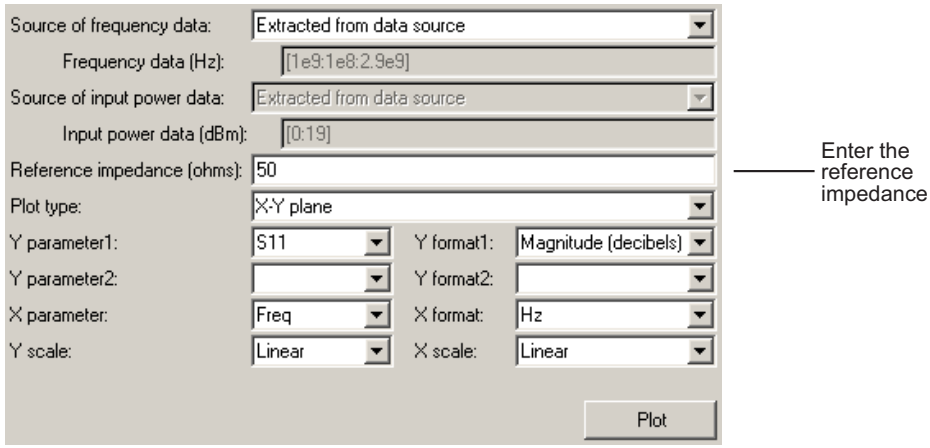


This value is the source of the frequency values at which to plot block data. The following table summarizes the available types of sources for the various types of blocks.

Source of frequency data	Description	Blocks
`User-specified`	Vector of frequencies that you enter. When you select `User-specified` in the **Source of frequency data** list, the **Frequency range (Hz)** field is displayed. Enter a vector specifying the range of frequencies you want to plot. For example, to plot block data from 0.3 MHz to 5 GHz by 0.1 MHz, enter `[0.3e6:0.1e6:5e9]`. **Note** When you select **PhaseNoise** in the **Parameter** list and `User-specified` in the **Source of frequency data** list, the **Frequency range (Hz)** field is disabled. You use the **Phase noise frequency offset (Hz)** block parameter to specify the frequency values at which to plot block data.	All physical blocks
`Derived from Input Port parameters` (Available after running a simulation or clicking the **Update Diagram** button )	Modeling frequencies derived from the Input Port block parameters. For information on how the blockset computes the modeling frequencies, see “Determine Modeling Frequencies” on page A-3.	All physical blocks
`Same as the S-parameters`	Frequency values specified in the **Frequency** block parameter.	S-Parameters Passive Network, S-Parameters Amplifier, S-Parameters Mixer

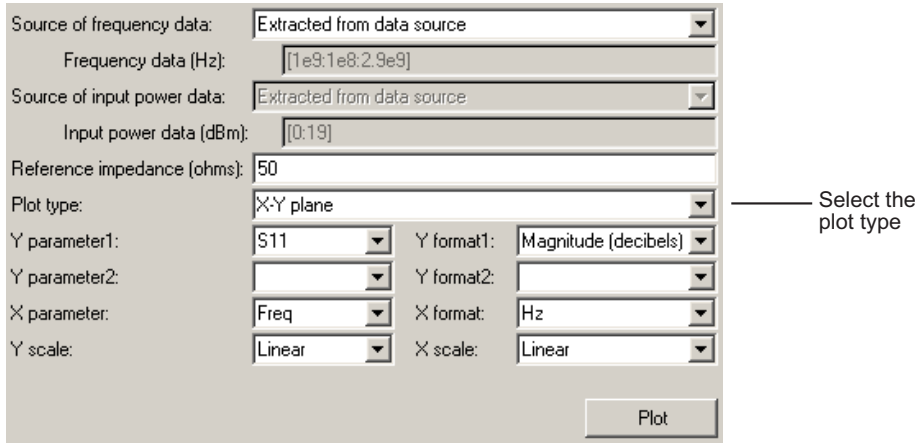
Source of frequency data	Description	Blocks
`Same as the Y-parameters`	Frequency values specified in the **Frequency** block parameter.	Y-Parameters Passive Network, Y-Parameters Amplifier, Y-Parameters Mixer
`Same as the Z-parameters`	Frequency values specified in the **Frequency** block parameter.	Z-Parameters Passive Network, Z-Parameters Amplifier, Z-Parameters Mixer
`Extracted from data source`	Frequency values imported into the **Data file** or **RFDATA object** block parameter.	General Passive Network, General Amplifier, and General Mixer

3 Enter the **Reference impedance**.



This value is the reference impedance to use when plotting small-signal parameters.

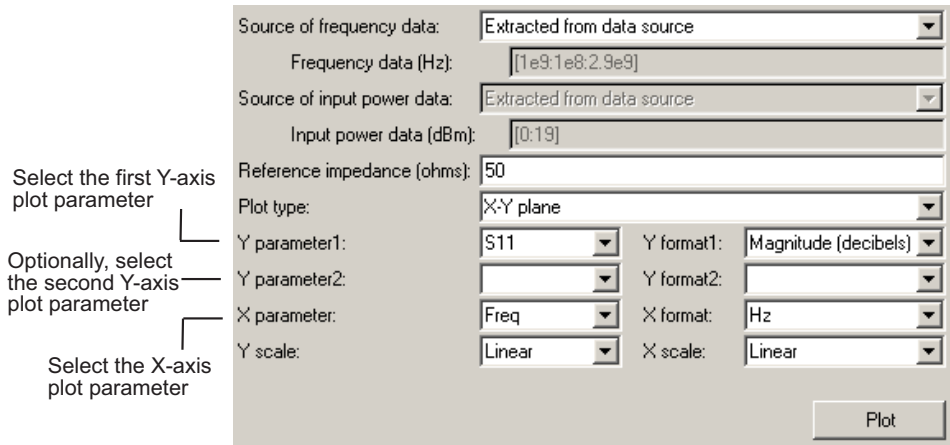
4 Select the **Plot type**.



This value is the type of plot. For a description of the options, see “Types of Plots” on page 7-3.

5 Select the following parameters:

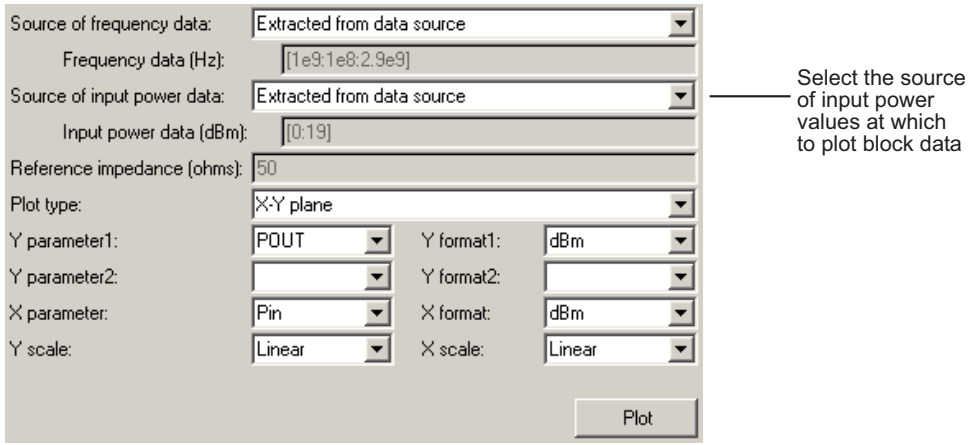
- **Y Parameter1** — The first parameter for the Y-axis.
- **Y Parameter2** — The second parameter for the Y-axis (optional).
- **X Parameter** — The parameter for the X-axis.



These parameters specify the data to be plotted. The available choices vary with the type of plot. For a description of the options for a particular plot type, see the topic on that plot type in “Plot Formats” on page 7-4.

6 If you select a large-signal parameter for one or more *y*-axis parameters, select the **Source of power data**.

Note Large-signal parameters are available only for General Amplifier or General Mixer blocks that contain power data.



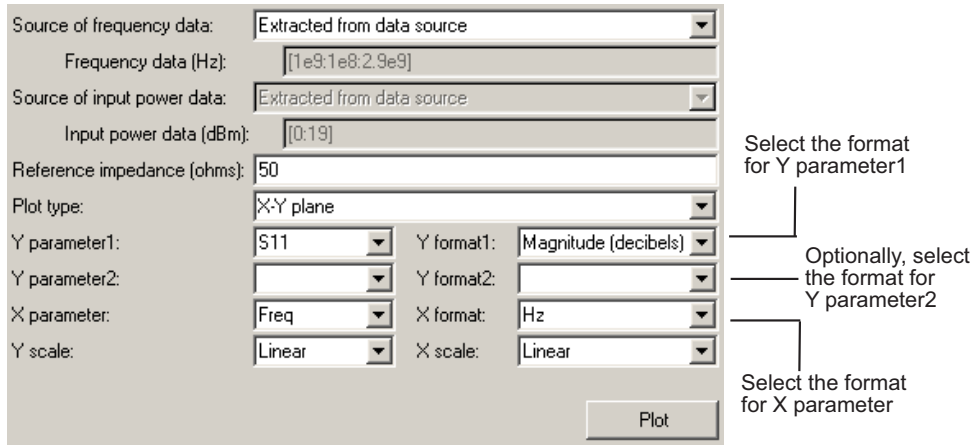
This value is the source of the input power values at which to plot block data. The following table summarizes the available types of sources for the General Amplifier and General Mixer blocks.

Source of frequency data	Description
`Extracted from data source`	Input power values imported into the **Data file** or **RFDATA object** block parameter.

Source of frequency data	Description
`User-specified`	Vector of power values that you enter. When you select `User-specified` in the **Source of power data** list, the **Input power data (dBm)** field is displayed. Enter a vector specifying the range of power values you want to plot. For example, to plot block data from 1 dBm to 10 dBm by 2 dBm, enter `[1:2:10]`.

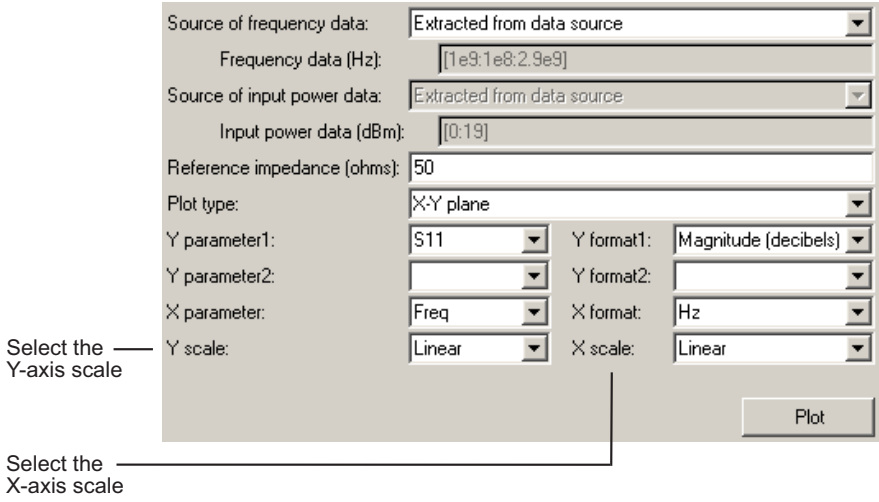
7 Select the following formats:

- **Y Format1** — The format for the first Y parameter.
- **Y Format2** — The format for the second Y parameter (optional).
- **X Format** — The format for the X parameter.



These are the X and Y formats for plotting the selected parameter. The available choices vary based on the selected parameter. For a description of the options for a particular plot type, see the topic on that plot type in “Plot Formats” on page 7-4.

8 Select the **X Scale** and **Y Scale**.



These are the scales on which to plot the data. The available choices are `Linear` and `Log`.

9 Click **Plot**.

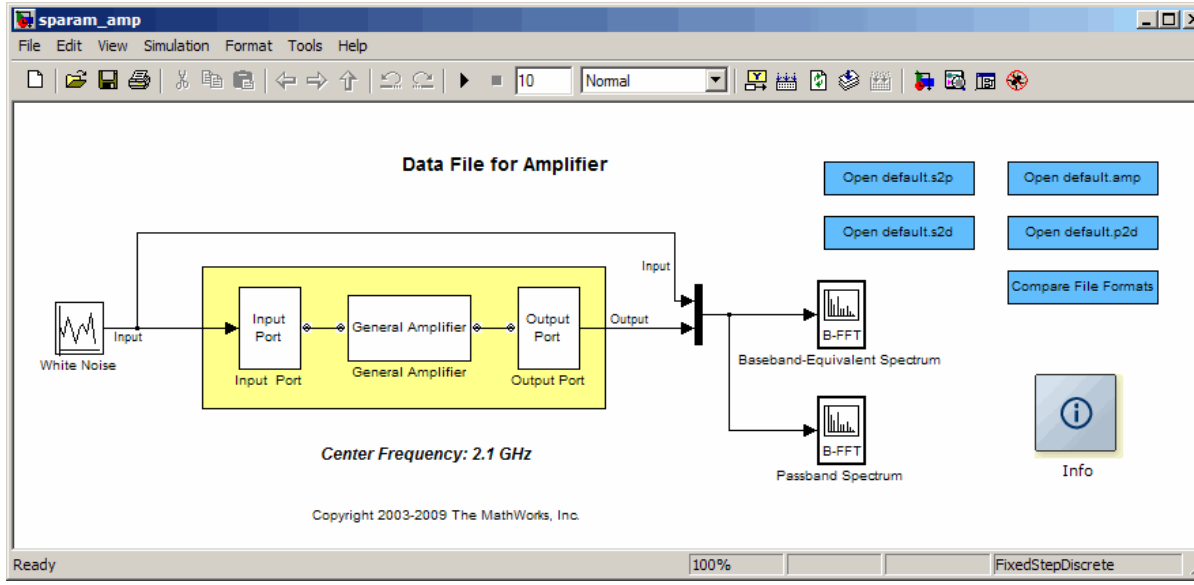
Note By default, the blockset does not add a legend to some plots. To display the plot legend, type `legend show` at the MATLAB prompt.

Example — Plot Component Data on a Z Smith Chart

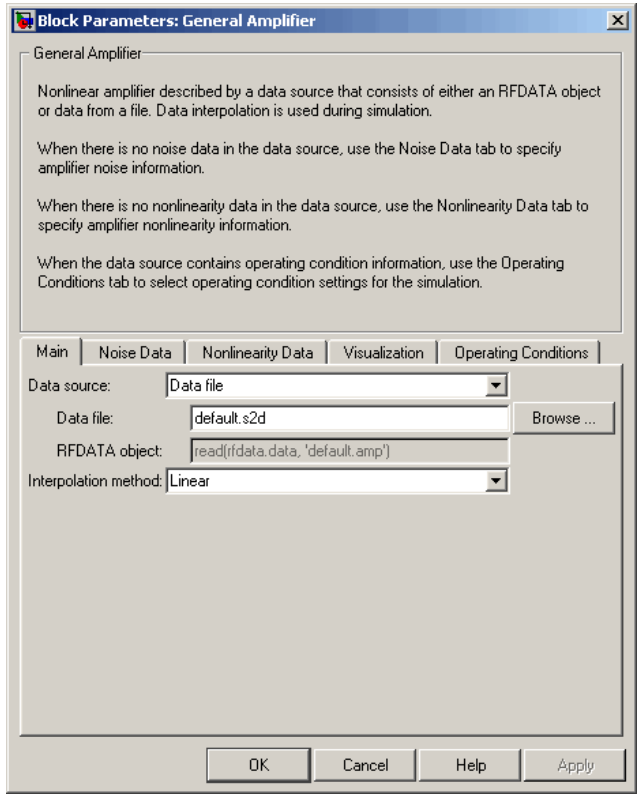
In this example, you simulate the frequency response of an amplifier using data from the `default.s2d` S2D file.

Using a RF Blockset Equivalent Baseband example model, you import the data file into a General Amplifier block and validate the amplifier by plotting the S-parameters of the block on a Z Smith Chart.

1 Type `sparam_amp` at the MATLAB prompt to open the “AMP Data File for Amplifier” example.



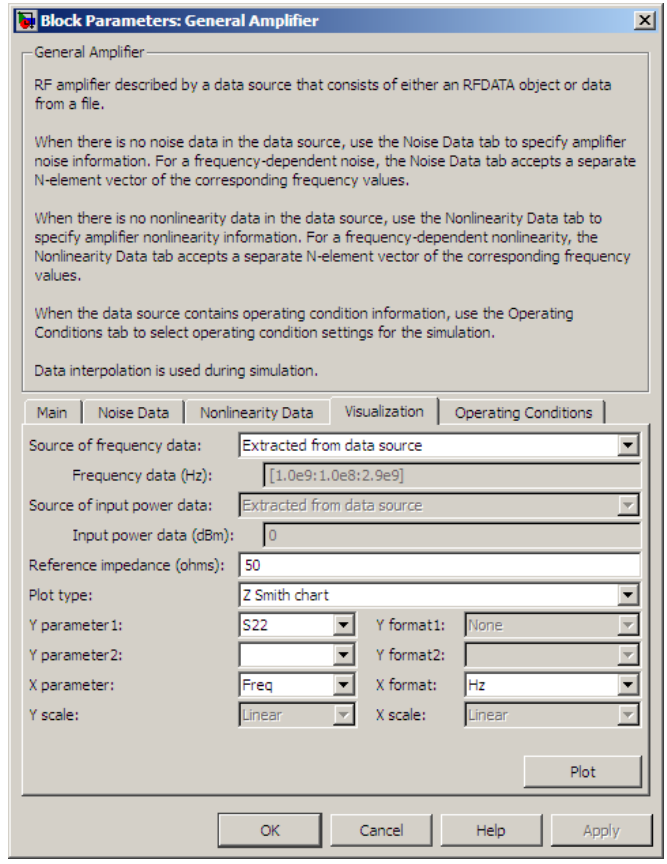
2 Double-click the General Amplifier block to display its parameters.



As shown in the preceding figure, the **Data source** parameter is set to `Data file`and the **Data file** parameter is set to `default.s2d`. These values tell the blockset to import data from the file `default.s2d`. The block uses this data, along with the other block parameters, in simulation.

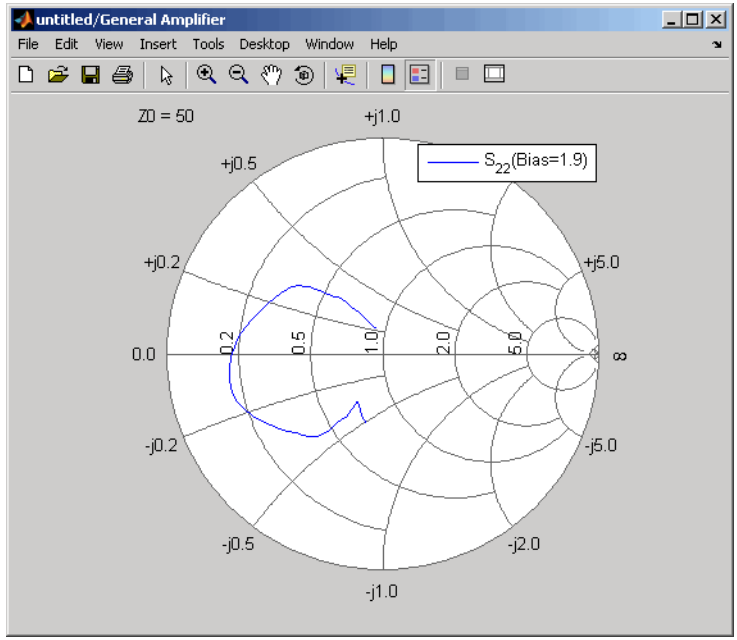
3 Select the **Visualization** tab and set the General Amplifier block parameters as follows:

- In the **Plot type** list, select `Z Smith chart`.
- In the **Y Parameter1** list, select `S22`.



4 Click **Plot**.

This action creates a Z Smith Chart of the S_{22} parameters using the frequency data from the `default.s2d` file.



General Amplifier Frequency Response

Note To display data tips for a plotted line, select **Tools > Data Cursor**. Click the data cursor on the plotted line to see the frequency and the parameter value at that point. See the MATLAB documentation for more information.

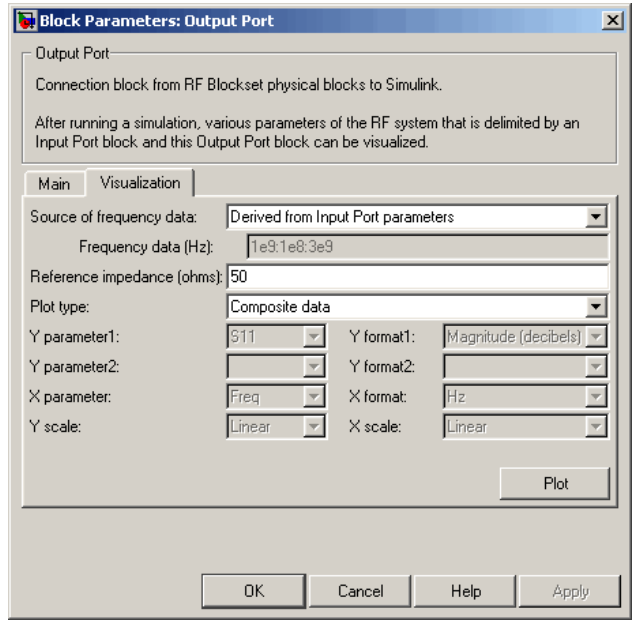
Update Plots

When you run a simulation, the blockset continues to display any open plots, but does not update the plots to reflect new simulation results. You must update the subsystem plots after the simulation to display the behavior of the revised subsystem.

When you make changes to the parameters of blocks that represent individual RF components, you need to update any open plots, because the blockset does not automatically redraw the plots.

To update an existing plot:

1. Double-click the block to open the block dialog box, and select the **Visualization** tab.



Example Block Dialog Box Showing Visualization Tab

2. Click **Plot**.

Modify Plots

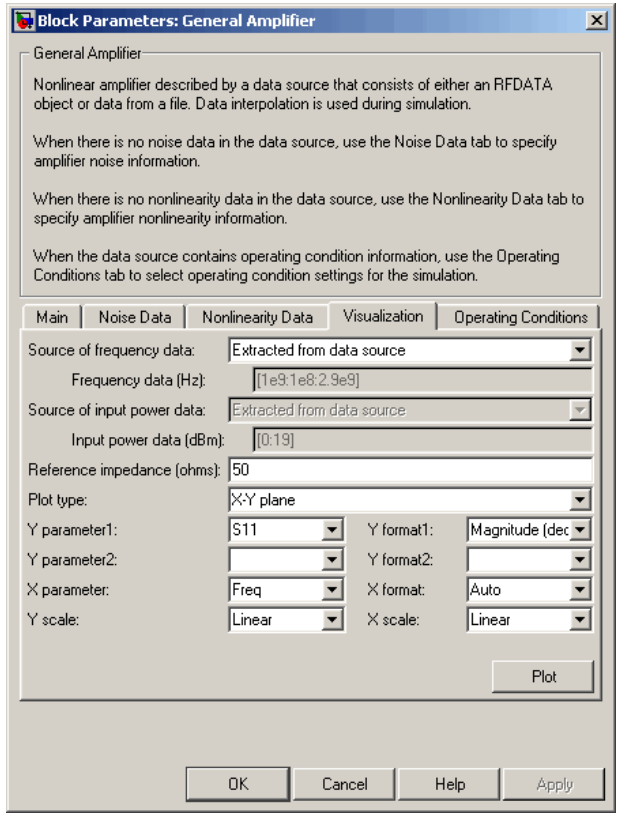
You can modify an existing plot by changing the plot options. The outcome depends on the parameter you change.

The following table summarizes the results of changing the plot options.

Block Parameter	Plot Change
Source of frequency data OR **Frequency data (Hz)**	Redraws plot using the new frequency data.
Source of power data OR **Input power data (dBm)**	Redraws plot using the new power data.
Plot type	Draws plot in a new figure using the new plot type. **Note** If the current plot options are valid for the new plot type, they retain their values. Otherwise, they revert to their default values.
Y Parameter1 OR **Y Parameter2**	If the new parameter has the same independent variable and format as the one on the plot, the blockset adds the new parameter to the existing plot. Otherwise, it redraws the plot for the new parameter and independent variable.
Y Format1 OR **Y Format2**	Redraws plot using the new format.
X Parameter	Redraws plot using the new independent variable.
X Format	Redraws plot using the new format.
X Scale	Redraws plot using the new scale.
Y Scale	Redraws plot using the new scale.

