# Cosimulation Using the Virtuoso® AMS Designer Simulator and The MathWorks MATLAB®/Simulink®

**Product Version 6.1.6**
**March 2014**

# Contents

# 1

# AMS-MATLAB/Simulink Cosimulation

**Note:** *AMS-MATLAB/Simulink cosimulation* stands for cosimulation using the Virtuoso® AMS Designer simulator and The MathWorks MATLAB®/Simulink®. *ADE* stands for the Virtuoso Analog Design Environment.

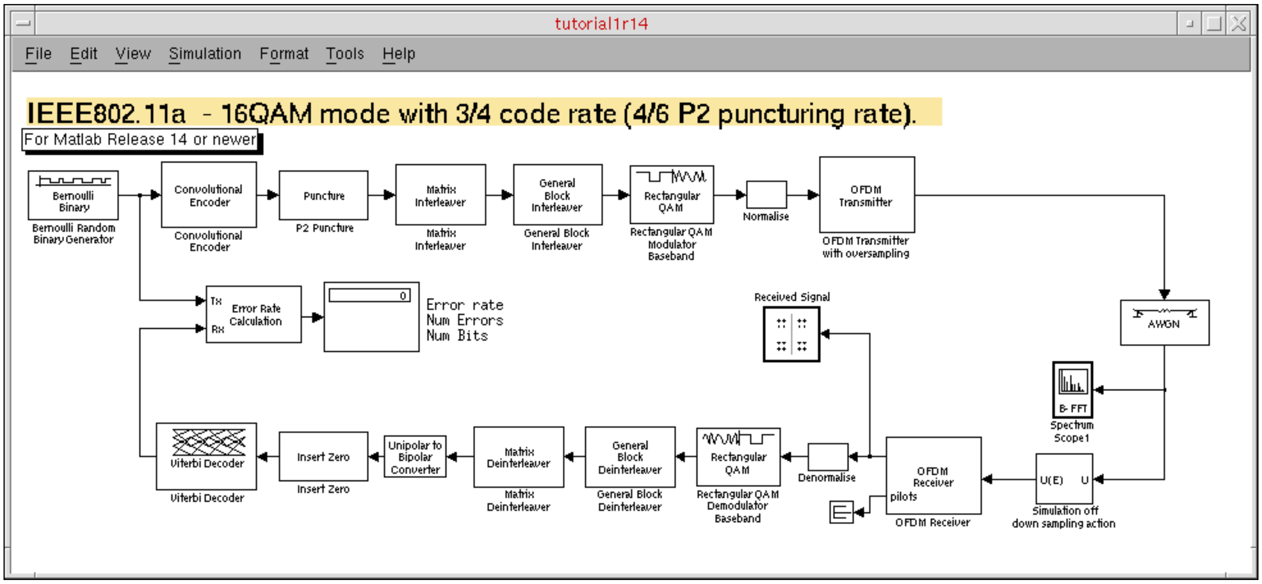
*Important*

Use IC 6.1.3, IUS 8.1 or later, and MATLAB704R14 or MATLAB R2007b or MATLAB R2008a for this tutorial. The estimated time to complete this tutorial is about one hour.

See the following topics for tutorial details:

# Introducing AMS-MATLAB/Simulink Cosimulation

Concept engineering and system-level simulation tools such as MATLAB®/Simulink® support the specification of high-level system concepts in the early stages of a design.




The picture above shows the top-level schematic for the system-level model of a wireless LAN (the IEEE 802.11a demo). The transmitter blocks encode and modulate binary random data and send the orthogonal frequency-division multiplexing (OFDM) signal to a white Gaussian noise channel model. The receiver blocks demodulate and decode the channel output. Finally, the system compares the received bits with the original bit stream to compute the bit error rate.