To modify a plot:

1. Double-click the block to open the block dialog box, and select the **Visualization** tab.



Example Block Dialog Box Showing Plot Parameters

2. Change the plot options.
3. Click **Plot**.

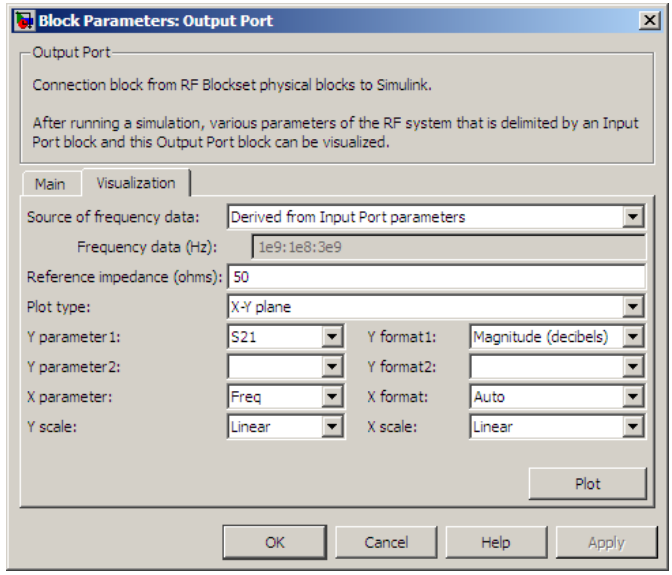
Create and Modify Subsystem Plots

In this section...

Plot the Network Parameters of a Subsystem

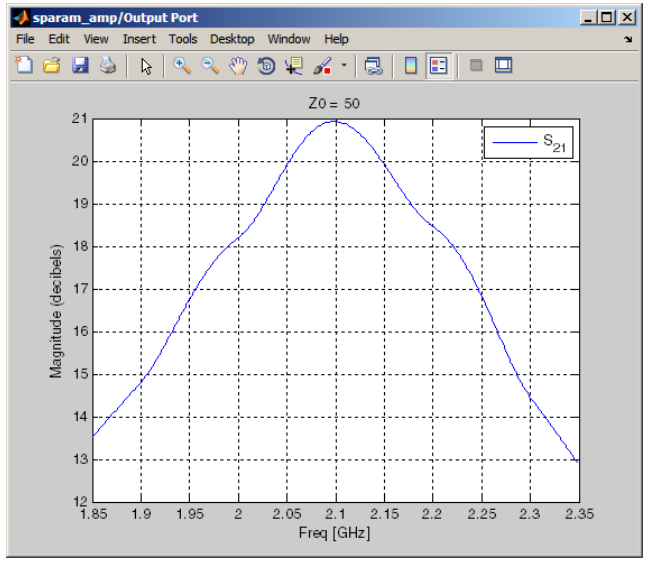
In this part of the example, you open and run a RF Blockset Equivalent Baseband example that uses file data to specify an amplifier in a physical subsystem. Then, you plot the network parameters of the physical subsystem, which consists of the General Amplifier, the Input Port, and the Output Port blocks.

1. Type `sparam_amp` at the MATLAB prompt to open the RF Blockset Equivalent Baseband example called “AMP Data File for Amplifier”.
2. In the model window, select **Simulation** > **Start** to run the simulation. Once the simulation has reached steady state, select **Simulation** > **Stop** to stop the simulation.
3. Double-click the Output Port block to open the block dialog box.
4. Select the **Visualization** tab and set the Output Port block parameters as follows:
   - In the **Source of frequency data** list, select `Derived from Input Port parameters`.
   - In the **Plot type** list, select `X-Y plane`.
   - In the **Y Parameter1** list, select `S21`.



5 Click **Plot**.

This action plots the magnitude of S_{21} (in decibels) as a function of frequency on an X-Y plot.

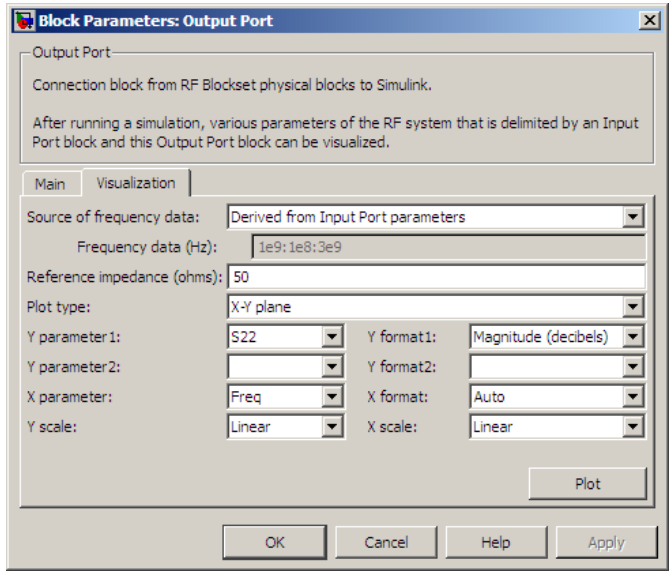


S_{21} versus Frequency for a Physical Subsystem

Add Data to an Existing Plot

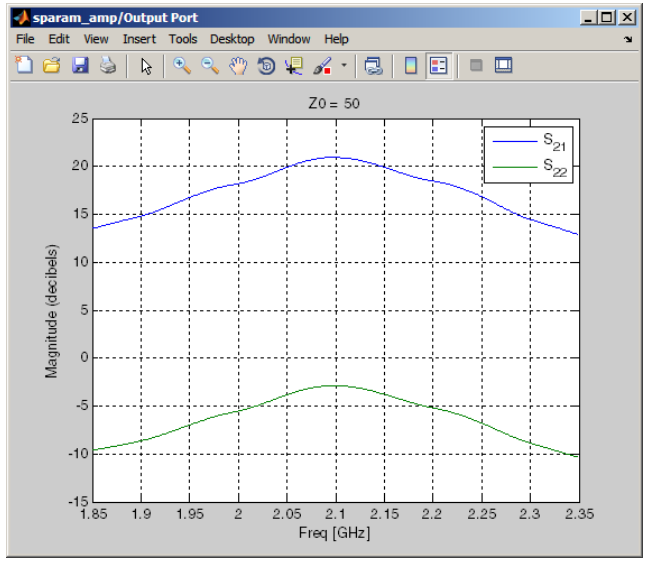
In this part of the example, you add data to the plot you created in “Plot the Network Parameters of a Subsystem” on page 7-28.

1. Double-click the Output Port block to open the block dialog box.
2. Select the **Visualization** tab and change the value of **Y Parameter1** to S22.



3 Click **Plot**.

This action adds S_{22} to the plot.



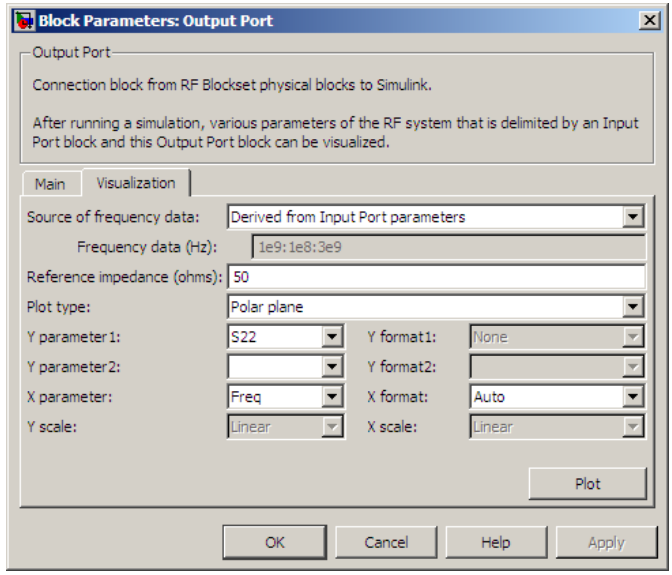
S_{21} and S_{22} versus Frequency for a Physical Subsystem

Change Data on an Existing Plot

In this part of the example, you change the data on the plot you created in the previous steps of the example by modifying the **Plot type**.

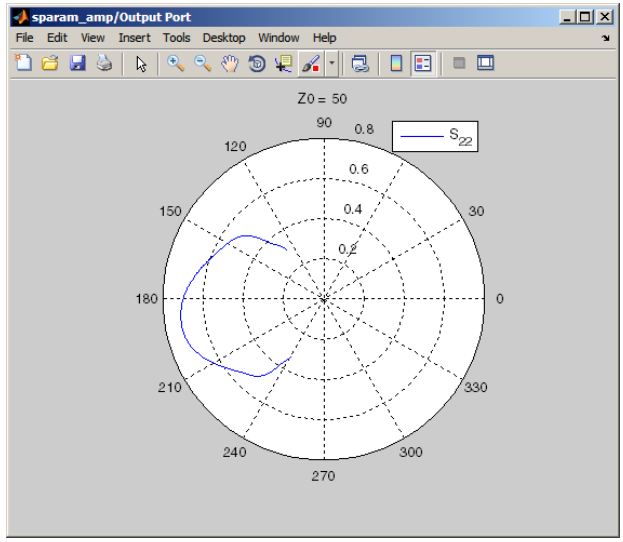
1. Double-click the Output Port block to open the block dialog box.
2. Select the **Visualization** tab and change the value of **Plot type** to `Polar plane`, as shown in the following figure.

   As the figure shows, the value of **Y Parameter1** remains as S_{22}, the last parameter selected for the previous plot.



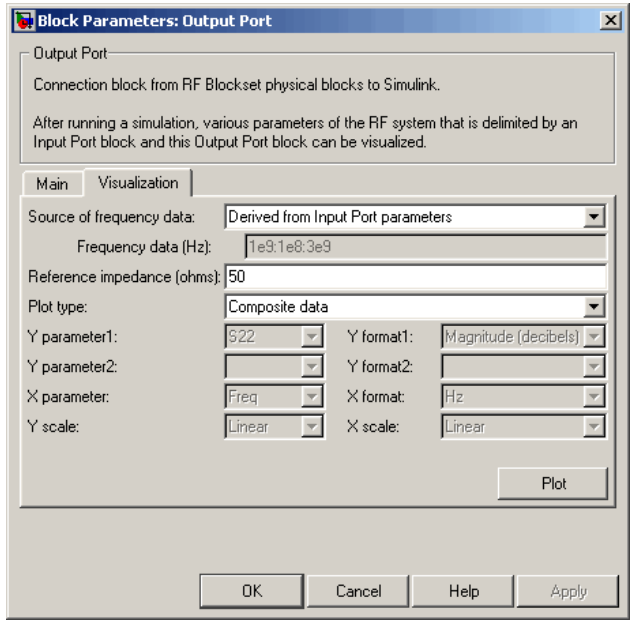
3 Click **Plot**.

This action creates a Polar plane plot of S_{22} as a function of frequency.



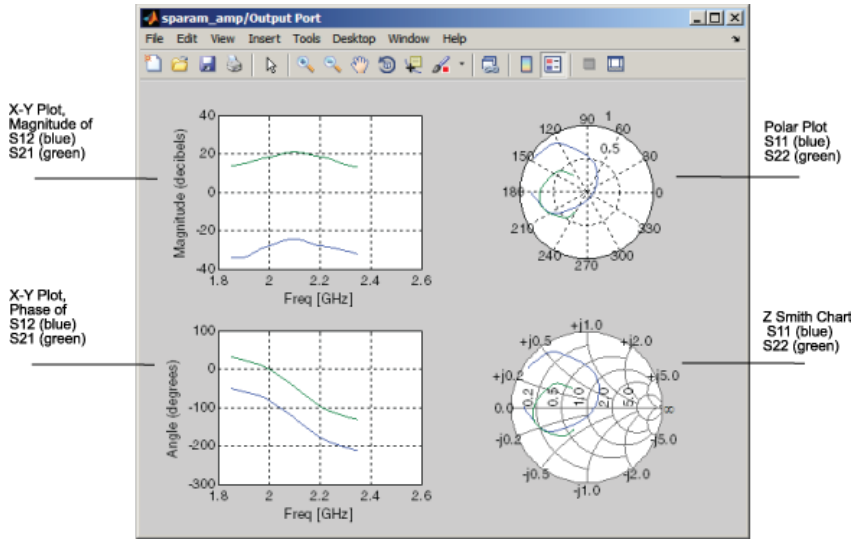
S_{22} versus Frequency for a Physical Subsystem

4 In the Output Port block dialog box, change the **Plot type** to `Composite data` to generate four plots in one figure. The parameters for the plots are defined by the block, so the **Y Parameter1**, **Y Parameter2**, and **X Parameter** fields becomes invisible.



5 Click **Plot**.

This action creates a composite plot.



Composite Plot for a Physical Subsystem

A

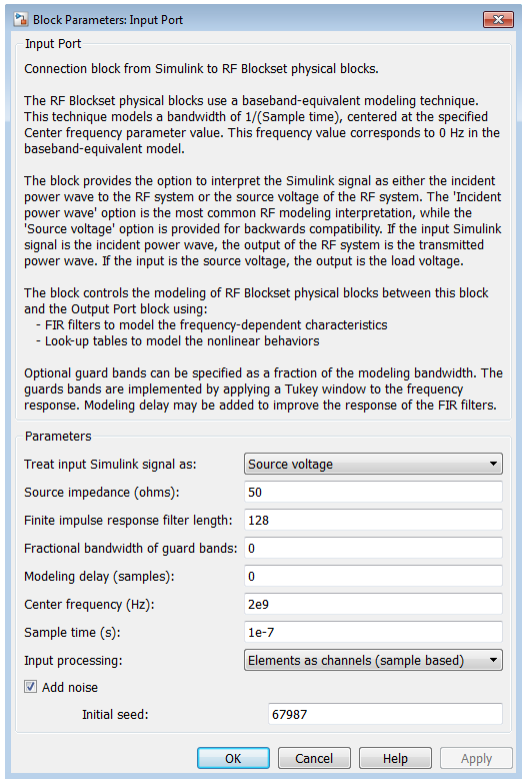
RF Blockset Equivalent Baseband Algorithms

Simulate an RF Model

When you simulate a model that contains physical blocks, RF Blockset Equivalent Baseband software determines the modeling frequencies of the physical system using the Input Port block parameters. The *modeling frequencies* are the frequencies at which the blockset takes information from the blocks to construct the baseband-equivalent model. Then, the software determines the block parameter values at those frequencies and uses the information to create a baseband-equivalent model for Simulink time-domain simulation.

Determine Modeling Frequencies

When you simulate an RF model, the Output Port block uses Input Port block parameters to determine the modeling frequencies f for the physical system that is bracketed between the Input Port block and the Output Port block. f is an N-element vector, where N is the finite impulse response filter length. The modeling frequencies are a function of the center frequency f_c and the sample time t_s. The following figure shows the Input Port block parameters that determine the modeling frequencies.



f_n is the nth element of the vector of modeling frequencies, f, and is given by

$$f_n = f_{\min} + \frac{n-1}{t_s N} \quad n = 1,\ldots,N$$

where

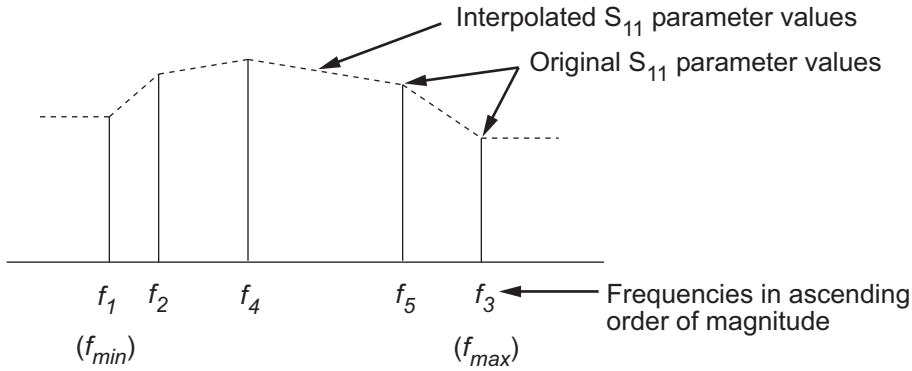
$$f_{\min} = f_c - \frac{1}{2t_s}$$

Map Network Parameters to Modeling Frequencies

In a physical system, each block provides network parameters at different frequencies. These frequencies are not necessarily the modeling frequencies for the physical system in which the block resides. To create a baseband-equivalent model, RF Blockset Equivalent Baseband software must calculate the values of the S-parameters at the modeling frequencies.

Individual physical blocks calculate the S-parameters at the modeling frequencies determined by the Input Port block parameters. Each block interpolates its S-parameters to determine the S-parameters at the modeling frequencies. If the block contains network Y- or Z-parameters, it first converts them to S-parameters.

Specifically, the block orders the S-parameters in the ascending order of their frequencies, f_n. Then, it interpolates the S-parameters using the MATLAB `interp1` function. For example, the curve in the following diagram illustrates the result of interpolating the S_{11} parameters at original frequencies f_1 through f_5.



The **Interpolation method** field in the individual block dialog boxes enables you to specify the interpolation method as `Cubic`, `Linear` (default), or `Spline`. For more information about these methods, see the `interp1` reference page in the MATLAB documentation.

As shown in the previous diagram, each block uses the parameter values at f_{min} for all modeling frequencies smaller than f_{min}. The block uses the parameter values at f_{max} for all modeling frequencies greater than f_{max}. In both cases, the results may not be accurate, so

you need to specify network parameter values over a range of frequencies that is wide enough to account for the block behavior.

Model Noise in an RF System

In this section...

The RF Blockset Equivalent Baseband Physical library blocks can model noise. The Input Port block parameters specify whether to include noise in a simulation. When you include noise information in your model, the blockset simulates the noise of the physical system by combining the noise contributions from each individual block. This section explains how the blockset simulates noise from user-specified information. For information on how to add noise to an RF model, see “Model Noise” on page 6-26.

Output-Referred Noise in RF Models

In general, you can specify output-referred noise in one of three ways:

- `Noise temperature` — Specifies the noise in kelvin.
- `Noise factor` — Specifies the noise by the following equation:

$$\text{Noise factor} = 1 + \frac{\text{Noise temperature}}{290}$$

- `Noise figure` — Specifies the noise in decibels relative to the standard reference noise temperature of 290 K. In terms of noise factor

  Noise figure = 10log(Noise factor)

These three specifications are equivalent, because you can compute each one from any of the others.

The blockset lets you simulate the noise associated with any physical block in your RF model.

The blockset automatically determines the noise properties of passive blocks from their network parameters. The blockset gets these network parameters either explicitly from

the block dialog box or specified data files, or implicitly by calculating them from the specified block parameters.

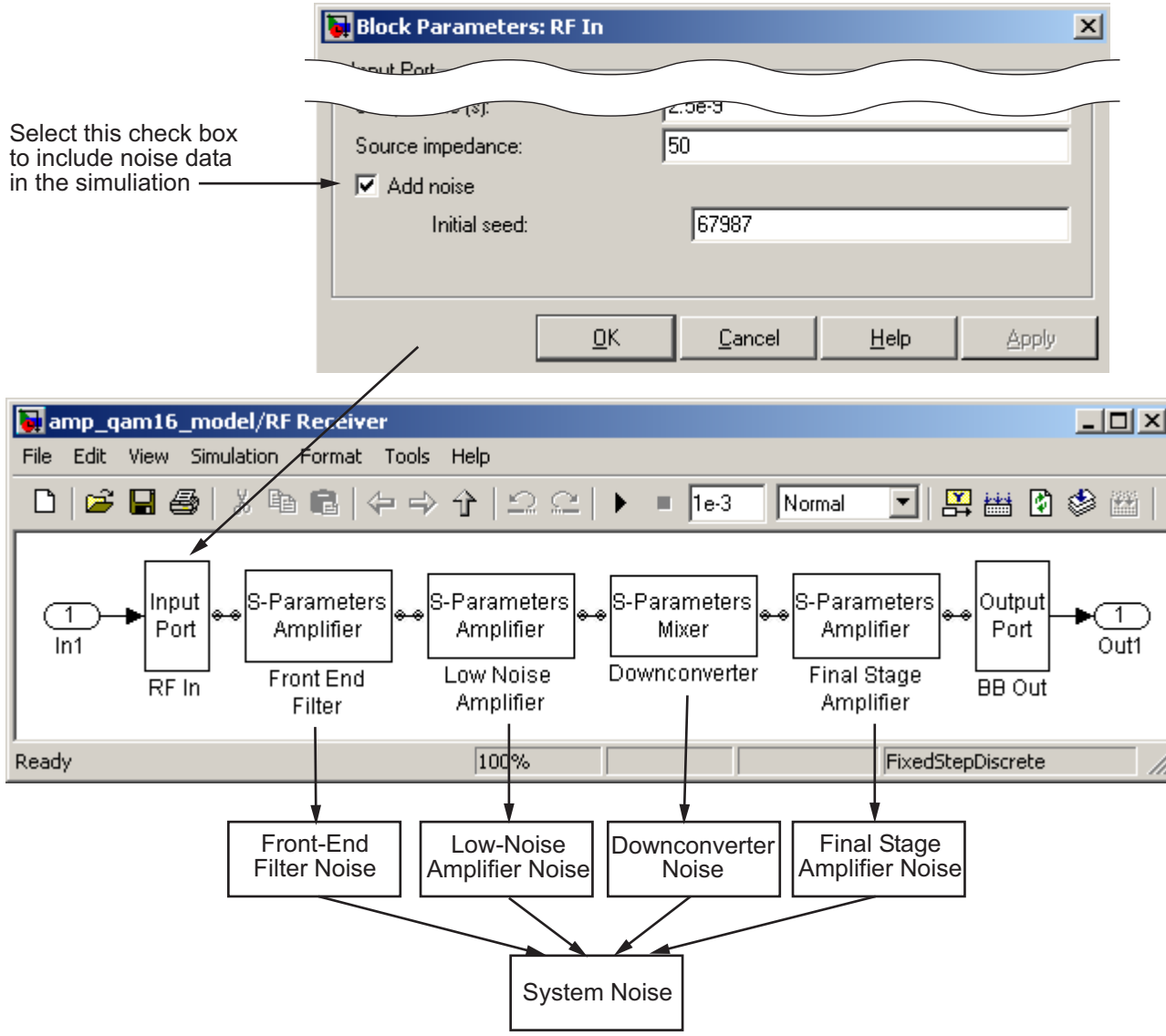
For active devices such as amplifiers and mixers, the noise properties cannot be inferred from network parameters. Therefore, for the amplifier and mixer blocks, you must specify the noise information explicitly, either in the dialog block or the associated data file.

For physical amplifier and mixer blocks, you can specify active block noise in one of the following ways:

- Spot noise data
- Frequency-independent noise figure, noise factor, or noise temperature values
- Frequency-dependent noise figure data (`rfdata.nf`) or spot noise data (`rfdata.noise`) object

These noise specification options are described in “Amplifier and Mixer Noise Specifications” on page 6-26.

When you run the simulation, the blockset first computes the noise figure values for each individual block at the modeling frequencies. Then, it computes the noise figure of the physical system from the individual noise figure values and uses the system noise figure information to calculate the output noise power. This process is shown in the following figure.


Block Parameters: RF In
Source impedance:
50
Add noise
Initial seed:
67987
OK
Cancel
Help
Apply
Select this check box to include noise data in the simuliation
amp_qam16_model/RF Receiver
File Edit View Simulation Format Tools Help
1e-3
Normal
1
In1
Input Port
RF In
S-Parameters Amplifier
Front End Filter
S-Parameters Amplifier
Low Noise Amplifier
S-Parameters Mixer
Downconverter
S-Parameters Amplifier
Final Stage Amplifier
Output Port
BB Out
1
Out1
Ready
100%
FixedStepDiscrete
Front-End Filter Noise
Low-Noise Amplifier Noise
Downconverter Noise
Final Stage Amplifier Noise
System Noise

Calculate Noise Figure at Modeling Frequencies

To include noise information in a simulation, the blockset must compute the noise figure values of each individual block at the modeling frequencies.

If you specify the frequency-independent noise figure value directly, or if the blockset computes the noise figure value from the block resistance, the blockset uses this value for the noise figure value at each of the modeling frequencies.

If you specify the noise factor or noise temperature value, the blockset computes the noise figure value from the specified value using the equations in the preceding section and uses the computed value for the noise figure value at each of the modeling frequencies.

If you specify frequency-dependent noise figure values using an `rfdata.nf` object, the blockset interpolates the values using the **Interpolation method** specified in the block dialog box to get the noise figure value at each of the modeling frequencies.

If you specify spot noise data, the blockset computes frequency-dependent noise figure information from this data. It takes the minimum noise figure, NF_{min}, equivalent noise resistance, R_n, and optimal source admittance, Y_{opt}, values in the file and interpolates to find the values at the modeling frequencies. Then, the blockset uses the following equation to calculate the noise correlation matrix, C_A:

$$C_A = 2kT\begin{bmatrix} R_n & \frac{NF_{\min}-1}{2} - R_n Y_{opt}^{*} \\ \frac{NF_{\min}-1}{2} - R_n Y_{opt} & R_n \left|Y_{opt}\right|^2 \end{bmatrix}$$

where k is Boltzmann's constant, and T is the noise temperature in Kelvin.

The blockset then calculates the noise factor, F, from the noise correlation matrix as follows:

$$F = 1 + \frac{z^{+} C_A z}{2kT\,\mathrm{Re}\{Z_S\}}$$

$$z = \begin{bmatrix} 1 \\ Z_S^{*} \end{bmatrix}$$

In the two preceding equations, Z_S is the nominal impedance, which is 50 ohms, and z^+ is the Hermitian conjugation of z.

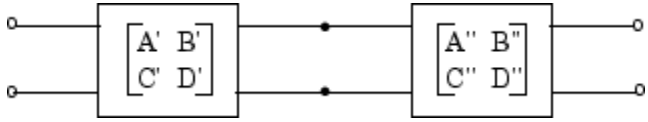
The blockset obtains the noise figure, *NF*, from the noise factor:

$$NF = 10\log(F)$$

Calculate System Noise Figure

The blockset uses a recursive process to calculate system noise figure. The noise correlation matrices for the first two elements of the cascade are combined into a single matrix, and the process is repeated.

The following figure shows a cascaded network consisting of two 2-port networks, each represented by its ABCD-parameters.



First, the blockset calculates noise correlation matrices C_A' and C_A'' for the two networks. Then, the blockset combines C_A' and C_A'' into a single correlation matrix C_A using the equation

$$C_A = C_A' + \begin{bmatrix} A' & B' \\ C' & D' \end{bmatrix} C_A'' \begin{bmatrix} A' & B' \\ C' & D' \end{bmatrix}$$

The ABCD-parameter matrices in the cascade combine according to matrix multiplication:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} A' & B' \\ C' & D' \end{bmatrix} \begin{bmatrix} A'' & B'' \\ C'' & D'' \end{bmatrix}$$

If there is another element in the cascade, the same calculations will be performed using these ABCD-parameters as well as the ABCD-parameters corresponding to the following element. The recursion will terminate with a noise correlation matrix pertaining to the entire system. The blockset then calculates the system noise figure from this matrix.

For more information about these calculation techniques, see the following article:

Hillbrand, H. and P.H. Russer, “An Efficient Method for Computer Aided Noise Analysis of Linear Amplifier Networks,” *IEEE Transactions on Circuits and Systems*, Vol. CAS-23, Number 4, pp. 235–238, 1976.

Calculate Output Noise Power

The blockset uses noise power to determine the amplitude of the noise that it adds to the physical system using a Gaussian distributed pseudorandom number generator. It uses both the noise temperature and the modeling bandwidth to calculate the noise power:

Noise power = kTB

where k is Boltzmann's constant, T is the noise temperature in Kelvin, and B is the bandwidth in hertz.

The blockset computes noise temperature from the specified or calculated noise figure values for the system, and it computes the modeling bandwidth from the model's sample time and center frequency.

Create Complex Baseband-Equivalent Model

In this section...

Baseband-Equivalent Modeling

RF Blockset Equivalent Baseband software simulates the physical system in the time domain using a complex baseband-equivalent model that it creates from the passband frequency-domain parameters of the physical blocks. This type of modeling is also known as lowpass equivalent (LPE), complex envelope, or envelope modeling.

To create a complex baseband-equivalent model in the time domain based on the network parameters of the physical system, the blockset performs a mathematical transformation that consists of the following three steps:

1. “Calculate the Passband Transfer Function” on page A-13
2. “Calculate the Baseband-Equivalent Transfer Function” on page A-15
3. “Calculate the Baseband-Equivalent Impulse Response” on page A-16

Calculate the Passband Transfer Function

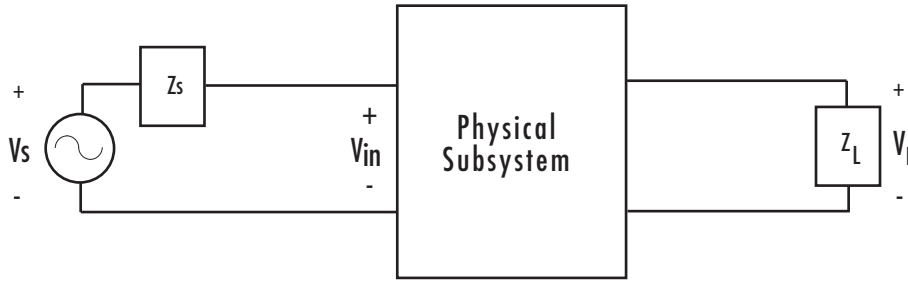
The blockset calculates the passband transfer function from the physical block parameters at the modeling frequencies by calculating the transfer function of the physical subsystem and then applying the Tukey window to obtain the passband transfer function.

Note To learn how the blockset uses the specified network parameters to compute the network parameters at the modeling frequencies, see “Map Network Parameters to Modeling Frequencies” on page A-5.

The transfer function of the physical subsystem is defined as:

$$H(f) = \frac{V_L(f)}{V_S(f)}$$

where V_S and V_L are the source and load voltages shown in the following figure, and f represents the modeling frequencies.



More specifically,

$$H(f) = \frac{S_{21}(1+\Gamma_l)(1-\Gamma_s)}{2(1-S_{22}\Gamma_l)(1-\Gamma_{in}\Gamma_s)}$$

where

$$\Gamma_l = \frac{Z_l - Z_o}{Z_l + Z_o}$$

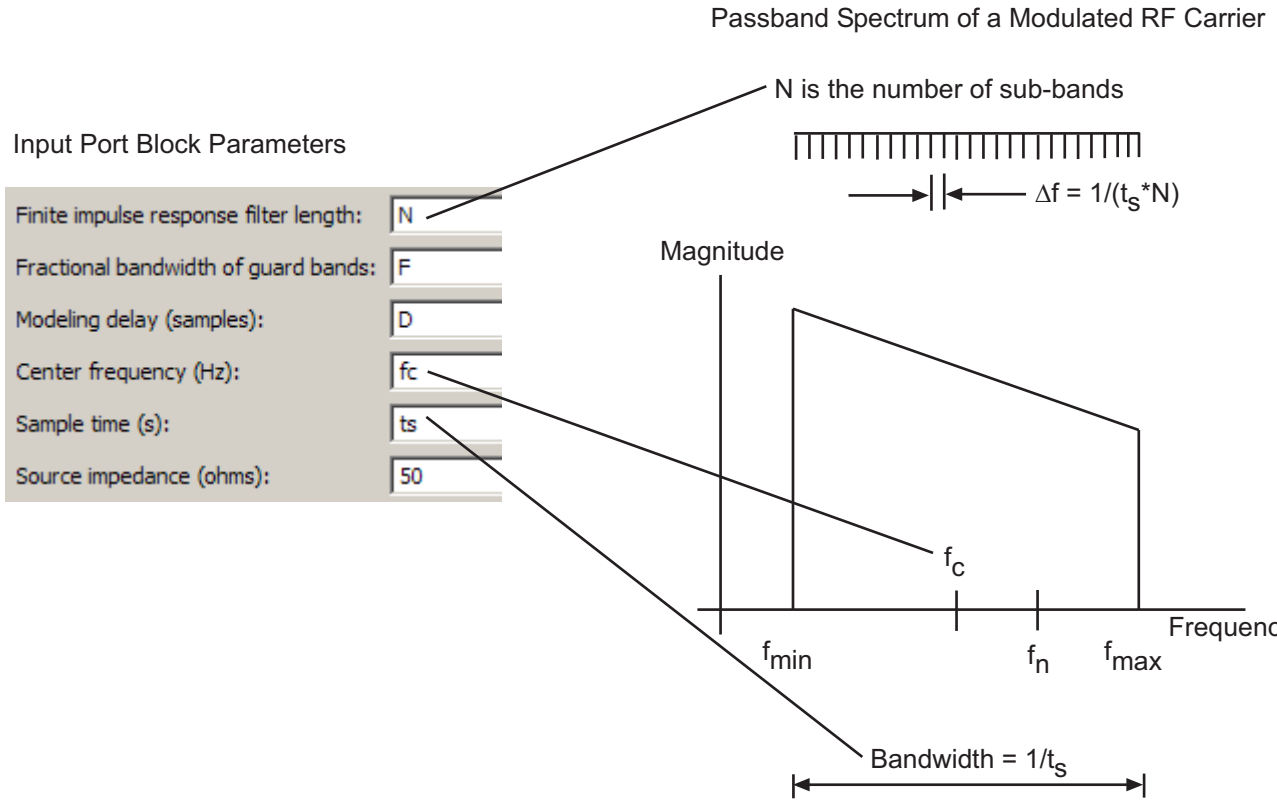
$$\Gamma_s = \frac{Z_s - Z_o}{Z_s + Z_o}$$

$$\Gamma_{in} = S_{11} + \left(S_{12}S_{21} \frac{\Gamma_l}{(1 - S_{22}\Gamma_l)} \right)$$

and

- Z_S is the source impedance.
- Z_L is the load impedance.
- S_{ij} are the S-parameters of a two-port network.

The blockset derives the transfer function of the physical subsystem from the Input Port block parameters as shown in the following figure.



The blockset then applies the Tukey window to obtain the passband transfer function:

$$H_{passband}(f) = H(f) \cdot \texttt{tukeywin}(N, F)$$

where `tukeywin` is the Signal Processing Toolbox™ `tukeywin` function.

Calculate the Baseband-Equivalent Transfer Function

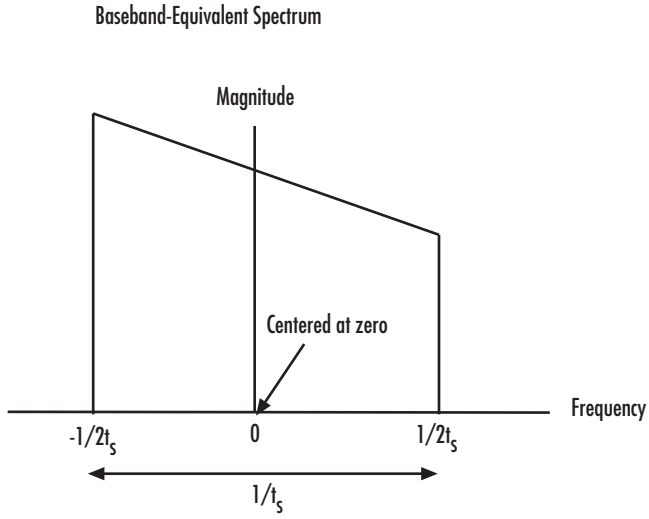
The blockset calculates the baseband transfer function, $H_{baseband}(f)$, by translating the passband transfer function to its equivalent baseband transfer function:

$$H_{baseband}(f) = H_{passband}(f + f_c)$$

where f_c is the specified center frequency.

The resulting baseband-equivalent spectrum is centered at zero, so the blockset can simulate the system using a much larger time step than Simulink can use for the same system. For information on why this translation allows for a larger time step, see “Simulation Efficiency of a Baseband-Equivalent Model” on page A-17.

The baseband transfer function is shown in the following figure.

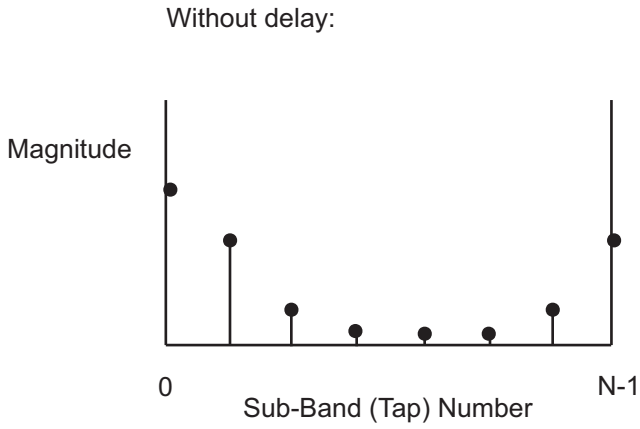


Calculate the Baseband-Equivalent Impulse Response

The blockset calculates the baseband-equivalent impulse response by performing the following steps:

1. Calculate the inverse FFT of the baseband transfer function. For faster simulation, the block calculates the IFFT using the next power of 2 greater than the specified finite impulse response filter length. Then, it truncates the impulse response to a length equal to the filter length specified. When the finite impulse response is truncated to the length specified by the user, the effect of the truncation is similar to windowing with a rectangular window.
2. Apply the delay specified by the **Modeling delay (samples)** parameter in the Input Port block dialog box. Selecting an appropriate value for this delay ensures that the baseband-equivalent model has a causal response by moving the time window such

that the model energy is concentrated at the center of the window, as shown in the following figure:



Simulation Efficiency of a Baseband-Equivalent Model

The baseband-equivalent modeling technique improves simulation speed by allowing the simulator to take larger time steps. To simulate a system in the time domain, Simulink would require a step size of:

$$t_{step} = \frac{1}{2f_{\max}}$$

Using the baseband-equivalent model of the same system, whose spectrum has been shifted down by f_c, allows for a much larger time step of:

$$t_{step} = \frac{1}{2(f_{\max} - f_c)} = \frac{1}{f_{\max} - f_{\min}}$$

Example — Select Parameter Values for a Baseband-Equivalent Model

Baseband-Equivalent Modeling Example Overview

In this example, you model an RF transmission line stimulated by a pulse and plot the baseband-equivalent model that the blockset uses to simulate the transmission line in the time domain. You compare the effects of using different parameter values for the baseband-equivalent model. This example helps you understand how to use these parameters to best apply the baseband-equivalent modeling paradigm of performing time-domain simulation using a limited band of frequency data.

Create the Model

In this part of the example, you perform the following tasks:

Select Blocks to Represent System Components

In this part of the example, you select the blocks to represent:

- The input signal
- The RF transmission line
- The baseband-equivalent model display

The following table lists the blocks that represent the system components and a description of the role of each block.

Block	Description
Discrete Impulse	Generates a frame-based pulse input signal.
Real-Imag to Complex	Converts the real pulse signal into a complex pulse signal.
Input Port	Establishes parameters that are common to all blocks in the RF transmission line subsystem, including the source impedance of the subsystem that is used to convert Simulink signals to the physical modeling environment.
RLCG Transmission Line	Models the signal attenuation caused by an RF transmission line.
Output Port	Establishes parameters that are common to all blocks in the RF transmission line subsystem. These parameters include the load impedance of the subsystem, which is used to convert RF signals to Simulink signals.
Complex to Magnitude-Angle	Converts the complex signal from the Output Port block into magnitude-angle format.
Vector Scope	Displays the time-domain baseband-equivalent model.

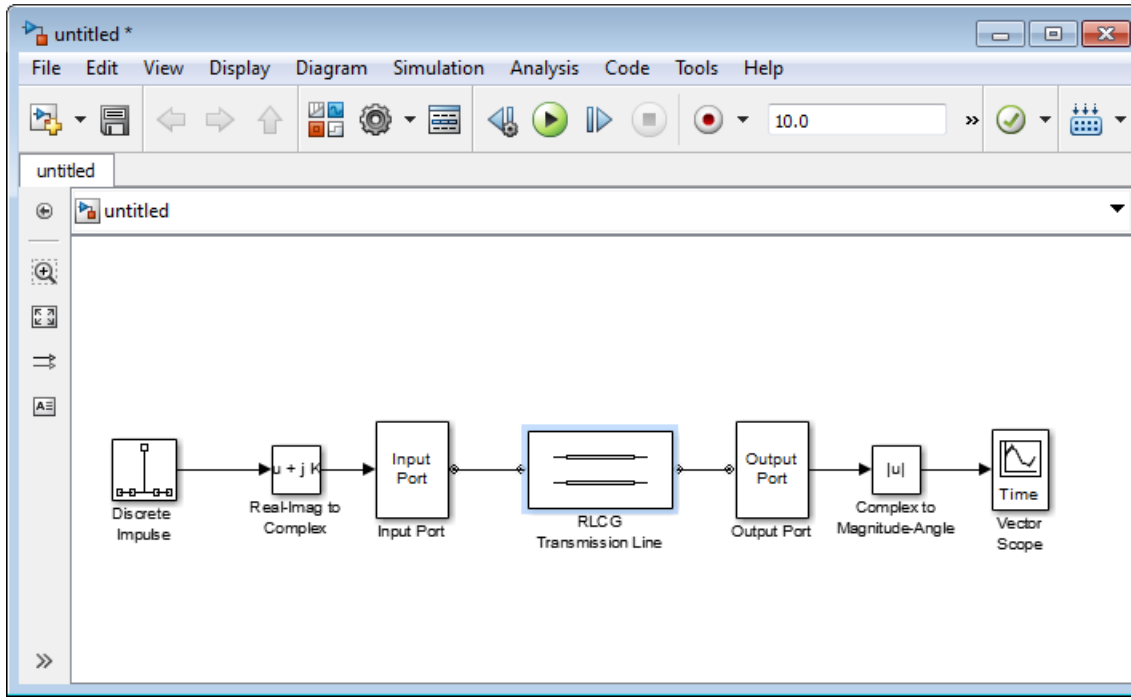
Build the Model

In this part of the example, you create a Simulink model, add blocks to the model, and connect the blocks.

1. Create a model.
2. Add to the model the blocks shown in the following table. The Library Path column of the table specifies the hierarchical path to each block.

Block	Library Path	Quantity
Discrete Impulse	**DSP System Toolbox > Sources**	1
Real-Imag to Complex	**Simulink > Math Operations**	1
Input Port	**RF Blockset > Equivalent Baseband > Input/ Output Ports**	1
RLCG Transmission Line	**RF Blockset > Equivalent Baseband > Transmission Lines**	1
Output Port	**RF Blockset > Equivalent Baseband > Input/ Output Ports**	1
Complex to Magnitude-Angle	**Simulink > Math Operations**	1
Vector Scope	**DSP System Toolbox > Sinks**	1

3 Connect the blocks as shown in the following figure.



Now you are ready to specify model variables.

Specify Model Variables

Type the following at the MATLAB prompt to set up workspace variables for the model:

```
t_s = 5e-10;   % Sample time
f_c = 3e9;         % Center frequency
taps = 64;     % Filter length
```

Now you are ready to specify the block parameters.

Specify Model Parameters

In this part of the example, you specify the following parameters to represent the behavior of the system components:

- “Input Signal Parameters” on page A-21
- “Transmission Line Subsystem Parameters” on page A-22
- “Signal Display Parameters” on page A-26

Input Signal Parameters

You generate the frame-based complex pulse source signal using two blocks:

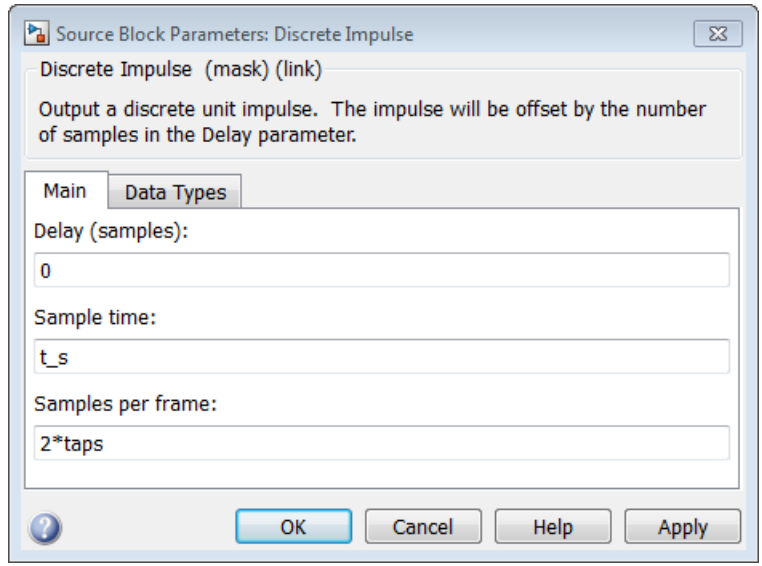
- The Discrete Impulse block generates a real pulse signal.
- The Real-Imag to Complex block converts the real signal to a complex signal.

Note All signals in the RF model must be complex to match the signals in the physical subsystem, so you create a complex input signal.

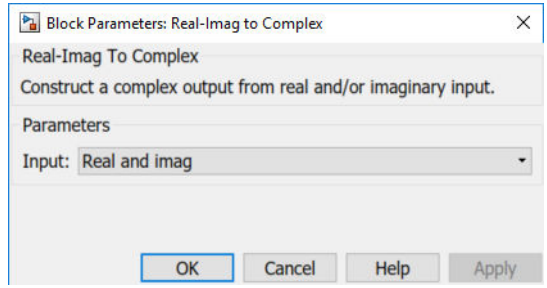
---

1 In the Discrete Impulse block parameters dialog box:

- Set **Sample time** to `t_s`.
- Set **Samples per frame** to `2*taps`.



2 Set the Real-Imag to Complex block **Input** parameter to `Real`. Changing this parameter changes the number of block inputs from two to one, making the block fully connected.



Transmission Line Subsystem Parameters

In this part of the example, you configure the blocks that model the RF filter subsystem—the Input Port, Transmission Line, and Output Port blocks.

1 In the Input Port block parameters dialog box:

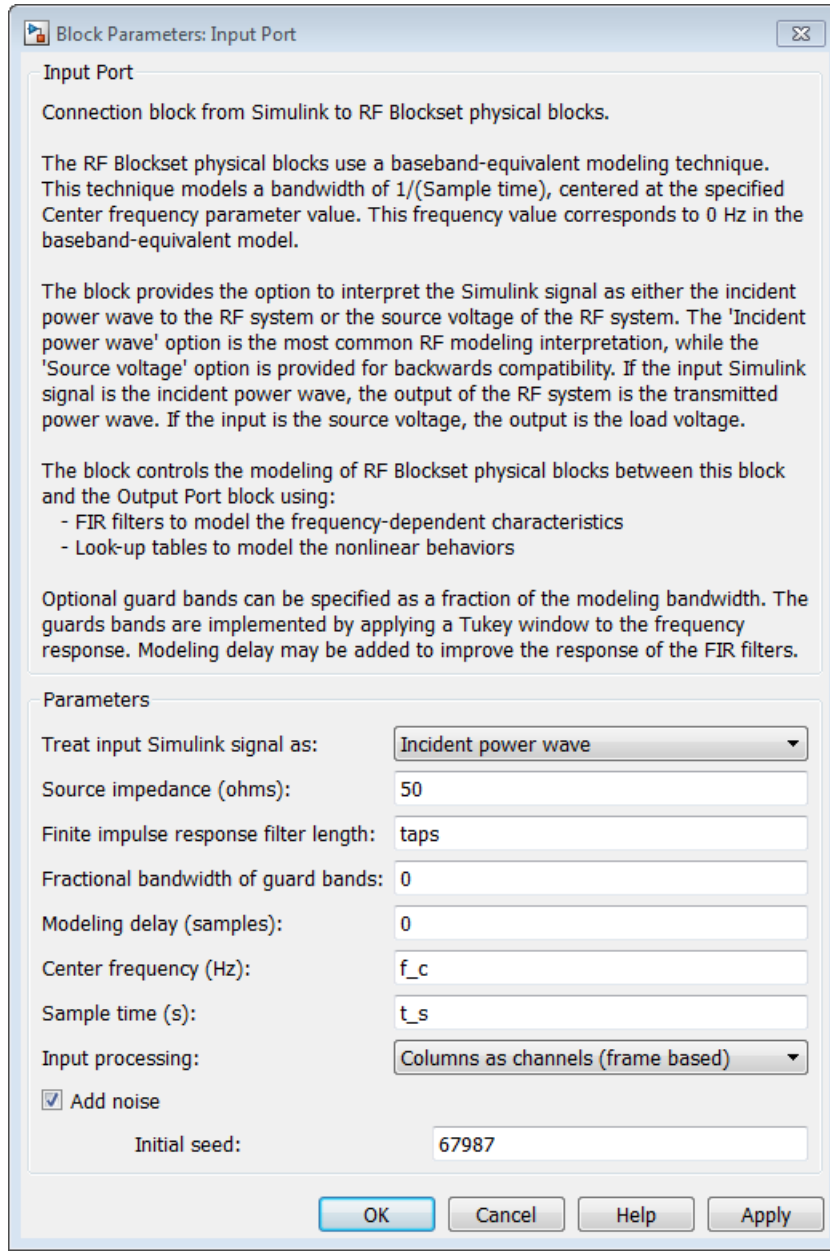
- Set **Treat input Simulink signal as** to `Incident power wave`.