**Note:** The tutorial database libraries include the IEEE 802.11a demo designs.

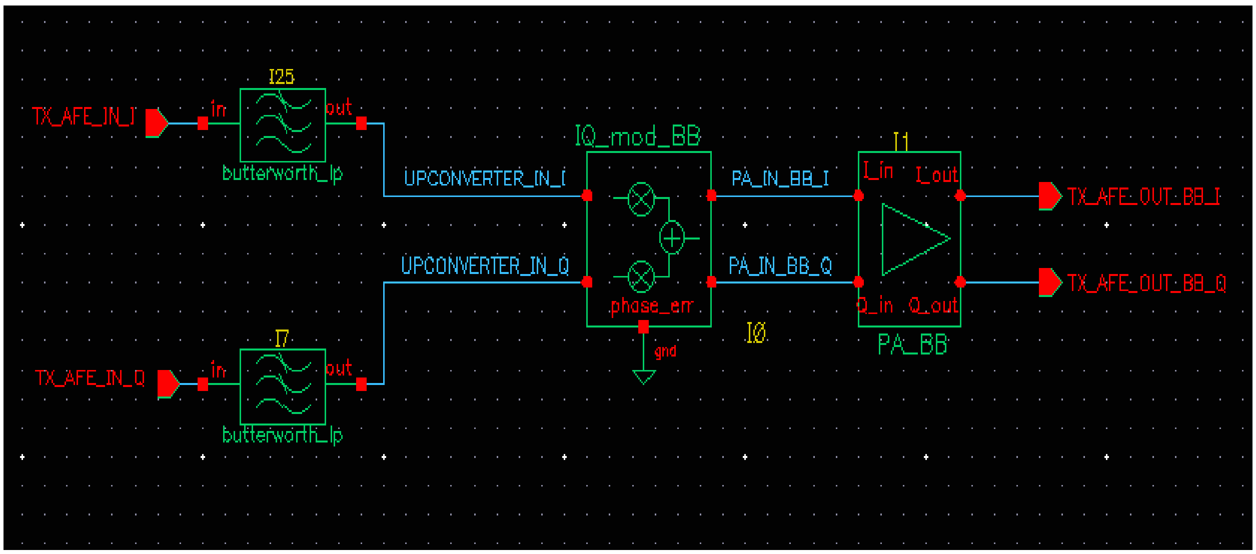
The standard compatible system-level model of the wireless LAN link comprises standard Simulink library modules. You can use this system-level model as the golden reference for the implementation of the system components.

Using cosimulation with the Virtuoso® AMS Designer simulator and The MathWorks MATLAB®/Simulink® (AMS-MATLAB/Simulink cosimulation), you can include the design and simulation of analog and mixed-signal subsystems in your system-level simulation. You can take into account the effects originating from the analog RF parts of the transmitter and receiver.

**Note:** The RF circuit in this example uses a complex base-band modeling approach. AMS-MATLAB/Simulink cosimulation supports all kinds of circuits, including base-band designs.

You can design your analog and mixed-signal subsystems using Cadence® Virtuoso® software. You can simulate these designs using the Virtuoso Spectre RF and AMS Designer circuit simulators.

The picture below shows the RF transmitter module design for this tutorial consisting of filters, an up-converting mixer, and an amplifier. For faster simulation time, we modeled the complex base-band domain of the RF components using the Verilog®-A modeling language. You can use this same approach to perform equivalent behavioral pass-band and transistor-level simulations.




When measuring the RF subsystem characteristics (such as intercept points, noise figure, and corner frequencies), you might think you can use only simple one- and two-tone sinusoidal sources as stimuli. Using AMS-MATLAB/Simulink cosimulation, you can use more realistic stimuli such as modulated signals. You can also achieve the corresponding post-processing required for system performance evaluation.

AMS-MATLAB/Simulink cosimulation combines the best of system-level simulation with analog and RF simulation. Simulink provides large libraries of DSP algorithms for generating complicated signals and for post-processing. Virtuoso® software provides an optimal design environment for analog/RF and mixed-signal subsystems. The Virtuoso AMS Designer simulator is a powerful single-kernel, mixed-signal simulator for transistor-level circuits and all common behavioral languages.