This option tells the blockset to interpret the input signal as the incident power wave to the RF subsystem, and not the source voltage of the RF subsystem.

Note If you use the default value for this parameter, the software interprets the input Simulink signal as the source voltage. As a result, the source and the load that model the Input Port and Output Port blocks, respectively, introduce 6 dB of loss into the physical system at all frequencies. For more information on why this loss occurs, see the note in “Convert to and from Simulink Signals” on page A-31.

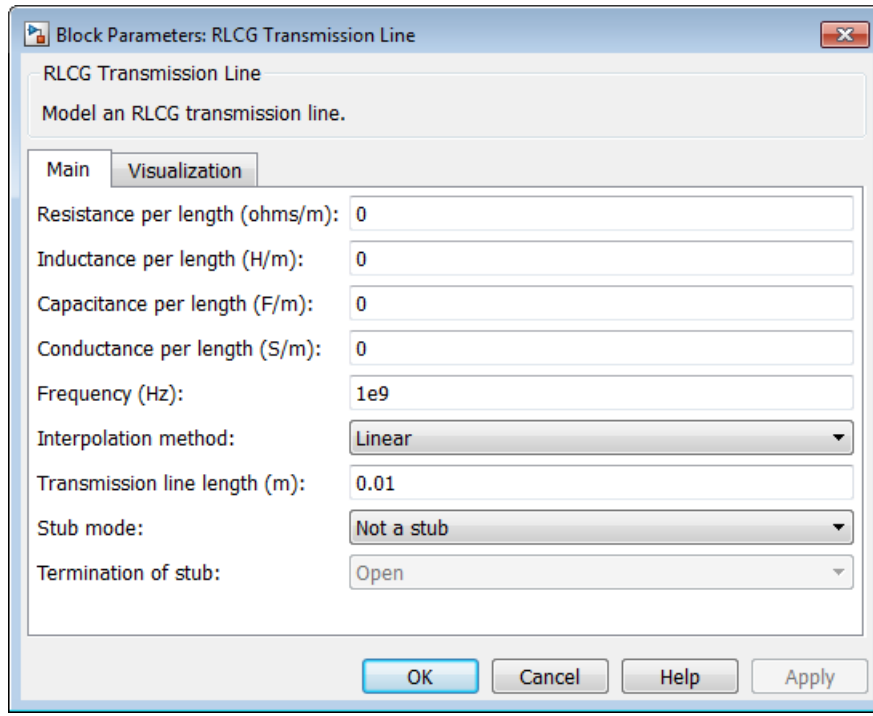
- Set **Finite impulse response filter length** to `taps`.
- Set **Center frequency** to `f_c`.
- Set **Sample time (s)** to `t_s`.

  This sample time is equivalent to a modeling bandwidth of 1/`t_s` seconds.
- Set **Input Processing** to `Columns as channels (frame based)`.

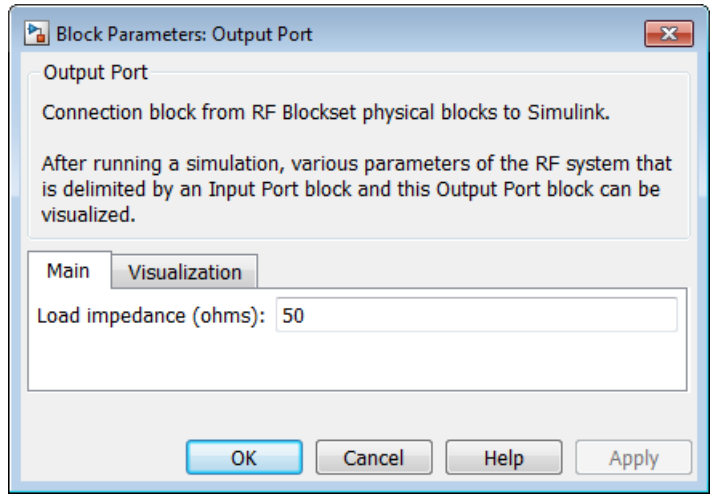

Block Parameters: Input Port
Input Port
Connection block from Simulink to RF Blockset physical blocks.
The RF Blockset physical blocks use a baseband-equivalent modeling technique. This technique models a bandwidth of 1/(Sample time), centered at the specified Center frequency parameter value. This frequency value corresponds to 0 Hz in the baseband-equivalent model.
The block provides the option to interpret the Simulink signal as either the incident power wave to the RF system or the source voltage of the RF system. The 'Incident power wave' option is the most common RF modeling interpretation, while the 'Source voltage' option is provided for backwards compatibility. If the input Simulink signal is the incident power wave, the output of the RF system is the transmitted power wave. If the input is the source voltage, the output is the load voltage.
The block controls the modeling of RF Blockset physical blocks between this block and the Output Port block using:
- FIR filters to model the frequency-dependent characteristics
- Look-up tables to model the nonlinear behaviors
Optional guard bands can be specified as a fraction of the modeling bandwidth. The guards bands are implemented by applying a Tukey window to the frequency response. Modeling delay may be added to improve the response of the FIR filters.
Parameters
Treat input Simulink signal as:
Incident power wave
Source impedance (ohms):
50
Finite impulse response filter length:
taps
Fractional bandwidth of guard bands:
0
Modeling delay (samples):
0
Center frequency (Hz):
f_c
Sample time (s):
t_s
Input processing:
Columns as channels (frame based)
Add noise
Initial seed:
67987
OK
Cancel
Help
Apply

2 In the RLCG Transmission Line block parameters dialog box:

- Set **Inductance per length (H/m)** to `50`.
- Set **Capacitance per length (F/m)** to `.02`.
- Set **Frequency (Hz)** to `f_c`.
- Set **Transmission line length (m)** to `0.5*t_s`.



3 Accept the default parameters for the Output Port block to use a load impedance of 50 ohms.

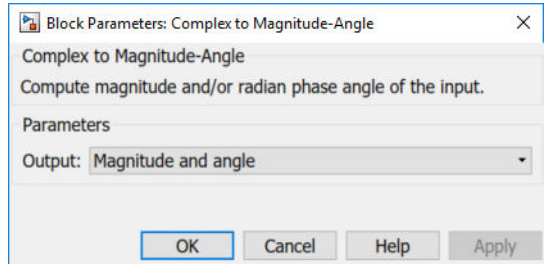


Signal Display Parameters

In this part of the example, you specify the parameters that set up the baseband-equivalent model display. You use the Complex to Magnitude-Angle block to convert the RF subsystem output to magnitude format, and you use the Vector Scope block to display the output signal.

For the Vector Scope block, you specify that the signal is an amplitude as a function of time, and you set the range of the *x*- and *y*-axes to make sure that the entire signal is visible.

1 Set the Complex to Magnitude-Angle block **Output** parameter to `Magnitude`. Changing this parameter changes the number of block outputs from two to one, making the block fully connected.



2 In the Vector Scope block parameters dialog:

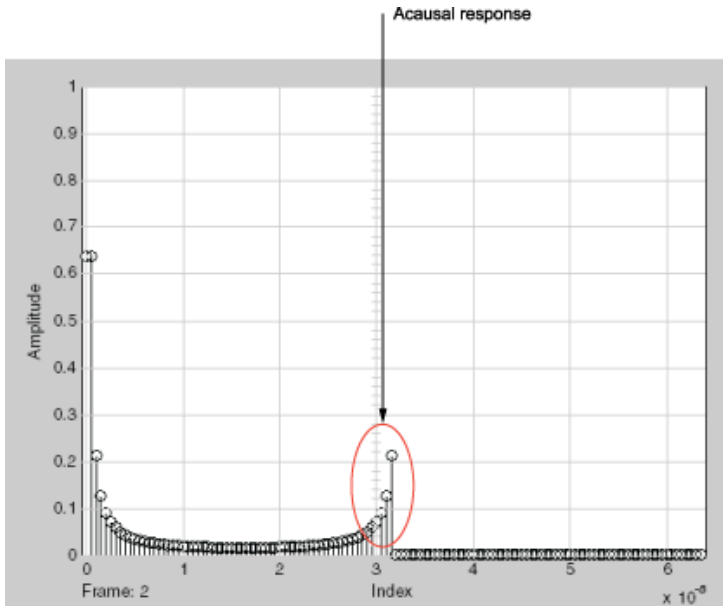
- In the **Scope Properties** tab, set **Input domain** to `User-defined`.
- In the **Axis Properties** tab:
  - Set **X-axis title** to `Index`.
  - Set **Minimum Y-limit** to `0`.
  - Set **Maximum Y-limit** to `1`.
  - Set **Y-axis title** to `Amplitude`.
- In the **Line Properties** tab, set **Line markers** to `stem`.

Run the Simulation and Analyze the Results

Before you run the simulation, set the stop time by selecting **Simulation** > **Configuration Parameters** in the model window and entering `2*t_s*(taps-1)` for the **Stop time** parameter.

To run the simulation, select **Simulation** > **Start** in the model window.

You can view the baseband-equivalent model in the Vector Scope window while the model is running. This window appears automatically when you start the simulation. The following plot shows the baseband-equivalent model, which contains a significant amount of acausal energy because of the limited bandwidth of the model.



Baseband-Equivalent Model

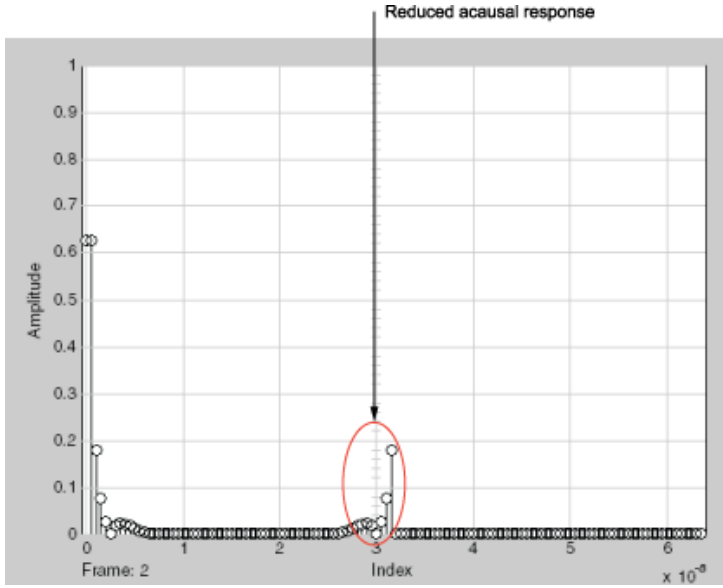
The next part of the example shows you how to reduce this acausal response.

Reducing Acausal Response in the Baseband-Equivalent Model

In this part of the example, you adjust the **Fractional bandwidth of guard bands** parameter. This parameter controls the shaping of the filter that the blockset applies to create the baseband-equivalent model.

1. Set the Input Port **Fractional bandwidth of guard bands** parameter to `0.2`.
2. Rerun the simulation.

The following figure shows the Vector Scope plot. You can see that the acausal response is lower than it was for the previous simulation, but there is still some energy wrapping around the end of the model because it is periodic.



Baseband-Equivalent Model with Filter Shaping

Note You can further reduce the acausal response in the baseband-equivalent model by increasing the value of the **Fractional bandwidth of guard bands** parameter above 0.2, but if you use a high value, you compromise the fidelity of the gain of the transmission line.

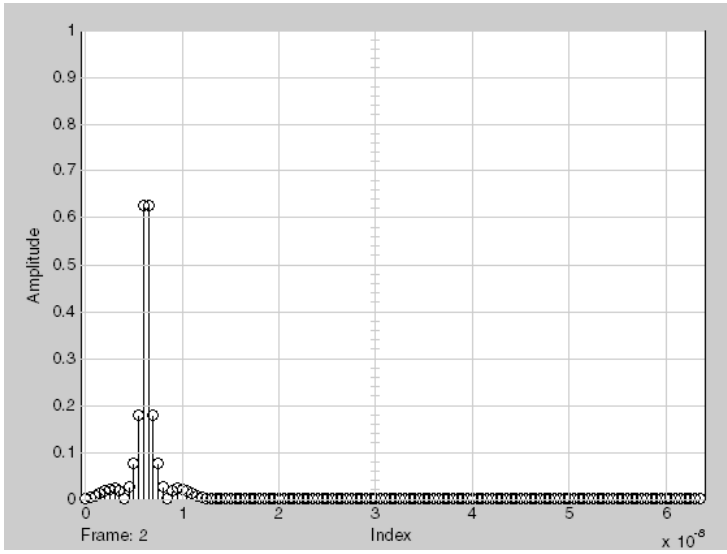
The next section shows you how to shift the response to avoid this wrapping.

Introduce Delay into the Baseband-Equivalent Model

In this part of the example, you adjust the **Modeling delay (samples)** parameter. This parameter controls the delay the blockset applies to create the baseband-equivalent model.

1. Set the Input Port **Modeling delay (samples)** parameter to 12.
2. Rerun the simulation.

The following figure shows the Vector Scope plot. The response of the baseband-equivalent model is concentrated in a small time window. This model provides the most accurate time-domain simulation of the specified band of frequency data.

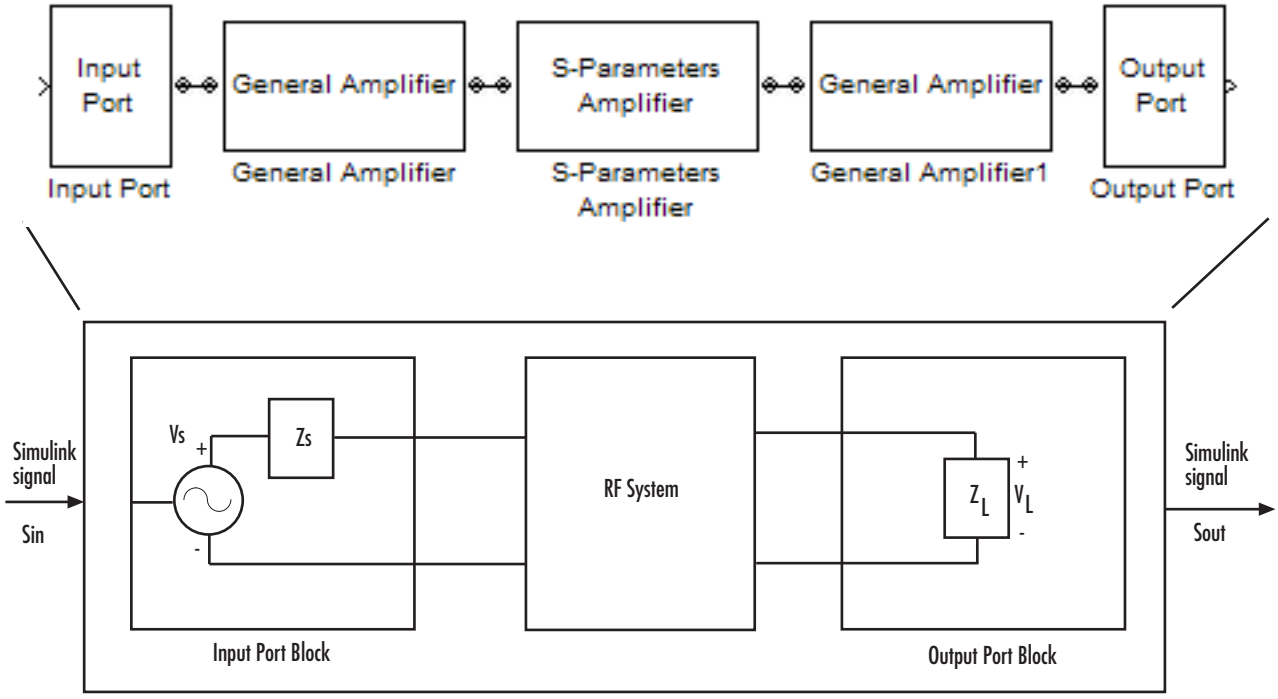


Baseband-Equivalent Model with Filter Shaping and Delay

Convert to and from Simulink Signals

Signal Conversion Specifications

When you simulate an RF model, the blockset must convert the mathematical Simulink signals to and from the physical modeling environment. The following figure shows the signals involved in the conversion.



Where:

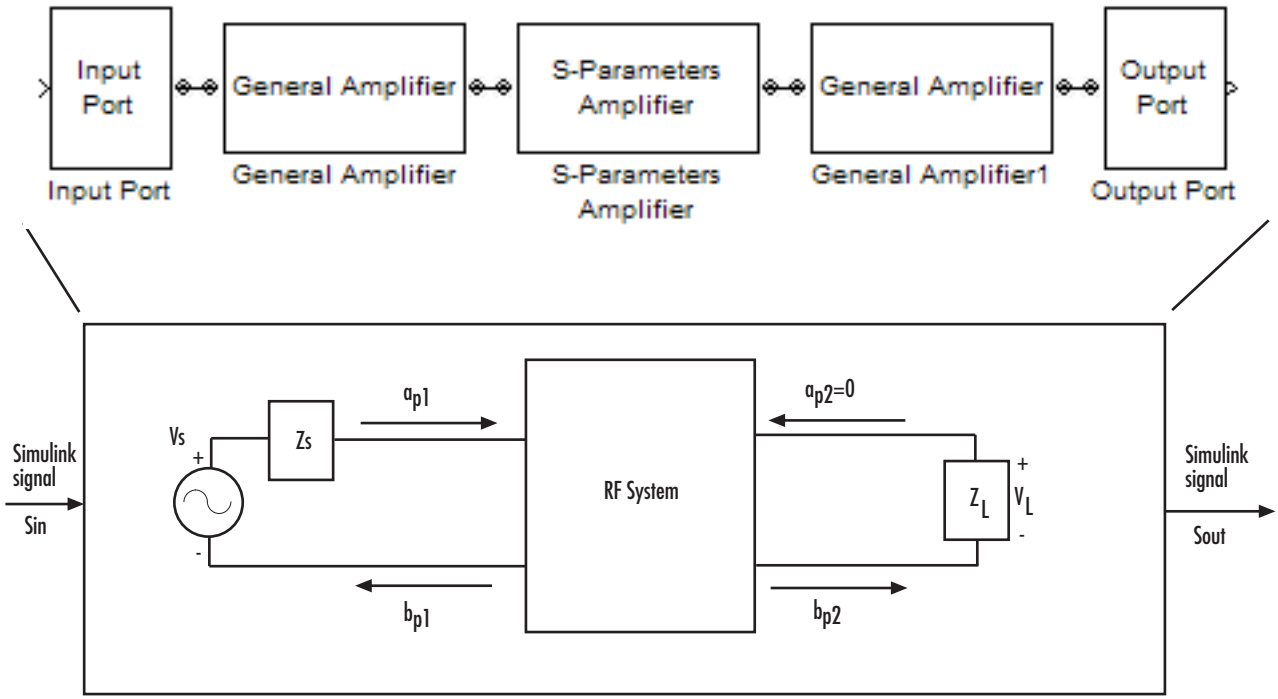
- Z_S is the **Source impedance (ohms)** parameter of the Input Port block.
- Z_L is the **Load impedance (ohms)** parameter of the Output Port block.

There are two options for interpreting the Simulink signal that enters the Input Port block:

- S_{in} is the incident power wave. For more information about this option, see “Interpret Simulink Signals as Incident Power Waves” on page A-32.
- S_{in} is the source voltage. For more information about this option, see “Interpret Simulink Signals as Source Voltages” on page A-34.

Interpret Simulink Signals as Incident Power Waves

The blockset provides the option to interpret the input Simulink signal, S_{in}, as the incident power wave, a_{p1}, at the first port of the RF system. The following figure shows the model for this interpretation.



In the figure, b_{p2} is the transmitted power wave at the second port of the RF system. This is the signal at the output of the Output Port block, S_{out}.

For a 2-port RF system, the incident and transmitted power waves are defined as:

$$a_{p1} = \frac{V_S}{2\sqrt{R_S}}$$

$$b_{p2} = \frac{\sqrt{R_L}}{Z_L} V_L$$

where:

- Z_S, the **Source impedance (ohms)** parameter of the Input Port block, is defined as:

  $Z_S = R_S + jX_S$
- Z_L, the **Load impedance (ohms)** parameter of the Output Port block, is defined as:

  $Z_L = R_L + jX_L$

Solving the power wave equations for S_{in} and S_{out} gives the following relationships:

$$S_{in} = \frac{V_S}{2\sqrt{R_S}}$$

$$S_{out} = \frac{\sqrt{R_L}}{Z_L} V_L$$

The implications of this interpretation are:

- $|S_{in}|^2$ is equal to the power available from the source, P_{avs}.
- $|S_{out}|^2$ is equal to the power delivered to the load, P_{out}.

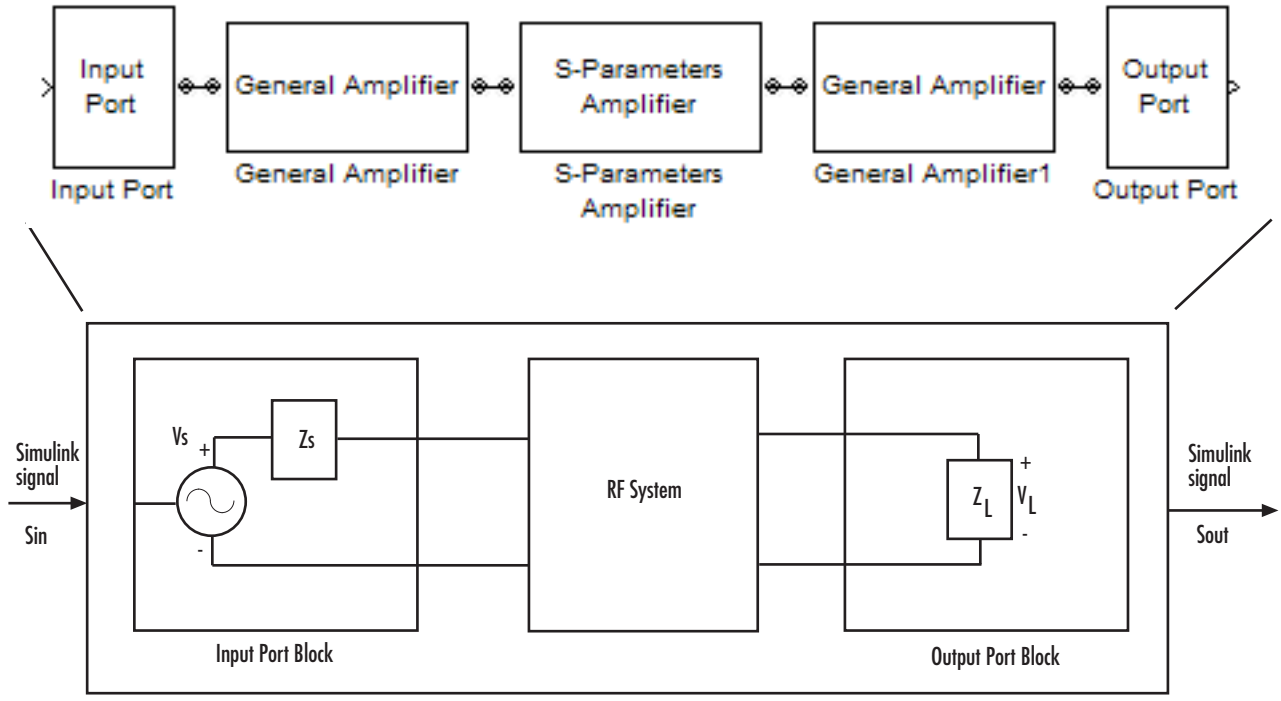
For a linear RF system, $P_{out} = G_t P_{avs}$ where G_t is the transducer power gain. As a result, the Simulink signals at the input and output of the RF system have the following relationship:

$$|S_{out}|^2 = G_t |S_{in}|^2$$

Note You can plot G_t from the Output Port block's **Visualization** tab.

Interpret Simulink Signals as Source Voltages

The blockset provides the option to interpret the input Simulink signal, S_{in}, as the source voltage, V_S, of the RF system. The following figure shows the model for this interpretation.



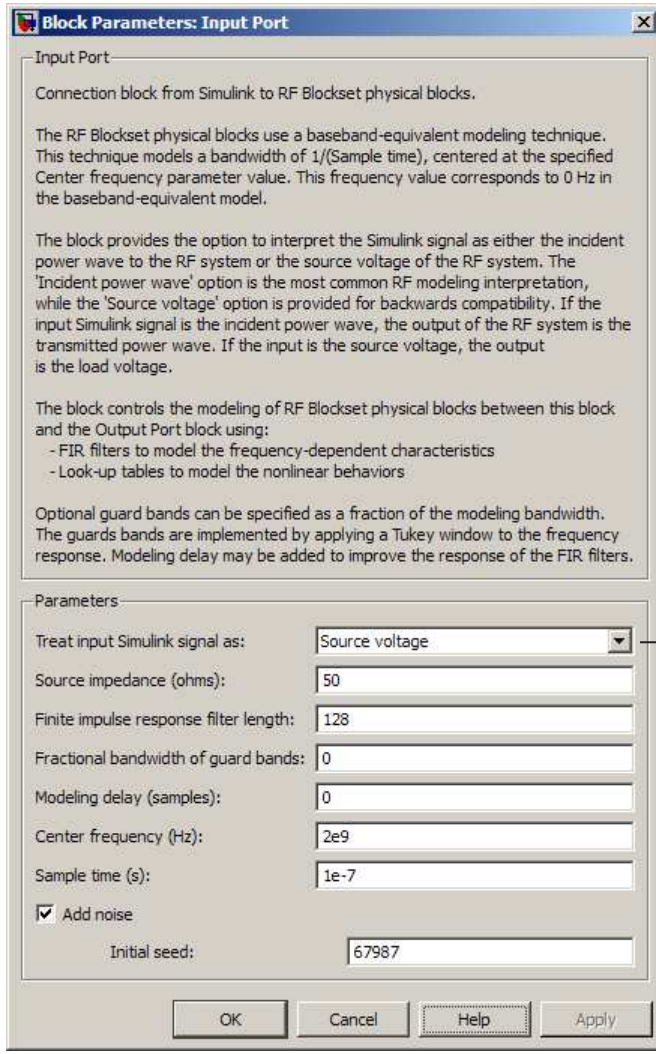
With this interpretation, the signal at the output of the Output Port block is the load voltage, V_L.

The blockset interpretation of the input Simulink signal as the source voltage, V_S, produces different results than the interpretation where the input Simulink signal is the incident power wave. When the source and load impedances are the same and real, the former interpretation produces 6 dB of loss compared to the latter.

Specify Input Signal Conversions

To specify the way in which the blockset interprets the input Simulink signal, you change the value of the **Treat input Simulink signal as** parameter in the Input Port dialog box. The available parameter values are:

- `Incident power wave` — Interpret the input signal as the incident power wave.
- `Source voltage` — Interpret the input signal as the source voltage.



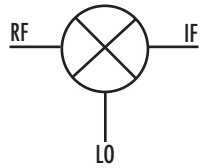
Change this parameter value to change the way the blockset interprets the input signal.

B

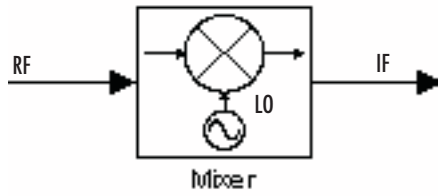
Model Mixers

2-Port Mixer Blocks

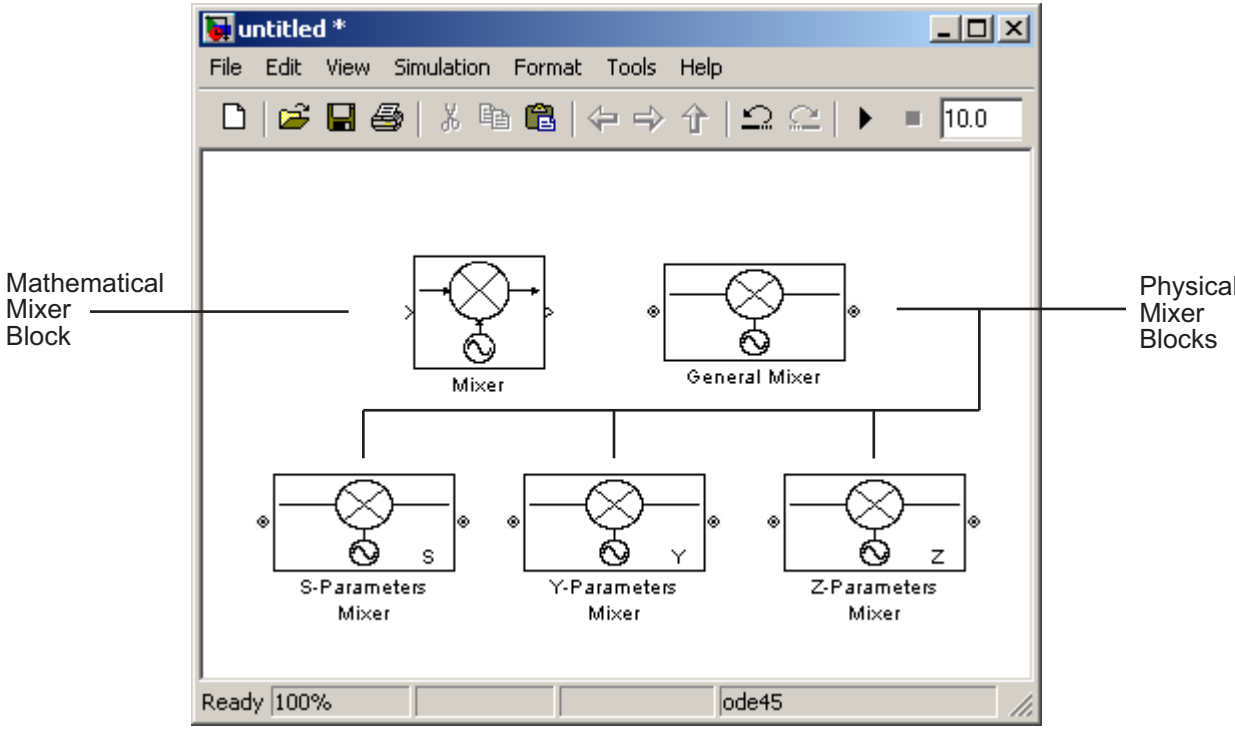
Typically, the block diagram of a mixer has three ports, as shown in the following representation of a downconversion mixer.



The mathematical and physical mixer blocks model both a mixer and a local oscillator, so they have only two ports.



The following figure shows the icons of the mixer blocks. The icons of all the mixer blocks show the internal local oscillator.



For the physical blocks, you can use the **LO frequency (Hz)** parameter to specify the local oscillator's frequency.

For more information, see the individual block reference pages.

Model a Mixer Chain

RF Blockset Equivalent Baseband software uses a baseband equivalent model to simulate RF components in the time domain. The blockset only models a band of frequencies around the carrier frequency of each component; the frequency band is determined by the following parameters of the corresponding Input Port block:

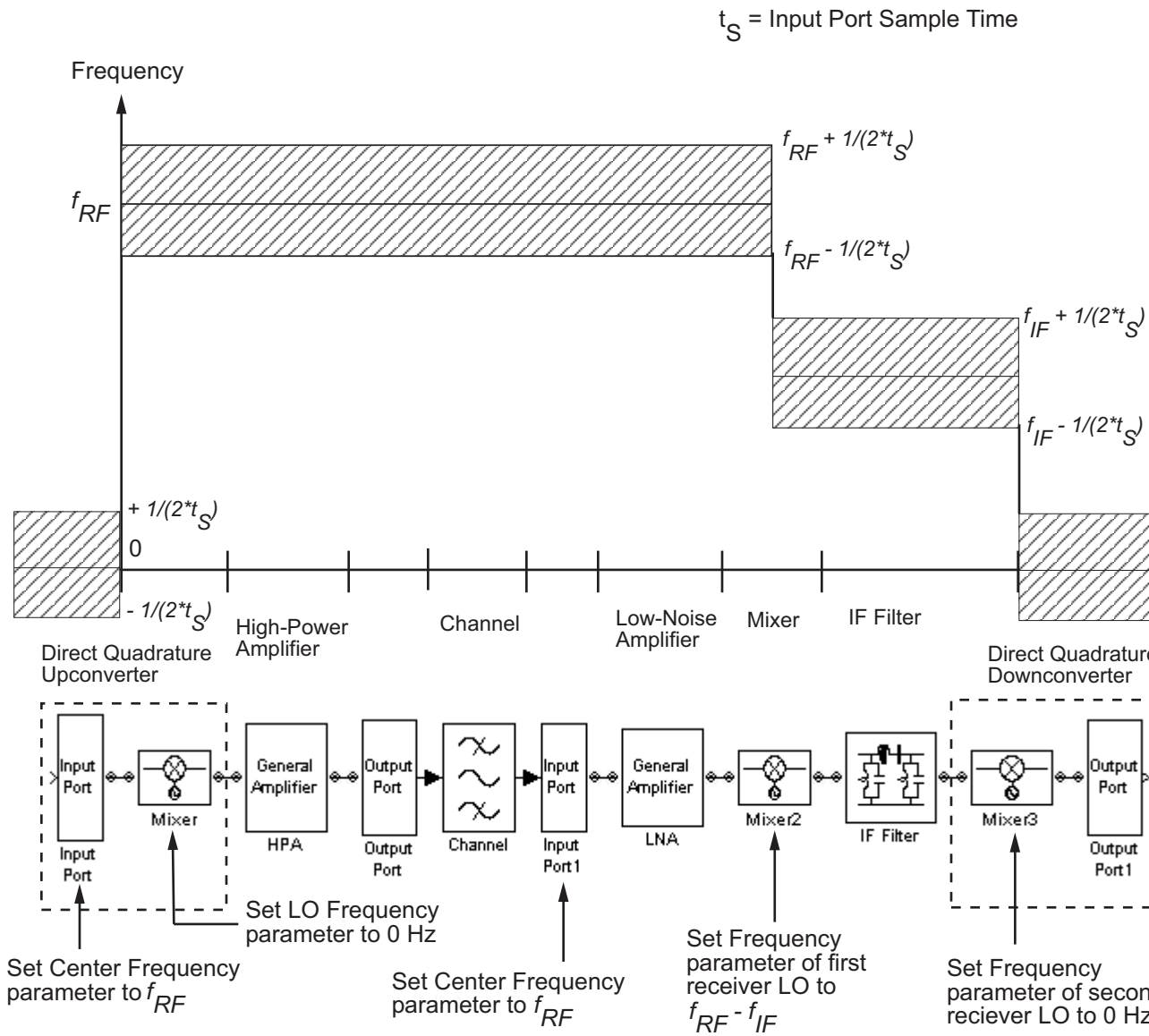
- Reciprocal of the **Sample time (s)**
- **Center frequency (Hz)**

When a mixer is present in a physical subsystem, it shifts the carrier frequency of the signal. This shift affects the frequencies that are used to create baseband equivalent model.

To illustrate how the mixer works, consider a typical RF mixer chain that consists of the following components:

- Direct Quadrature Upconverter
- High-Power Amplifier
- Channel
- Low-Noise Amplifier
- Downconverting Mixer
- IF Filter
- Direct Quadrature Downconverter

The following diagram shows these components and the band of frequencies that are simulated for each component. The signals at the input and output of the cascade are baseband complex. For the cascade, the diagram shows the real passband frequencies that are used to create the baseband-equivalent model, which is centered at zero. For a detailed explanation of how to use the blockset to model quadrature mixers, see “Quadrature Mixers” on page B-6.


t_S = Input Port Sample Time
Frequency
f_{RF} + 1/(2*t_S)
f_{RF}
f_{RF} - 1/(2*t_S)
f_{IF} + 1/(2*t_S)
f_{IF} - 1/(2*t_S)
+ 1/(2*t_S)
0
- 1/(2*t_S)
Direct Quadrature Upconverter
High-Power Amplifier
Channel
Low-Noise Amplifier
Mixer
IF Filter
Direct Quadrature Downconverter
Input Port
Mixer
Input Port
General Amplifier
HPA
Output Port
Output Port
Channel
Input Port
Input Port1
General Amplifier
LNA
Mixer2
IF Filter
Mixer3
Output Port
Output Port1
Set LO Frequency parameter to 0 Hz
Set Center Frequency parameter to f_{RF}
Set Center Frequency parameter to f_{RF}
Set Frequency parameter of first receiver LO to f_{RF} - f_{IF}
Set Frequency parameter of second reciever LO to 0 Hz

Quadrature Mixers

In this section...

Use RF Blockset Equivalent Baseband Software to Model Quadrature Mixers

RF Blockset software lets you model upconversion and downconversion quadrature mixers using Physical blocks. These mixers convert a complex baseband signal up to and down from the desired carrier frequency by mixing the real and imaginary parts of the signal with a cosine and sine of the same frequency.

Model Upconversion I/Q Mixers

You use the Input Port block to model the upconversion of in-phase/quadrature baseband signals to modulated signals at a finite real carrier frequency. The real component of the block input represents the in-phase signal. The imaginary component of the block input represents the quadrature signal.

To model a perfect quadrature upconversion mixer, use the Input Port block with the **Center frequency (Hz)** parameter set to the carrier frequency.

To model an imperfect quadrature upconversion mixer, use the Input Port block with the **Center frequency (Hz)** parameter set to the carrier frequency. Follow this block immediately by a mixer block with the **LO frequency (Hz)** parameter set to `0`. Specify imperfections as follows:

- S-parameters — Use S-parameters to specify imperfections such as frequency response. For a mixer, S_{21} describes the conversion gain, as explained in the Network Parameters section of the reference page for each mixer block. Use purely real and purely imaginary S_{21} parameters to represent multiplying the input signal by a pure cosine and a pure sine, respectively. Use a complex S_{21} parameter to represent multiplying the input signal by a combination of sine and cosine.

- Thermal noise — Use thermal noise to specify temperature-dependent random noise.
- Phase noise — Use the phase noise to specify noise to add to the angle component of the input signal.
- Nonlinearity — Use nonlinearity (specified as output power and phase as a function of input power and frequency in an AMP file or as third-order intercept point) to specify nonlinear mixer behavior as a function of input power.

Note If you specify a nonzero value for the local oscillator frequency of the mixer and set the **Type** parameter to `Upconverter`, the blockset converts the signal to a frequency above the center frequency. The final IF value is the sum of the Input Port center frequency and the mixer local oscillator frequency.

Model Downconversion I/Q Mixers

You use the Output Port block to model the downconversion of in-phase/quadrature modulated carrier signals to baseband signals. The real component of the block output represents the in-phase signal. The imaginary component of the block output represents the quadrature signal.

The finite real carrier frequency is set automatically as the sum of the center frequency of the Input Port block and the LO frequencies in any mixer blocks in the cascade.

Note In the cascade, upconversion mixers increase the carrier frequency and downconversion mixers decrease the carrier frequency.

The Output Port block models a perfect quadrature downconversion mixer. To model an imperfect quadrature downconversion mixer, precede the Output Port block immediately by a mixer block with the **LO frequency (Hz)** parameter set to `0`. Specify imperfections as follows:

- S-parameters — Use S-parameters to specify imperfections such as frequency response. For a mixer, S_{21} describes the conversion gain, as explained in the Network Parameters section of the reference page for each mixer block. Use purely real and purely imaginary S_{21} parameters to represent multiplying the input signal by a pure cosine and a pure sine, respectively. Use a complex S_{21} parameter to represent multiplying the input signal by a combination of sine and cosine.

- Thermal noise — Use thermal noise to specify temperature-dependent random noise.
- Phase noise — Use the phase noise to specify noise to add to the angle component of the input signal.
- Nonlinearity — Use nonlinearity (specified as output power and phase as a function of input power and frequency in an AMP file or as third-order intercept point) to specify nonlinear mixer behavior as a function of input power.

Note The mixer output frequency must be positive. This means that if you choose a downconverting mixer, the input carrier frequency f_{in} must be greater than the local oscillator frequency f_{lo}. Otherwise, an error appears.

Simulate I/Q Mixers

When you model an I/Q mixer in the blockset, the center frequency you specify in the Input Port block dialog is only used to build a complex-baseband equivalent model of the cascade that represents the mixer. The blockset simulates this model using a fixed time step equal to the sample time that you specify in the Input Port block dialog box.

To examine the model in the Simulink window:

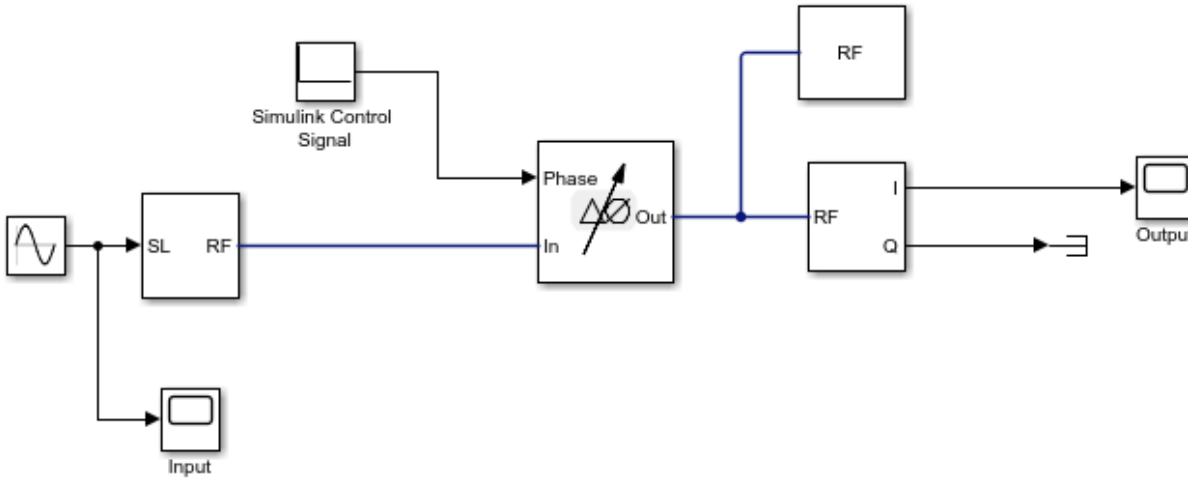
1. Select **Simulation > Update Diagram** to update the model diagram.
2. Right-click the Output Port block and select **Look Under Mask**.

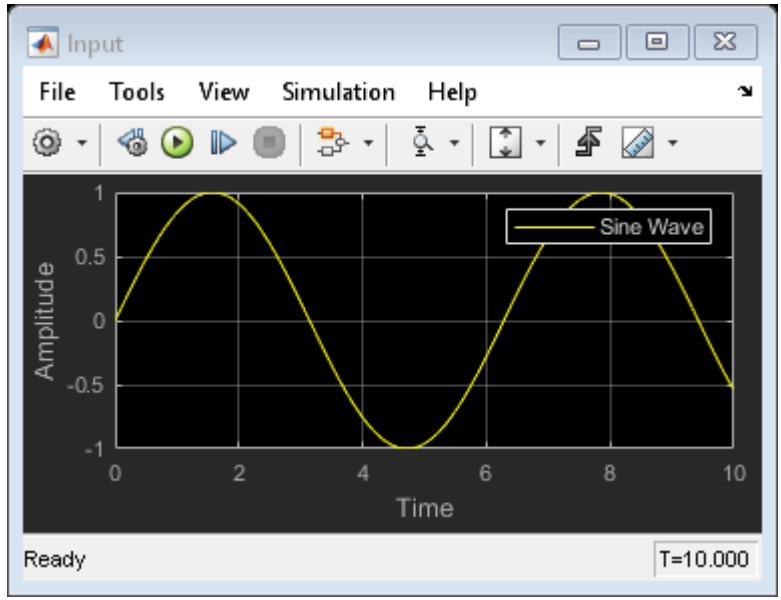
For more information on baseband-equivalent modeling, see “Model RF Components” on page 6-2.

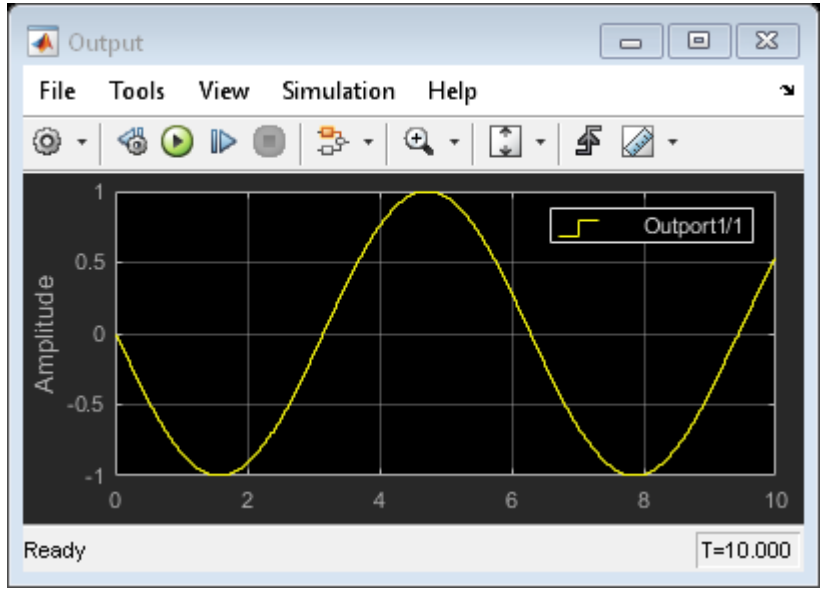
Examples

Vary Phase Of Signal During Simulation

Use the Variable Phase Shift block to shift the phase of a sine wave to 180 degrees.Use Repeating Sequence Stair block as a Simulink control signal to control the phase of the signal. To see the variation in phase to 180 degrees, first open and run the model. During simulation, change the value of the Simulink control signal to 90 degrees and see a change in phase in the Output Scope.

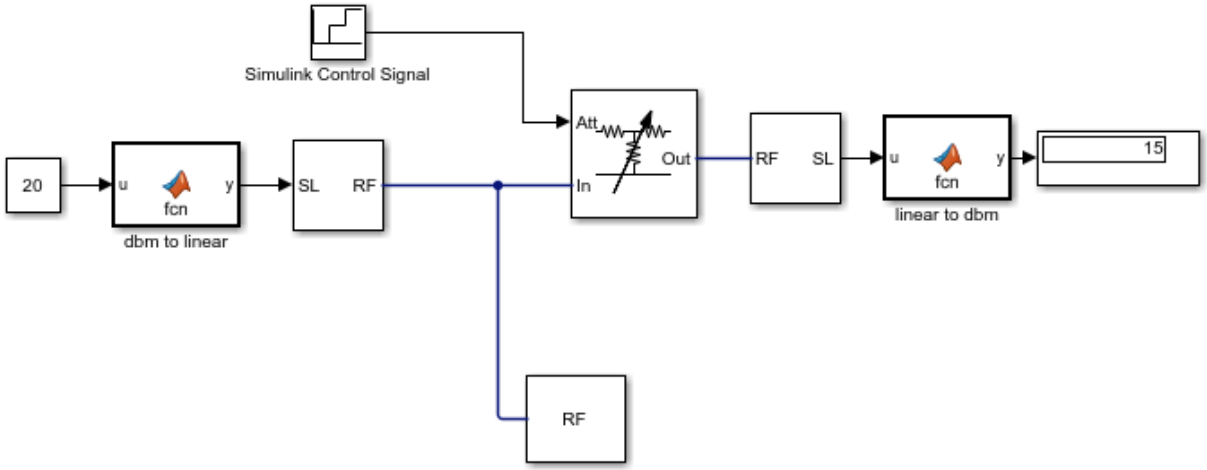



Input
File
Tools
View
Simulation
Help
Sine Wave
Amplitude
Time
1
0.5
0
-0.5
-1
0
2
4
6
8
10
Ready
T=10.000


Output
File
Tools
View
Simulation
Help
Outport1/1
Amplitude
1
0.5
0
-0.5
-1
0
2
4
6
8
10
Ready
T=10.000

Vary Attenuation of Signal During Simulation

Use the Variable Attenuator block to attenuate a 20 dB constant signal. Use the Repeating Sequence Stair block as a Simulink control signal to vary the attenuation of the signal. In this model,the signal attenuation varies between 5 and 10 dB during simulation. To see the variation in attenuation, open and run the model.