# Understanding the Different Cosimulation Flows

AMS-MATLAB/Simulink cosimulation supports different flows that support different groups of users:

| Flow | Description |
|---|---|
| ADE Flow | Run cosimulation by starting MATLAB/Simulink from the Virtuoso® Analog Design Environment (ADE)<br><br>**Note:** This flow is for users who are familiar with ADE.<br><br>See<br><br>■ Running Cosimulation from ADE Using the Fixed-Cell Coupler on page 39<br><br>■ Running Cosimulation from ADE Using the simulinkCoupler on page 55 |
| Simulink Flow | Run the cosimulation from MATLAB/Simulink (without starting ADE) using the `runSimulation` script that comes from the ADE flow<br><br>**Note:** This flow is for users who do not need to use the Virtuoso design environment or who need to debug in the Simulink environment.<br><br>See “Running Cosimulation from MATLAB/Simulink” on page 61. |
| AMS Environment Flow | Start MATLAB first, then start the AMS environment from the Virtuoso hierarchy editor (HED), and run simulations separately using each of these programs<br><br>**Note:** You can use this same flow from ADE, but this tutorial will cover only the flow from the AMS environment (HED).<br><br>See “Running Cosimulation from the AMS Designer Environment” on page 67. |

# Setting Up the AMS-MATLAB/Simulink Cosimulation Example

To set up the example files, do the following in a terminal window:

1. Make and change to a directory for the example:

```
mkdir amsTutorials
cd amsTutorials
```

2. Copy the migration example files to this directory:

```
cp -r $CDSHOME/tools/dfII/samples/tutorials/AMS/MATLABCosimulation.tar.gz .
```

3. Decompress the archive file:

```
gunzip MATLABCosimulation.tar.gz
tar xf MATLABCosimulation.tar
```

4. Change to the following directory:

```
cd MATLABCosimulation
```

5. Source the setup file:

```
source SETUP
```

The `SETUP` file sets the `MATLABPATH` and `TUT_DIR` environment variables.

This tutorial demonstrates cosimulation using the IEEE 802.11a demo that appears in "Introducing AMS-MATLAB/Simulink Cosimulation" on page 7. To set up the cosimulation, do the following:

1. Verify that the standalone simulations run successfully in MATLAB/Simulink and AMS Designer.

2. Place and configure the coupler module on the Simulink schematic.

   See "Adding the Simulink Coupler Module to the Testbench" on page 20.

3. Place and configure the corresponding coupler module on the ADE schematic.

   See "Placing the coupler_2_3_a Instance on the Schematic" on page 34.

You are ready to run cosimulation using each of the three flows outlined in "Understanding the Different Cosimulation Flows" on page 9.