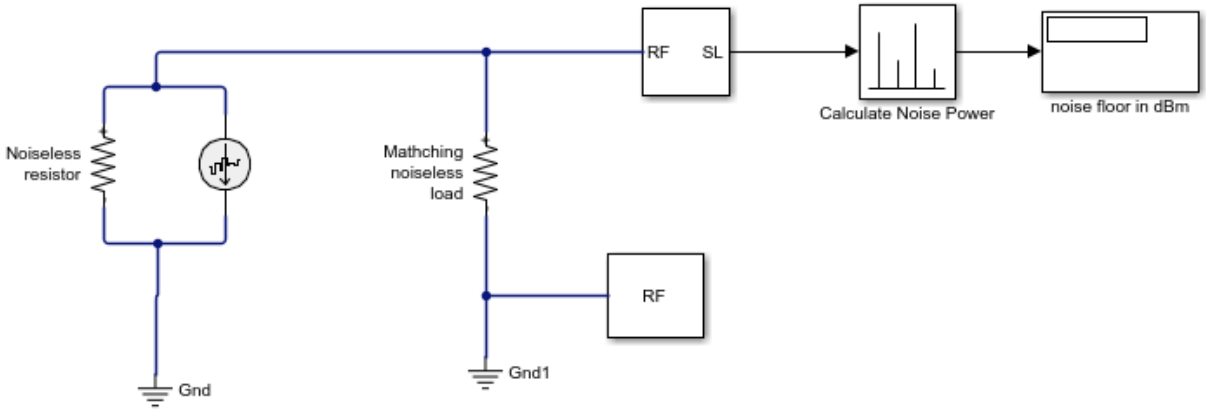


Explicitly Simulate Resistor Thermal Noise

Use the `Noise` block to calculate the classic thermal noise floor, `kT`, for a matched resistor circuit. Model configuration is as follows:

- Time step of the model is 1e-6 and frequency is 2 GHz.
- The `Resistor` noise source is modelled explicitly to make it noiseless. The resistance is 50 ohms. In the Resistor blocks, Simulate Noise is not selected.
- `Noise` current source parallel to the `Resistor` block models the noise. In the `Noise` block, the Source type is set to Ideal current to make it a current source. The Noise spectral density is defined as (4kT/R)(A2/Hz). The value of `k` is * The Masked block, Calculate Noise Power, calculates the noise floor as a standard deviation of the output signal.

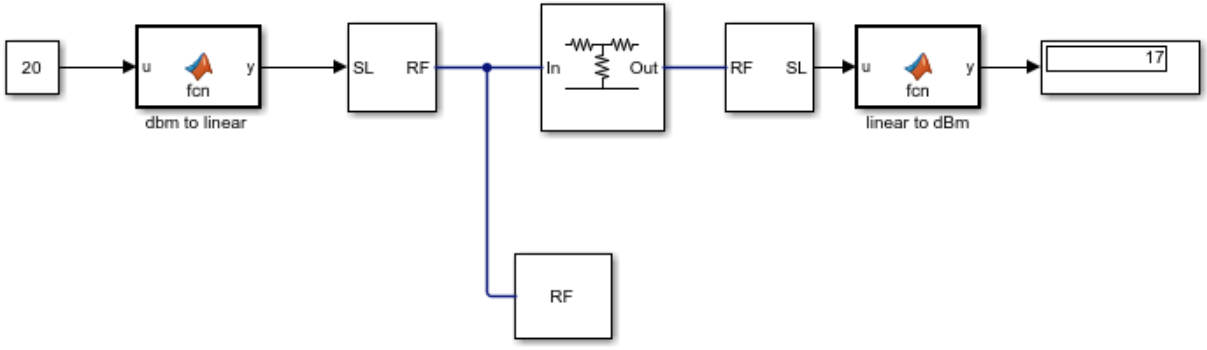
```
open_system('model_simrf_noise_source1')
```



To run the model, select Simulation > Run. With the bandwidth included using the Configuration block, noise power is in the range of -173.98 to 174.1 dBm

Attenuate Signal Power

Use the `Attenuator` block to attenuate a constant signal of 20 dB by 3 dB.



Demodulate Two-Tone RF Signal Using IQ Demodulator

Use the IQ Demodulator block to demodulate a two-tone RF signal to DC level. Observe the impairments in the demodulated output signal such as images due to gain imbalance, intermodulation distortion, and output third-order intercept (OIP3).

The two tones are at 10MHz and 15 MHz. The power of each tone is -30 dBm. The carrier frequency is 2 GHz.

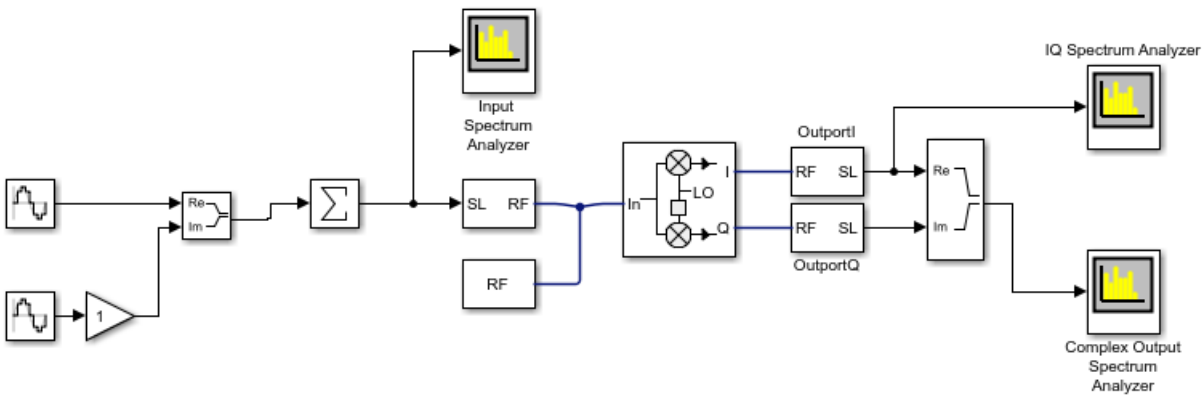
IQ Demodulator

The IQ Demodulator parameters are:

- Available power gain: 10 dB
- Local oscillator frequency: 2 GHz
- I/Q gain mismatch: 0.1 dB
- LO to RF Isolation: 90 dB
- Noise Figure: 6 dBm/Hz

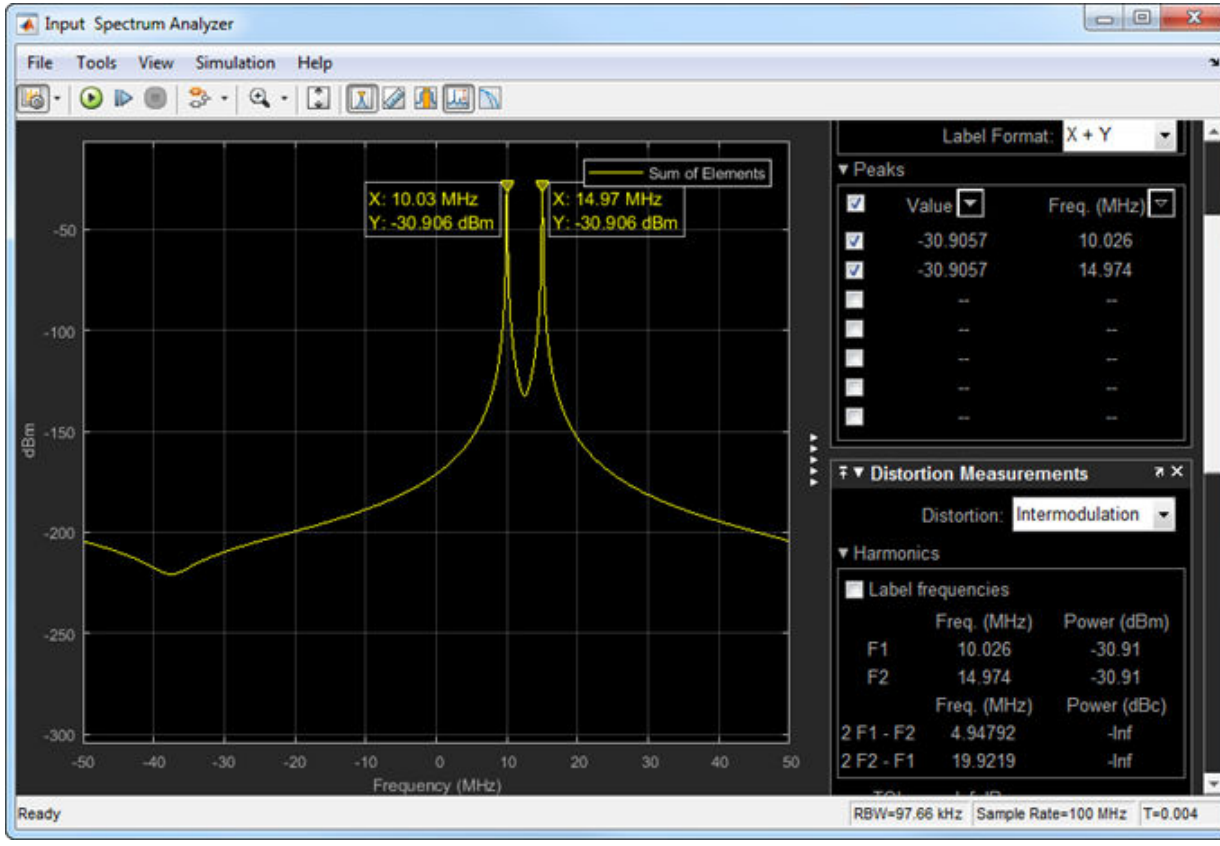
Open the model.

```
open('model_IQdemod')
```



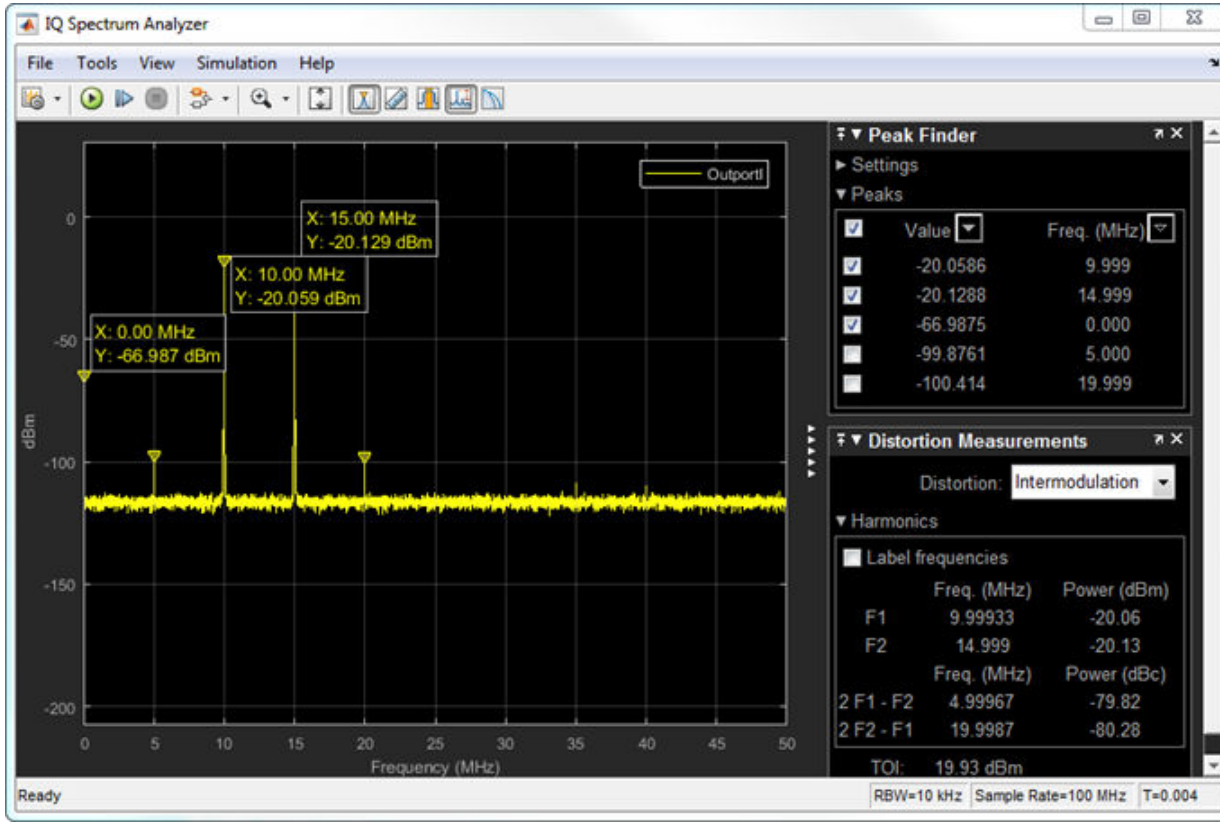
Run the model and observe the spectrum analyzers.

Input Spectrum Analyzer



In the input spectrum analyzer, you see the input RF signal with the two tones at 10 MHz and 15 MHz. The power level of each tone is -30 dBm. The carrier frequency is 2 GHz.

I/Q Spectrum Analyzer



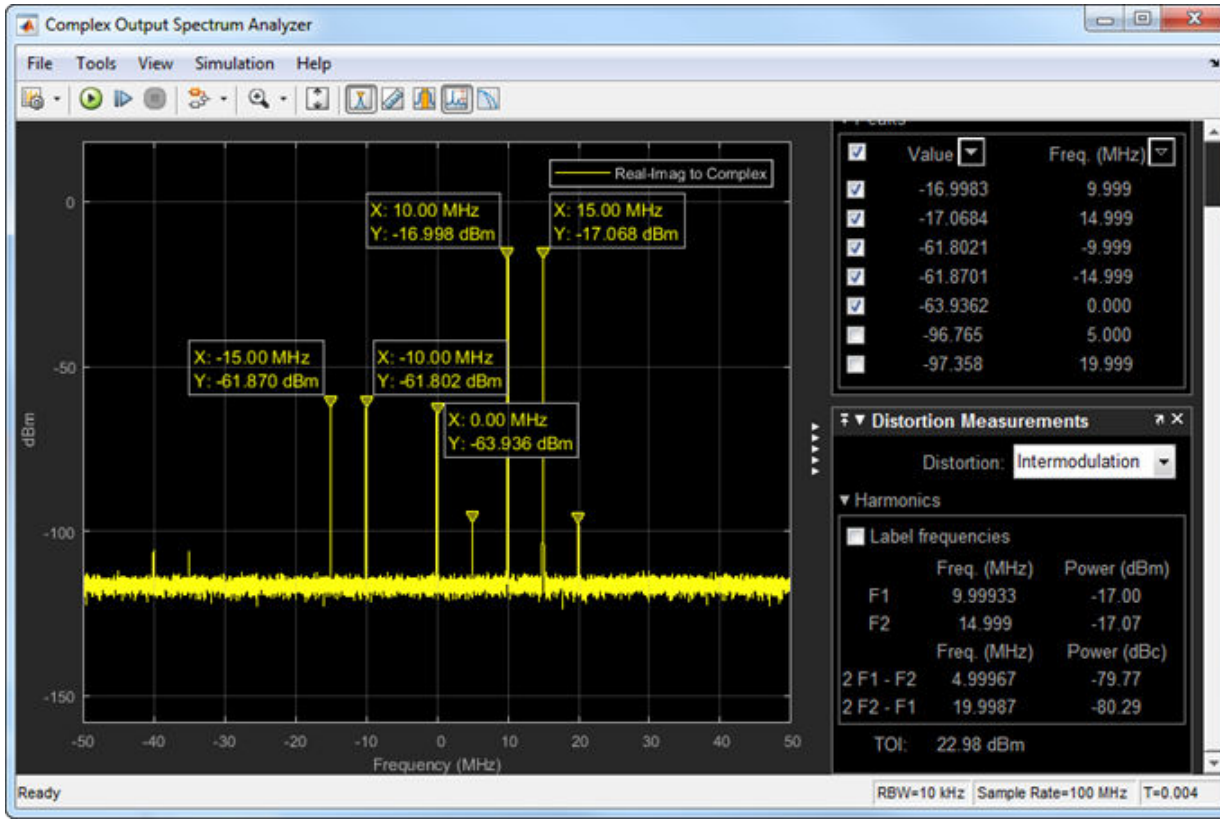
In the I/Q spectrum analyzer, you see the inphase part of the demodulated signal including the DC signal level and the two tones. The following formula gives you the DC power level of the signal:

$$DClevelI/Q = PowerLO - dBmIsolation + Gain$$

$$DclevelI/Q = 20 * log(1/sqrt(50)) + 30 - 90 + 10 = -67dBm$$

The output power level of the two tones are -20 dBm. You also see the OIP3 value (measured by the spectrum analyzer) at approximately 20 dBm.

Complex Output Spectrum Analyzer



In the complex output spectrum analyzer, you see the whole demodulated signal including the imaginary parts. The output power level of the two tones (10 MHz and 15 MHz) is -17 dBm.

Image Rejection Ratio

The images of the two tones are at -10 MHz and -15 MHz. The output power level of the images are -61.78 dBm. Image rejection ratio is given by the formula:

$$IMRR = [(gainimbalance)^2 + 1 - 2 * (gainimbalance)]/[(gainimbalance)^2 + 1 + 2 * (gainimbalance)]$$

$$GainImbalance = 10^(0.1/20) = 1.0116$$

$$IMRRdB = 10 * log10(3.3253e-05) = -44.78dB$$

$$Imagelevel = -17dBm - 44.78dB = -61.78dBm$$

Modulate Two-Tone DC Signal Using IQ Modulator

Use the IQ Modulator block to Modulate a two-tone DC signal to RF level. Observe the impairments in the modulated output signal such as images due to gain imbalance, intermodulation distortion, and output third-order intercept (OIP3).

The two tones are at 10MHz and 15 MHz.The power of each tone is -30 dBm. The carrier frequency is 0 GHz.

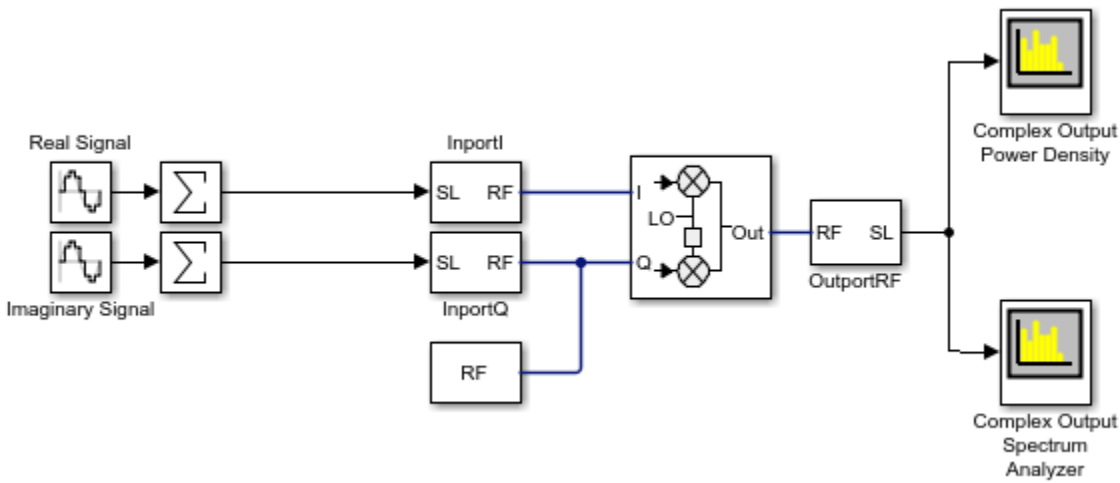
IQ Modulator

The IQ modulator parameters are :

- Available power gain: 10 dB
- Local oscillator frequency: 2 GHz
- I/Q gain mismatch: 0.1 dB
- LO to RF Isolation: 90 dB
- Noise Floor: -160 dBm/Hz
- IP3: 10 dBm

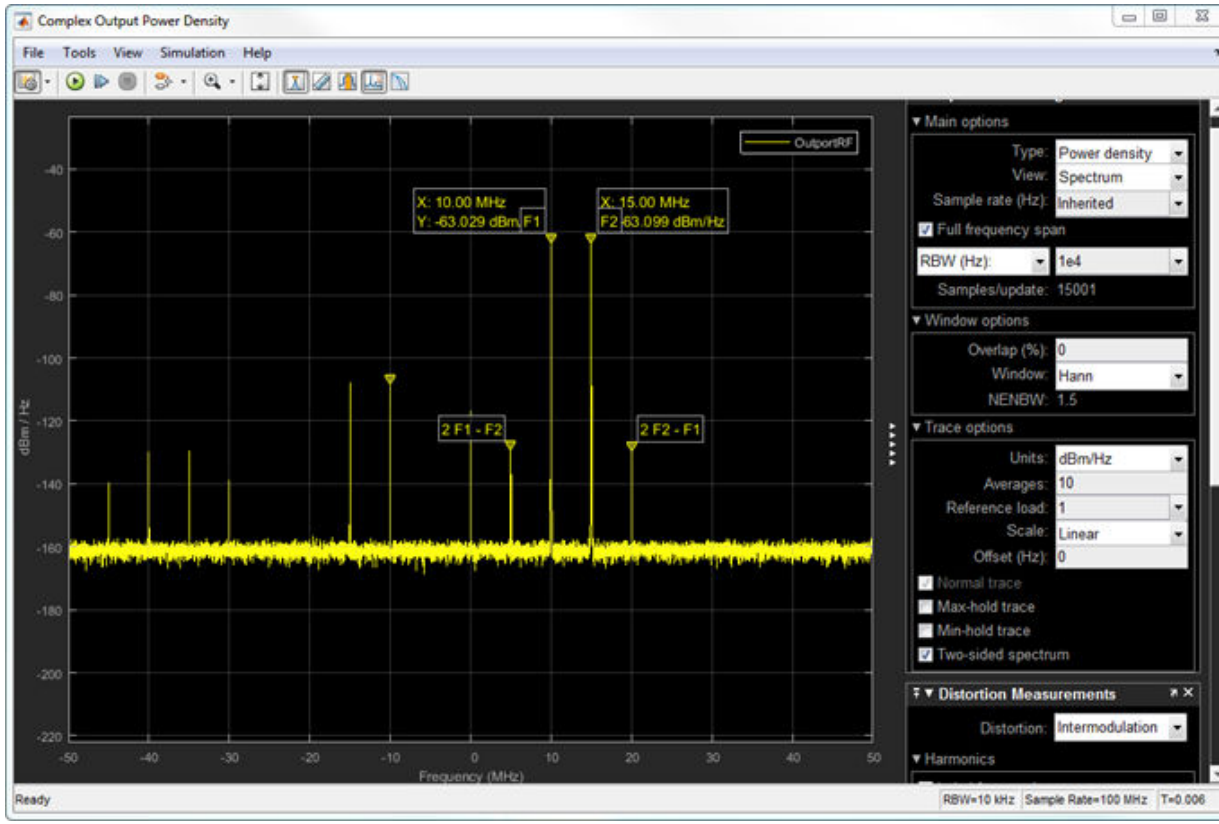
Open the model.

```
open('model_IQmod')
```



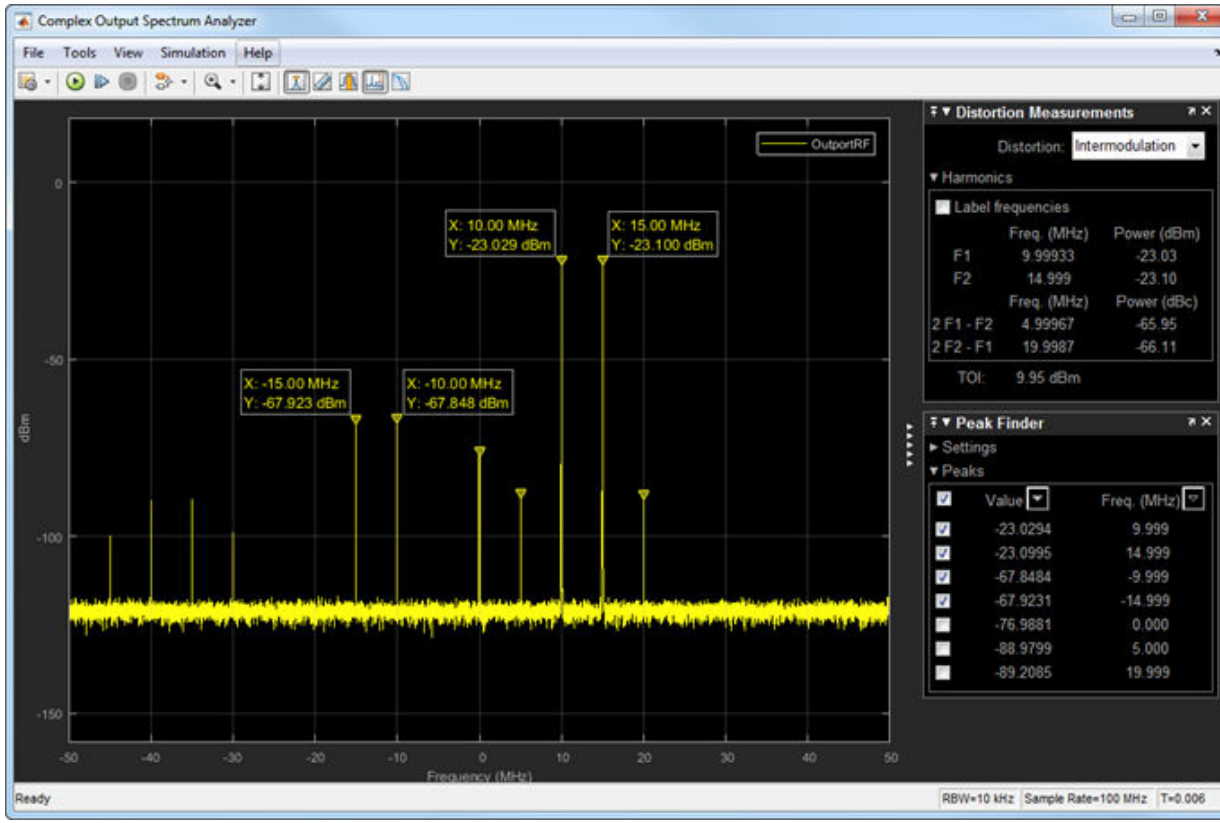
Run the model and observe the spectrum analyzers.

Complex Output Power Density



In the complex output power density spectrum analyzer, you see the noise floor of the signal at -160 dBm/Hz.

Complex Output Spectrum Analyzer



In the complex output spectrum analyzer, you see the whole modulated signal including the imaginary parts. The output power level of the two tones (10 MHz and 15 MHz)is -20 dBm.

$$Outputpowerlevel = Inputpowerlevel + gain = -30dBm + 10dB = -20dBm$$

The output third-order intercept(OIP3) is at 10 dBm. The spectrum analyzer measures this value.

Image Rejection Ratio

The images of the two tones are at -10 MHz and -15 MHz. The output power level of the two images are -67.8 dBm. Image rejection ratio is given by the formula:

$$IMRR = [(gainimbalance)^2 + 1 - 2 * (gainimbalance)]/[(gainimbalance)^2 + 1 + 2 * (gainimbalance)]$$

where,

$$GainImbalance = 10^(0.1/20) = 1.0116$$

$$IMRRdB = 10 * log10(3.3253e - 05) = -44.78dB$$

The image power level is given by the formula:

$$Imagelevel = -23dBm - 44.78dB = -67.78dBm$$

Spot Noise data In Amplifiers and Effects On Measured Noise Figure

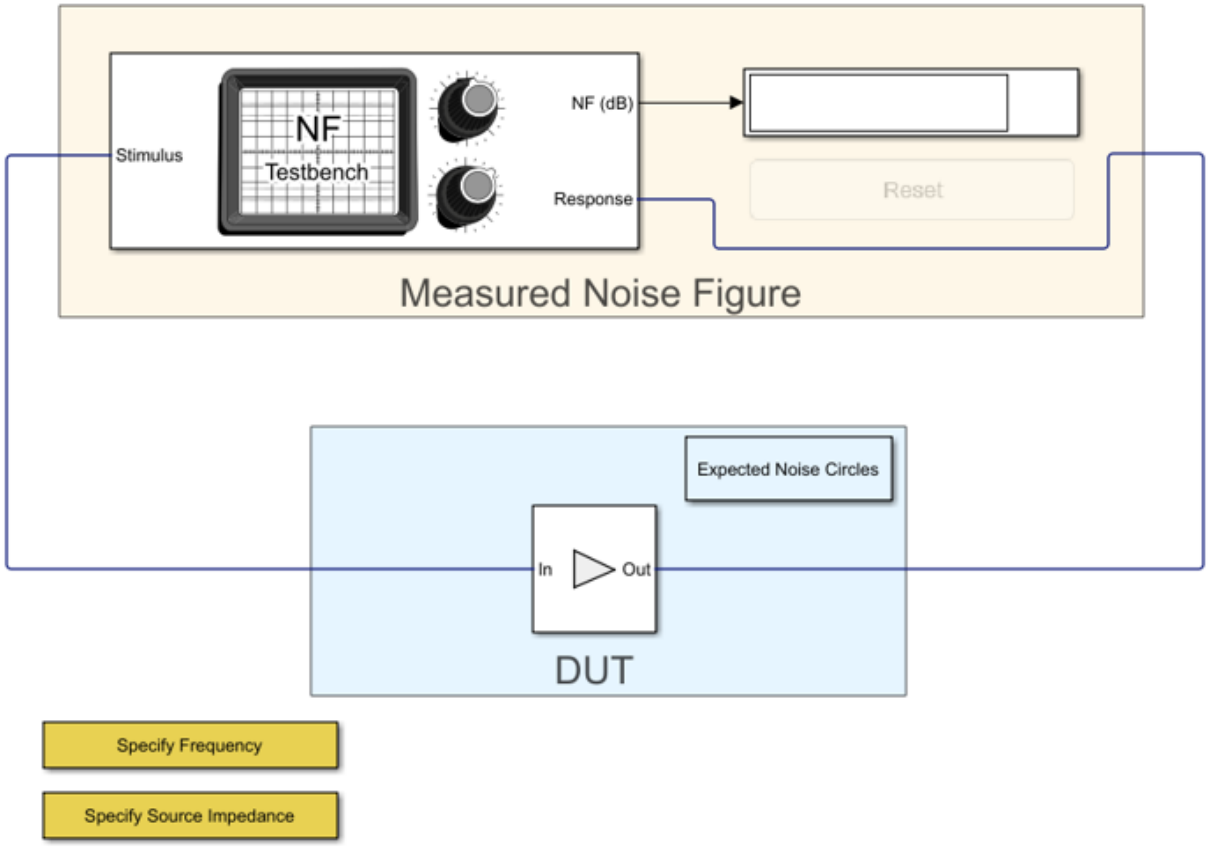
This example shows a test bench model to describe the noise introduced by a 2-port device.

The spot noise data parameters, `Fmin`, Γ `opt` , and `Rn`, fully describe the noise introduced by a 2-port device. These parameters along with the source impedance, `Zs` uniquely determine the measured noise figure of the device. You can use noise circles plotted on a Smith chart to show interaction between `Zs` and the noise figure.

Measure Noise Figure In RF Blockset

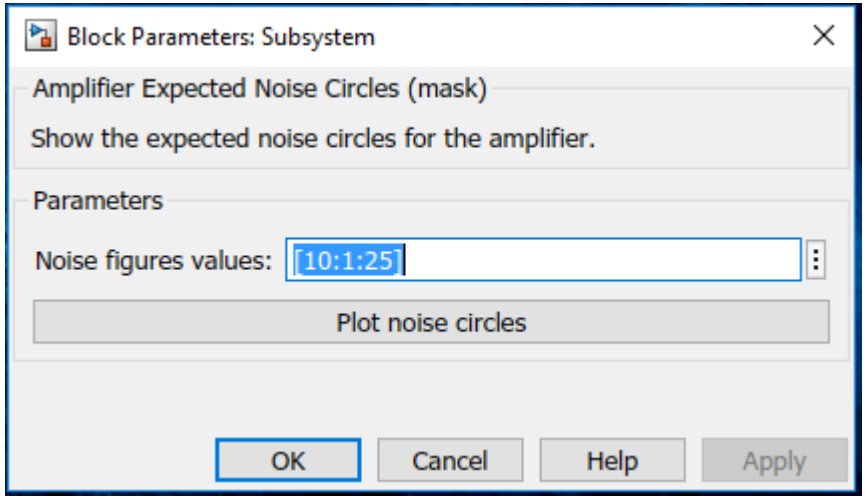
The model `Noise_figure_ex` simulates a simple noise figure measurement. In this model, device under test comprises of a single amplifier. To open the model,

```
open_system('Noise_figure_ex.slx')
```

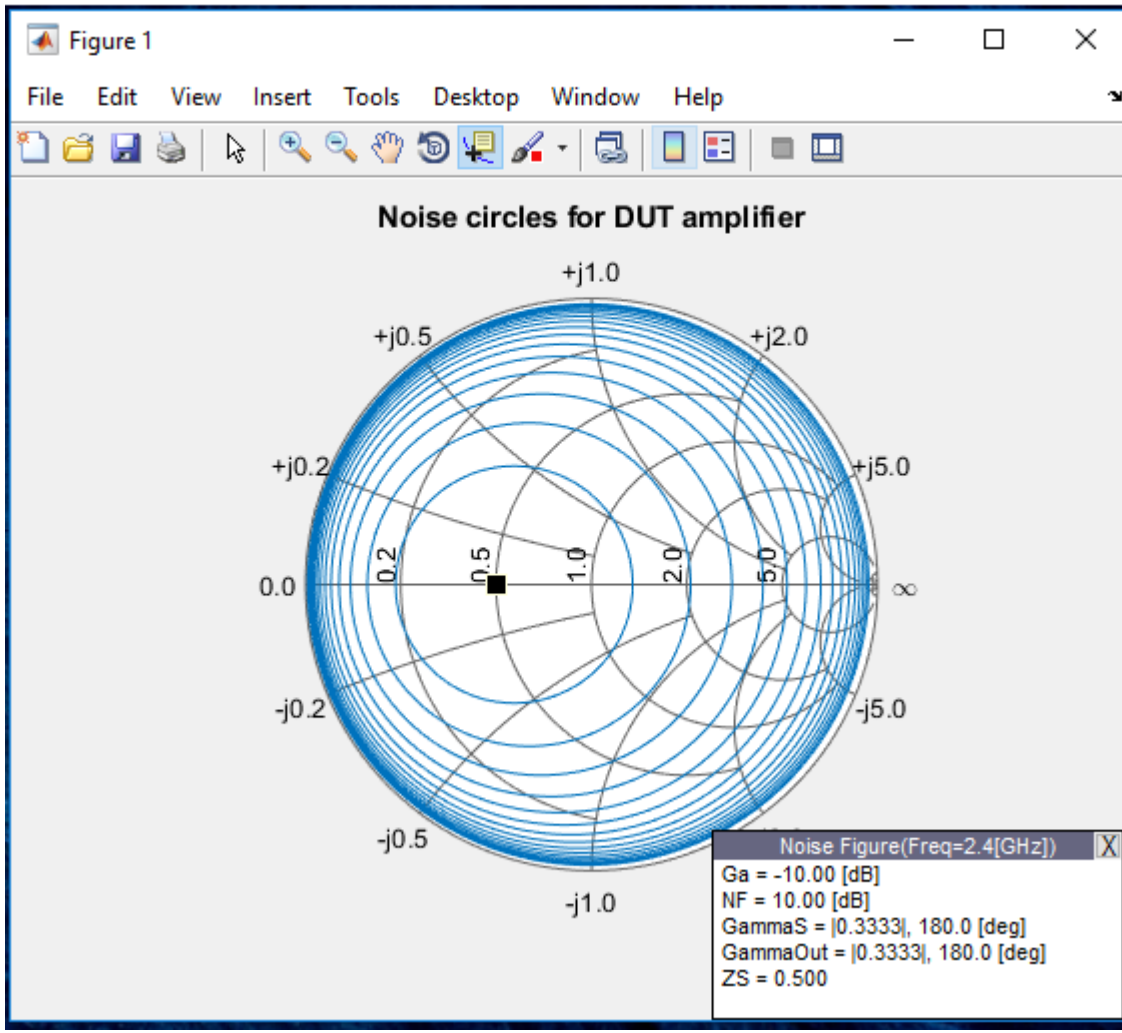


Please click on the **Open Script** button before running the model. To run the model, select **Simulation > Run**.

You see that the displayed value settles down at 10.0 dB of the measured noise figure. In this testbench, the amplifier parameters represent a simple attenuator of 10 dB matched to 25 Ohms at both input and output. Due to thermal equilibrium, the expected noise figure of the attenuator is 10 dB when matched, corresponding to the measured value. To gain a broader view of the expected noise figure values for the amplifier when the source impedance deviates from the matched value of 25 Ohm, double-click the `Expected Noise Circles` subsystem placed in the vicinity of the amplifier:



Click `Plot noise circles` ,to bring up a figure showing a Smith chart with circles corresponding to the noise figure values specified above the button:



The values shown in the Smith chart represent the expected noise figures obtained theoretically from the parameters specified in the amplifier [1]. The Smith chart is interactive and you can place the data cursor on any circle to view the corresponding noise figure, source impedance (normalized to the reference impedance of the amplifier,

Z0), and other RF properties. The initial data cursor position corresponds to a point on a noise circle that is closest to the source impedance specified. To control the complex value of this source impedance, double-click the `Specify Source Impedance` subsystem, specify the desired value in the edit box.

To validate that the simulated measured noise figure corresponds to theoretical values, specify a reference impedance from theSmith chart, using the `Specify Source Impedance` subsystem. Run the model again.

Measuring Other RF Systems

You can replace the amplifier in the model by any other RF Blockset system and measure its noise figure. In case the system is frequency depended, you can change the frequency for the measurements by double-clicking the `Specify Frequency` subsystem and specifying the desired frequency.This frequency is also used for the amplifier noise circles plot.

The plotted expected noise circles apply to the `Amplifier` block alone. The plotted circles capture correctly all types of data inputs specified in the amplifier, including s2p based network and noise data. Note that the `Available Gain` shown in the data window of the Smith chart is based on the original data and ignores inaccuracies introduced by the amplifier modeling method. The plotting fails if a block named `Amplifier` does not exist in the model.

Using Other RF Blockset Blocks

Three additional ways to implement the attenuator specified in the amplifier are:

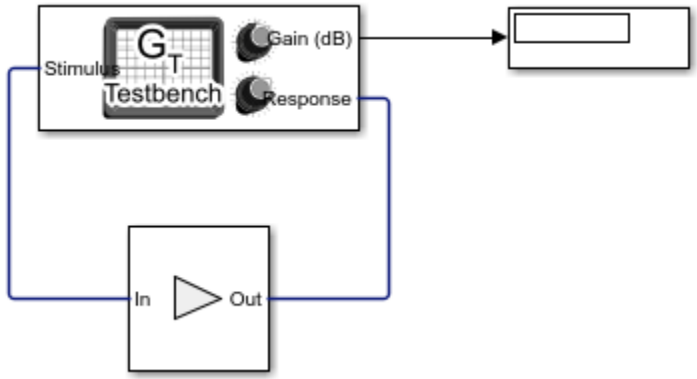
1. Use an RF Blockset `Attenuator` block with attenuation of 10dB, with input and output impedances set to 25 Ohms.
2. Implement the attenuator using three resistors arranged in a T or π topology.
3. Use an `S-parameter` block with the same Scattering matrix used in the amplifier. Select `Simulate noise` in the block. Simulating noise in a passive S-parameter block accounts for the resistive noise introduced by the S-parameters.

Transducer Gain TestBench

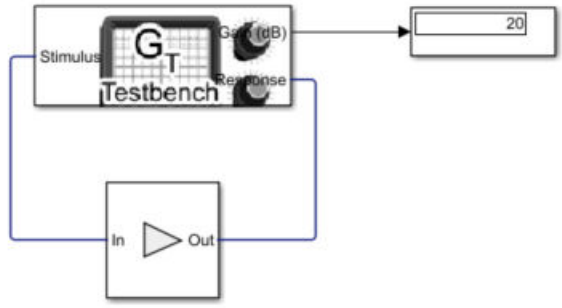
Use the `Transducer Gain Testbench` block to verify the gain of an `Amplifier` block.

Open the model and change the gain of the amplifier block to 20 dB.

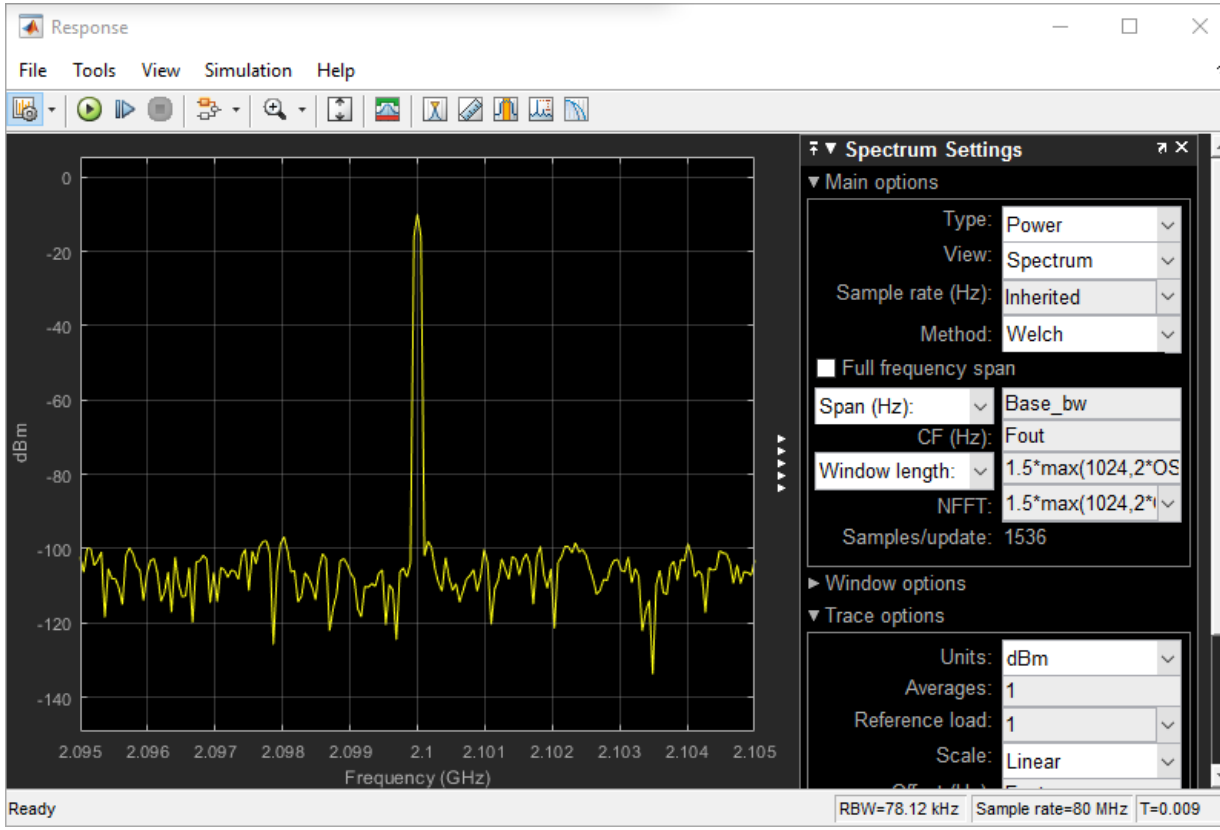
```
open_system('transducer_gain_example.slx')
```



Run the model. You will see that the output display shows a gain of 20 dB.



The scope response shows a gain of 20 dBm at 2.1 GHz, which is the specified frequency in the testbench dialogue box.

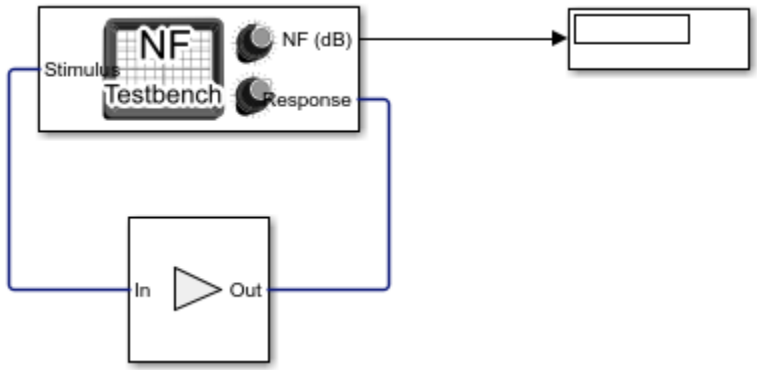

Response
File Tools View Simulation Help
Spectrum Settings
Main options
Type: Power
View: Spectrum
Sample rate (Hz): Inherited
Method: Welch
Full frequency span
Span (Hz): Base_bw
CF (Hz): Fout
Window length: 1.5*max(1024,2*OS
NFFT: 1.5*max(1024,2*
Samples/update: 1536
Window options
Trace options
Units: dBm
Averages: 1
Reference load: 1
Scale: Linear
0
-20
-40
-60
-80
-100
-120
-140
dBm
2.095 2.096 2.097 2.098 2.099 2.1 2.101 2.102 2.103 2.104 2.105
Frequency (GHz)
Ready
RBW=78.12 kHz
Sample rate=80 MHz
T=0.009

Noise Figure Testbench

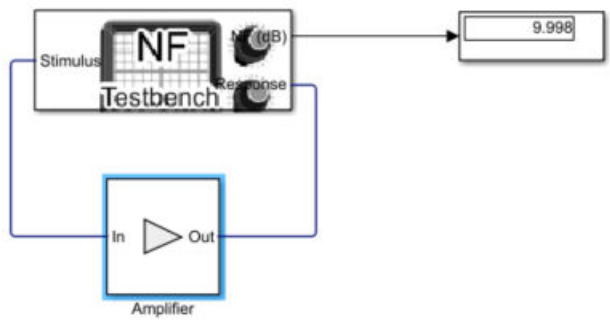
Use the `Noise Figure Testbench` block to verify the noise figure of an `Amplifier` block.

Open the model and change the noise figure of the amplifier block to 10 dB. In the Noise Figure Testbench block, clear `Show Response spectrum`.

```
open_system('noise_figure_example.slx')
```



Run the model. You will see that the display shows a noise figure close to 10 dB.

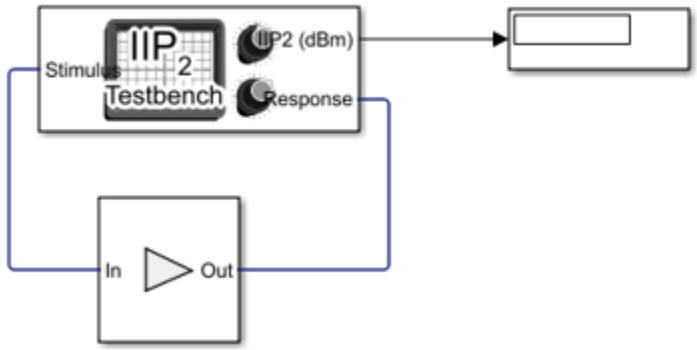


IIP2 Testbench

Use the `IIP2 Testbench` block to verify the input second order intercept (iip2) of an `Amplifier` block.