## Modifying the Simulink Model

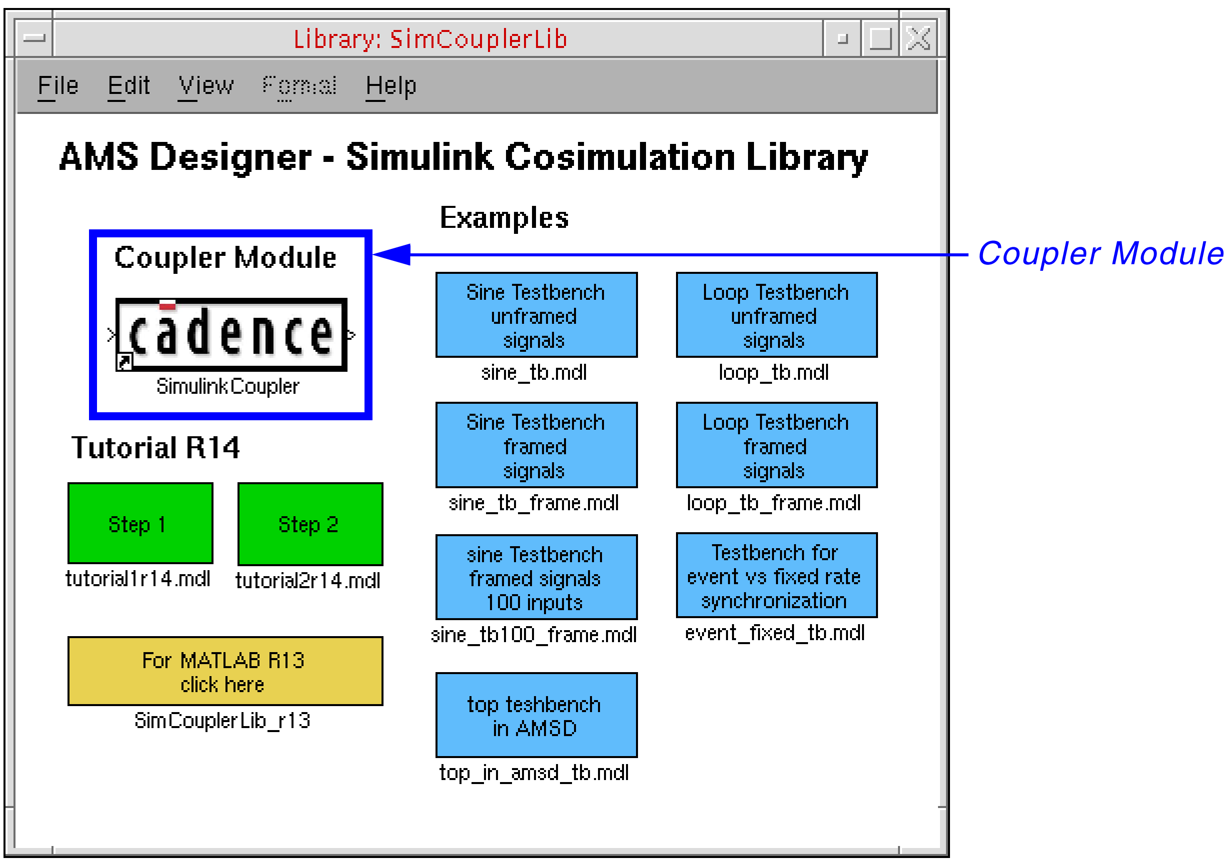
To modify the Simulink model, do the following:

1. Start MATLAB software:

```
matlab &
```

2. In the MATLAB Command Window, type the following command to open the library containing the coupler module and other examples:

```
open SimCouplerLib.mdl
```




a. To open an example, double-click its blue symbol.

b. To insert the coupler module, drag-and-drop the *Coupler Module* from the *AMS Designer – Simulink Cosimulation Library*.

*Important*

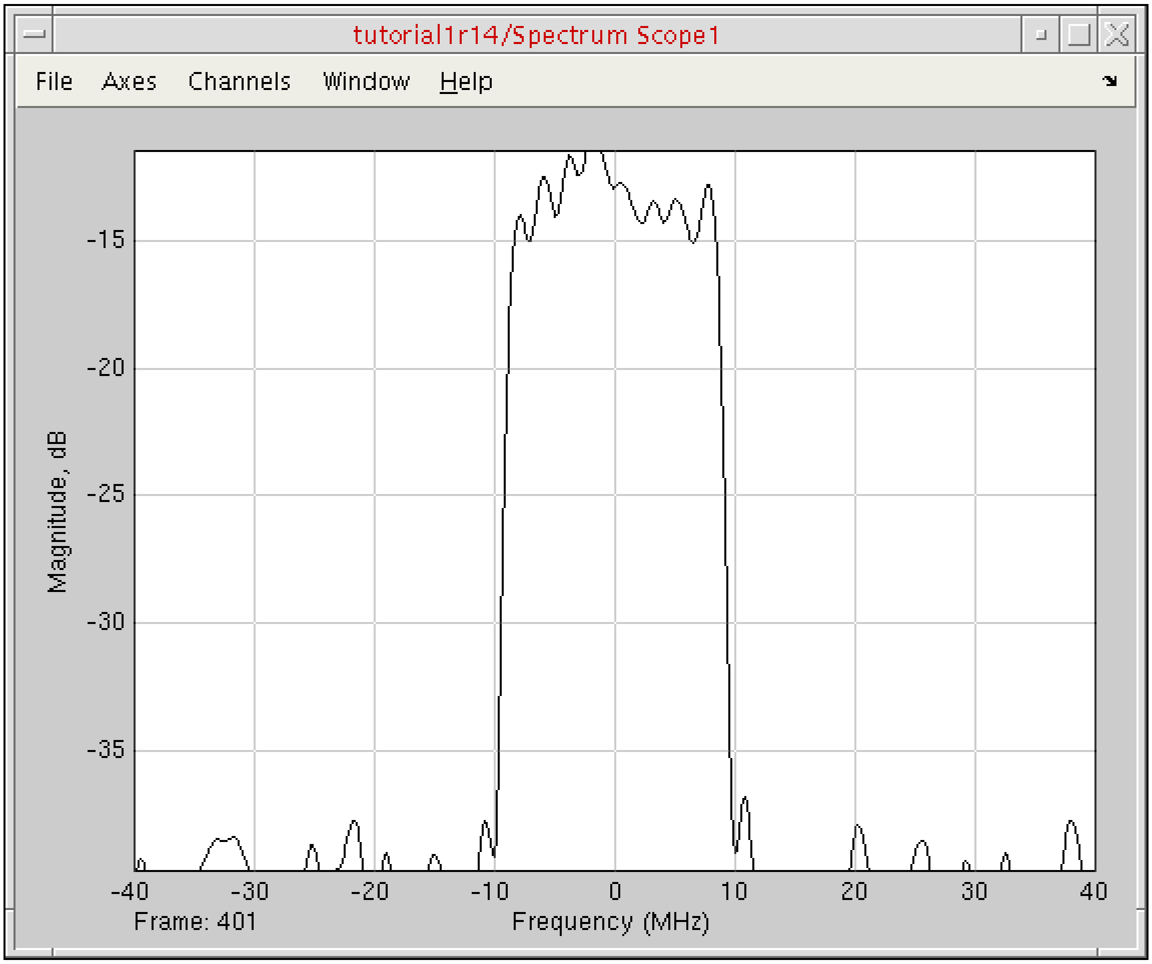
If you are using MATLAB version R13 (instead of R14), you must open the *SimCouplerLib_r13* at this point by double-clicking the yellow box. While the coupling technology is equivalent in both MATLAB releases, the MATLAB demo and the standard Simulink libraries are slightly different. R14 designs are not backward compatible with R13.