Open the model.

```
open_system('iip2_example.slx')
```



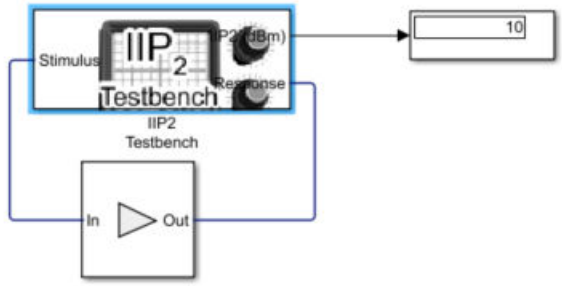
Open the amplifier block and change the following under **Nonlinearity**:

- Intercept points convention = Input
- IP2 = 10 dB

In the IIP2 Testbench block, clear the following:

- Simulate noise (both stimulus and DUT internal)
- Show Response spectrum

Run the model. You will see that the display shows a iip2 value of 10 dB.

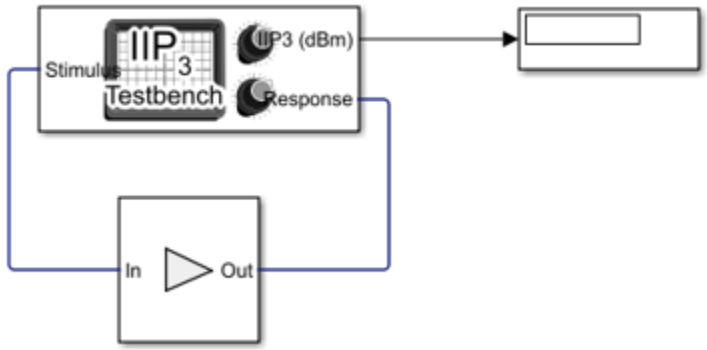

Stimulus
IIP
2
Testbench
IIP2
Testbench
10
In
Out

IIP3 Testbench

Use the `IIP3 Testbench` block to verify the input third order intercept (IIP3) of an `Amplifier` block.

Open the model.

```
open_system('iip3_example.slx')
```



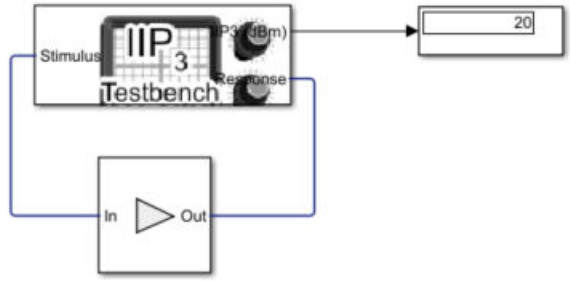
Open the Amplifier block and change the following under **Nonlinearity**:

- Intercept points convention = Input
- IP3 = 20 dB

In the IIP3 Testbench block, clear the following:

- Simulate noise (both stimulus and DUT internal)
- Show Response spectrum

Run the model. You will see that the display shows a IIP3 value of 20 dB.

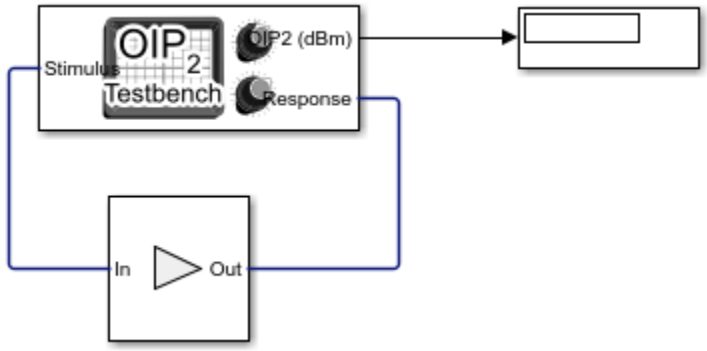

IIP3
Testbench
Stimulus
Response
20
In
Out

OIP2 Testbench

Use the `OIP2 Testbench` block to verify the output second order intercept (OIP2) of an `Amplifier` block.

Open the model.

```
open_system('oip2_example.slx')
```



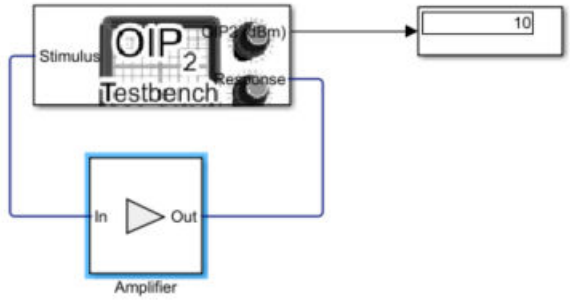
Open the amplifier block and change the following under **Nonlinearity**:

- Intercept points convention = Output
- IP2 = 10

In the OIP2 Testbench block, clear the following:

- Simulate noise (both stimulus and DUT internal)
- Show Response spectrum

Run the model. You will see that the display shows a OIP2 value of 10 dB.

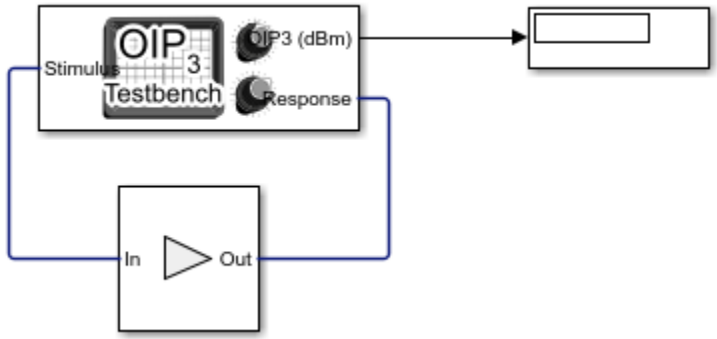

OIP
2
Testbench
Stimulus
Response
10
In
Out
Amplifier

OIP3 Testbench

Use the `OIP3 Testbench` block to verify the output third order intercept (oip3) of an `Amplifier` block.

Open the model.

```
open_system('oip3_example.slx')
```



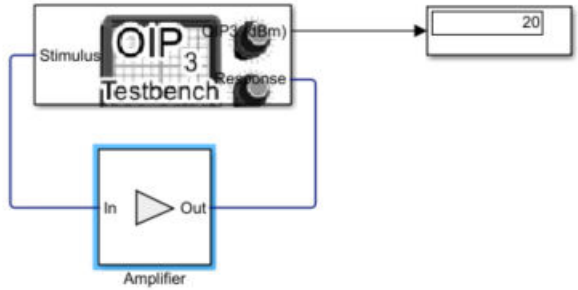
Open the amplifier block and change the following under **Nonlinearity**:

- Intercept points convention = Output
- IP3 = 20

In the OIP3 Testbench block, clear the following:

- Simulate noise (both stimulus and DUT internal)
- Show Response spectrum

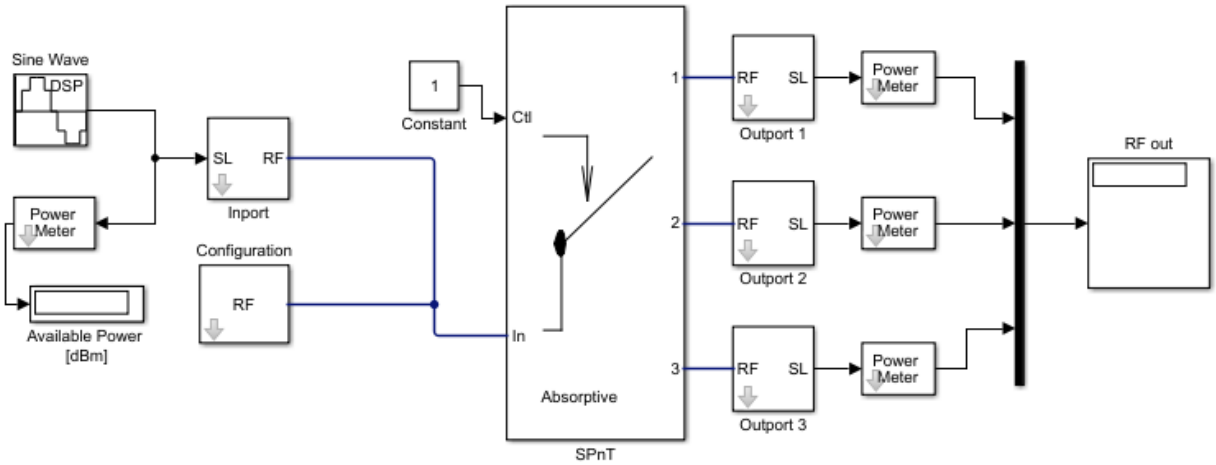
Run the model. You will see that the display shows a oip3 value of 20 dB.


OIP
3
Testbench
Stimulus
Response
20
In
Out
Amplifier

Single Pole Triple Throw Switch

Use the SPnT block to create a single pole triple throw switch to switch a signal between three outputs.

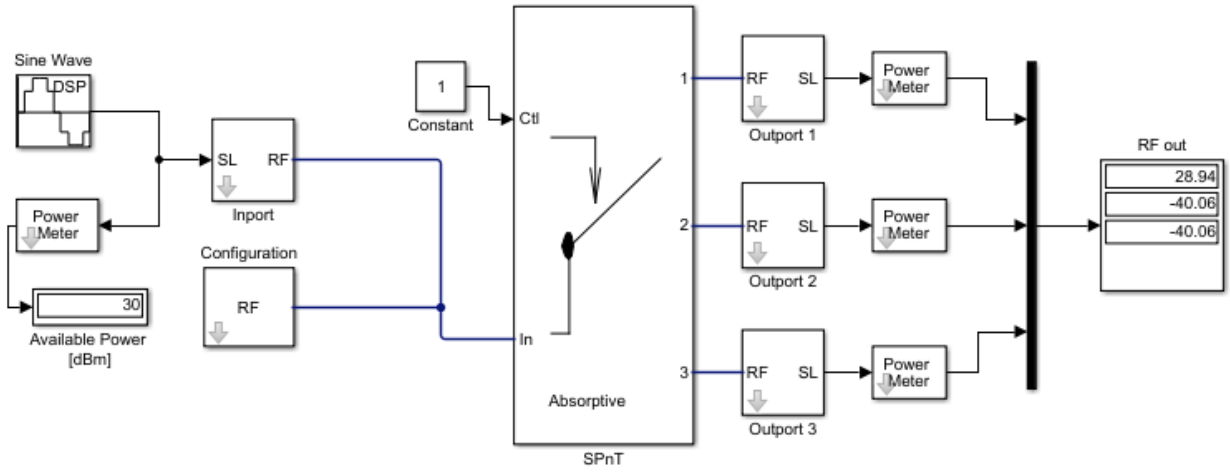
Open the model.



The model consists of:

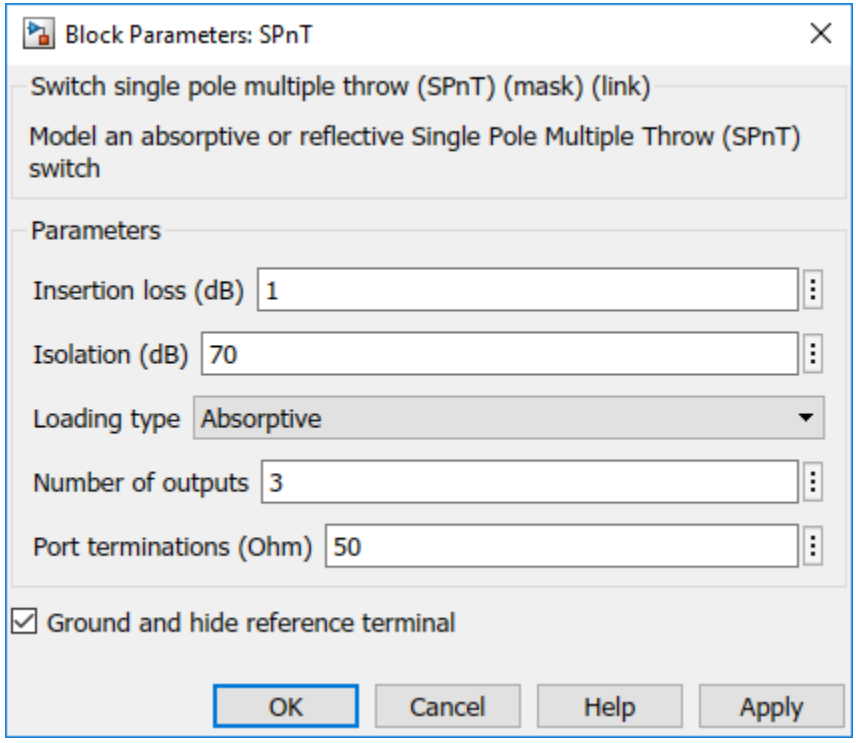
- Sine Wave block to generate a sine wave of amplitude 1
- Power Meter block to determine the power of the sine wave. This is the input power. The input power is 30 dB.
- Inport and Configuration blocks connected to the "In" port of the SpnT block.
- A Constant block connected to the "Ctl" port of the SPnT block. This signal is used to control the switch outputs.
- SPnT switch block with 3 output ports.
- Outport 1, Outport 2, Outport 3, and Power meter blocks to calculate the power of each output signal.
- Display block to display the three outputs.

Run the model.



The Display block shows that the signal power is available through the first port of the switch as the "Ctl" port is set to 1.

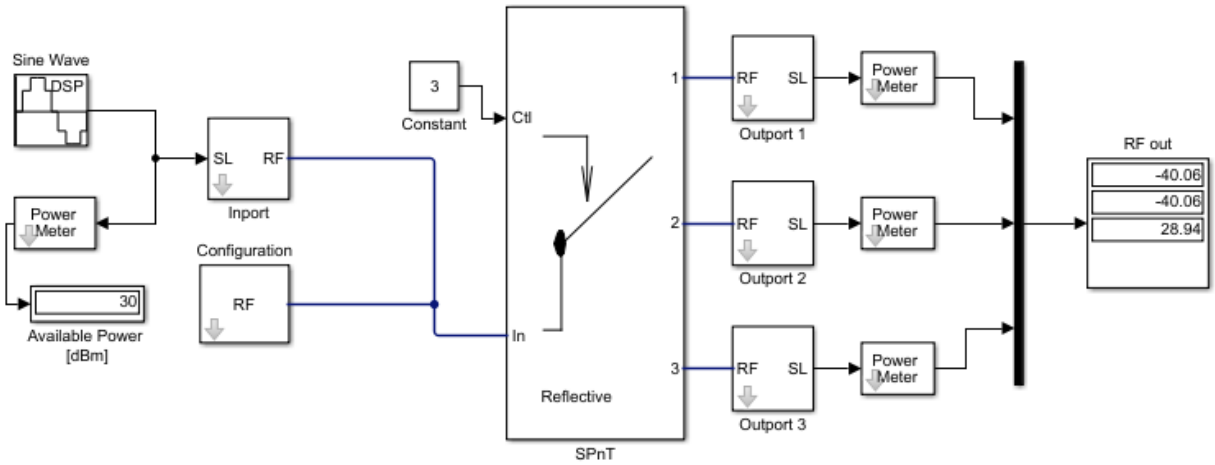
Open the SPnT block to see set values. Currently the switch is set to "Absorptive" using the "Load Type" parameter.



Change the "Load Type" value to "Reflective".

Change the value of the Constant block to 3.

Run the model again.



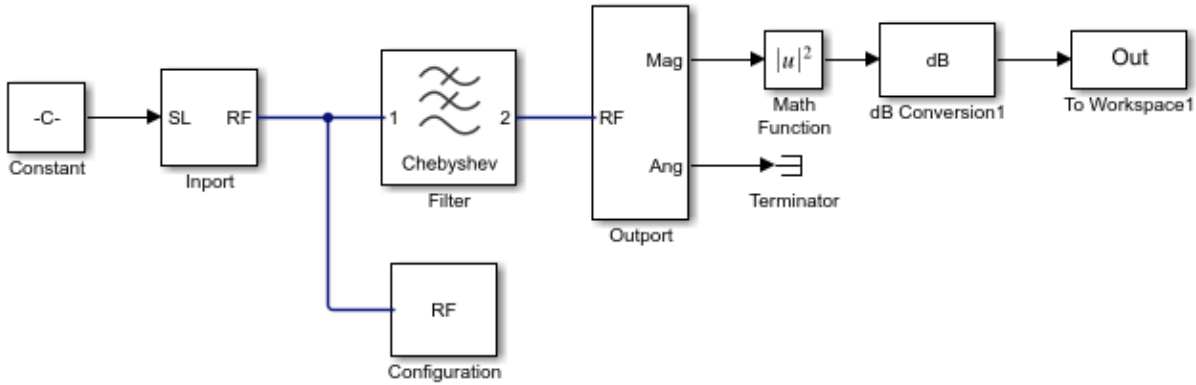
The Display block shows that the signal power is available through the third port of the switch as the "Ctl" port is set to 3.

Frequency Response of Lowpass Chebyshev Filter

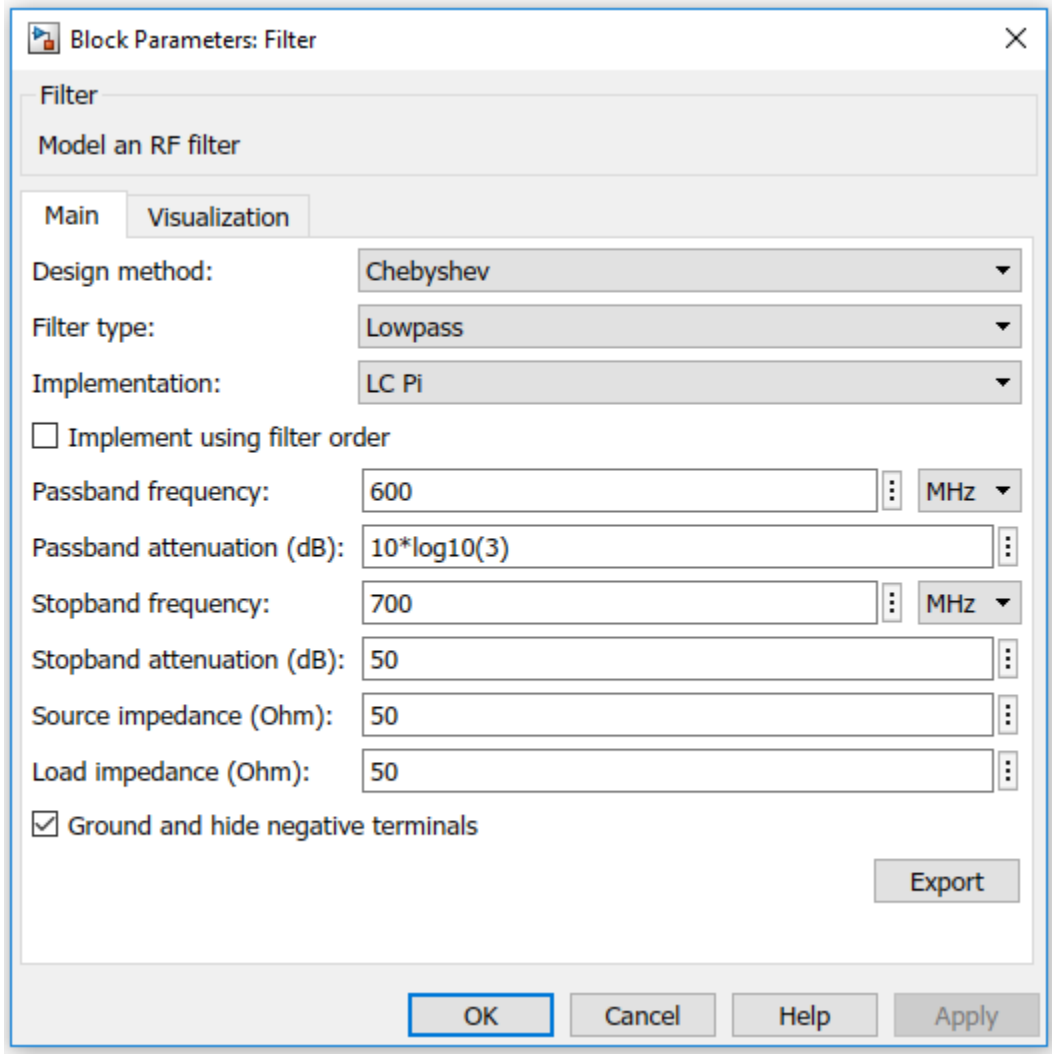
Use the `Filter` block to study the frequency response of a lowpass Chebyshev filter.

From the MATLAB command prompt, open the model.

```
open_system('ex_simrf_filter_lowpass_cheby_resp')
```



The `Constant` block sets the amplitude of the 201 carrier signals to ones(1, 201). The `Inport` block generates the 201 carrier frequencies for the mask value of logspace (7, 9, 201). Generate an 11th order lowpass LC Pi Chebyshev filter by setting appropriate block parameters in the `Filter` block.



The output signal from the `Filter` block is fed into the `Outport` block. The `Outport` block is configured to give both magnitude and angle of the signal. The angle output is terminated using the `Terminator` block. The magitude output is squared and converted to dB using `Math Function` and `dB Conversion` blocks.

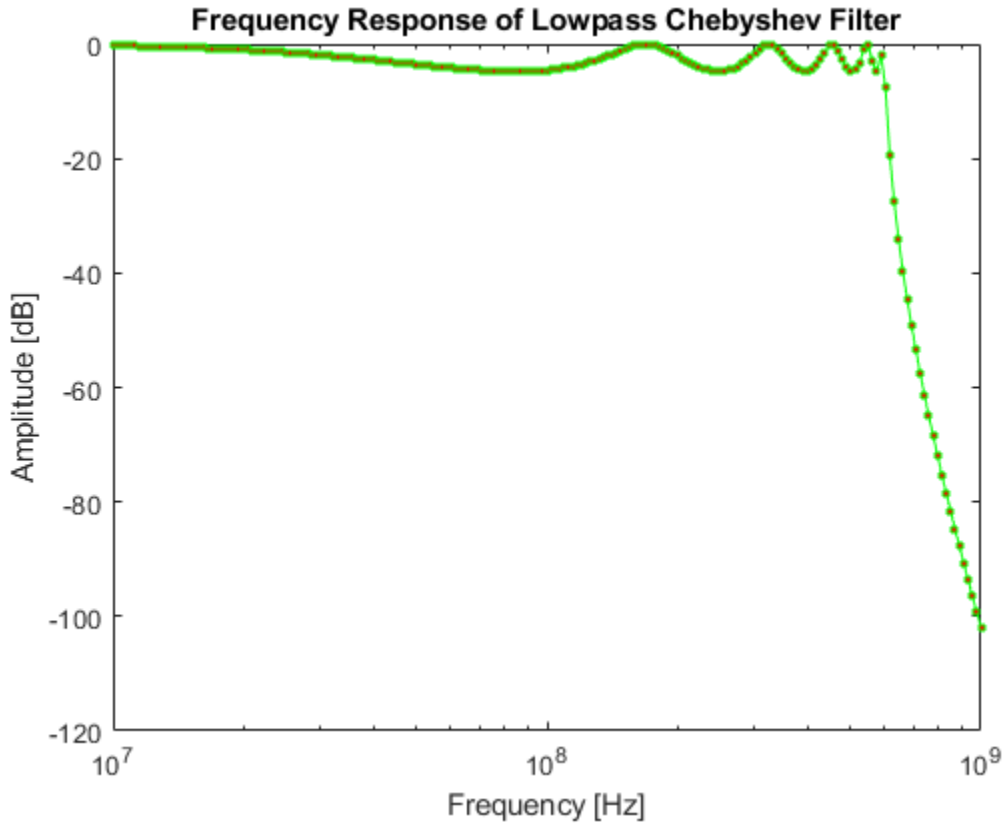
To run the model, select Simulation > Run. You can also use the following command:

```
sim('ex_simrf_filter_lowpass_cheby_resp')
```

The model creates an `Out` array in the MATLAB workspace. Since the simulation stop time is set to 0, the frequency response corresponds to the steady state solution.

To plot the frequency response, use the following commands in the MATLAB command window.

```
figure
freq = logspace(7,9,201);
h = semilogx (freq, Out, '-gs', 'LineWidth',1, 'MarkerSize',3, 'MarkerFaceColor','r');
xlabel('Frequency [Hz]');
ylabel('Amplitude [dB]');
title('Frequency Response of Lowpass Chebyshev Filter');
```



You can also use the Plot button in the Visualization tab of the `Filter` block parameters. Set the Frequency points to logspace (7, 9, 201) and X-axis scale to Logarithmic to achieve a similar plot.