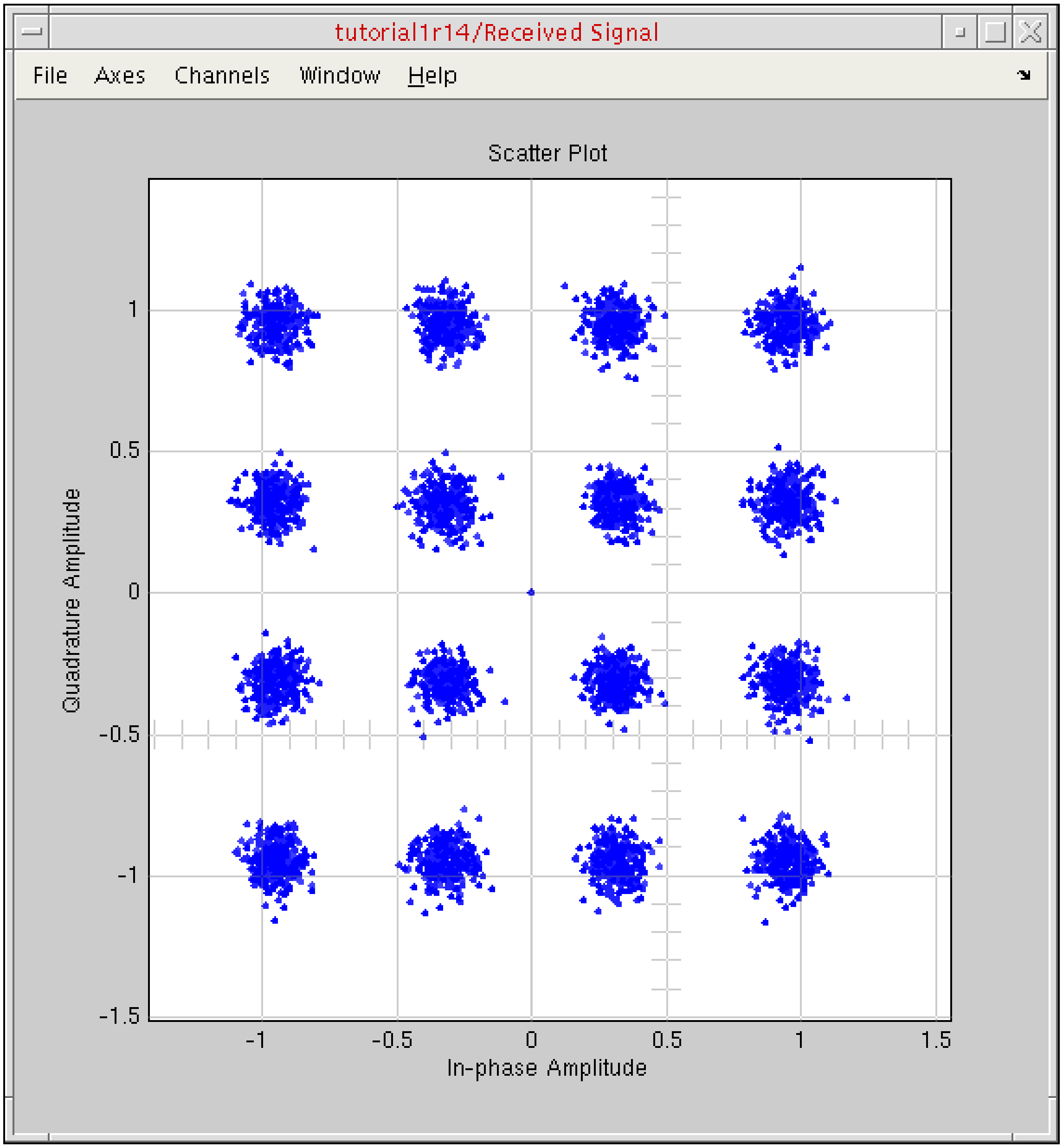
3. Double-click *Step 1* (leftmost green box) to open the end-to-end design for a wireless LAN transmission system.

   See the picture in "Introducing AMS-MATLAB/Simulink Cosimulation" on page 7.

4. Choose *Simulation – Start*.

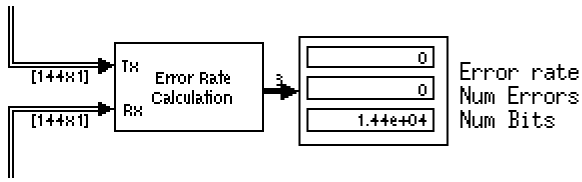
A spectrum scope and scatter diagram appear.






The minor deviations in the scatter diagram originated from the noisy channel.

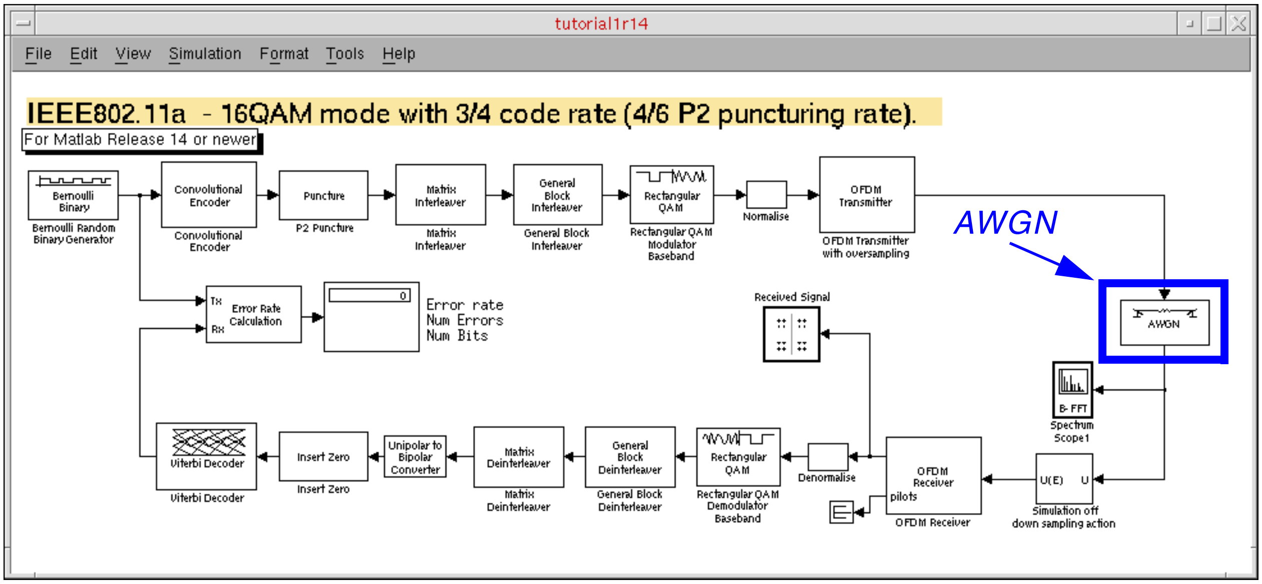
In the testbench window, the bit error rate information appears at the output of the *Error Rate Calculation* block.




**5.** Observe that the bit error rate is zero.

# Increasing the Noise for the AWGN Channel

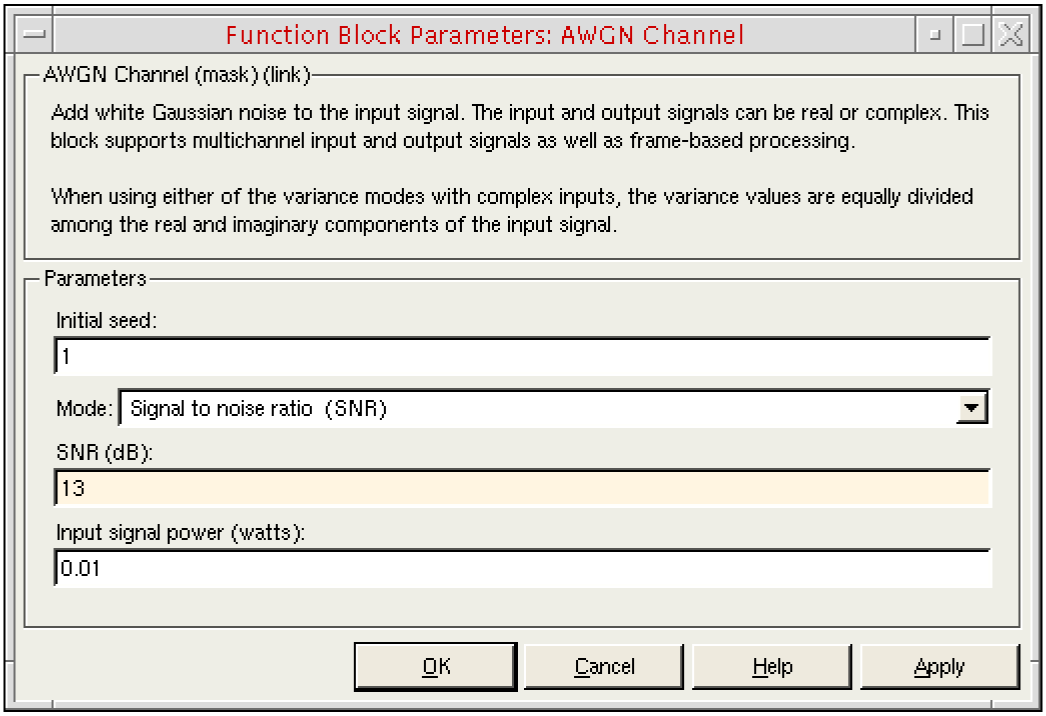
To increase the noise for the AWGN channel, do the following:




1. In the testbench window, double-click the *AWGN* block.

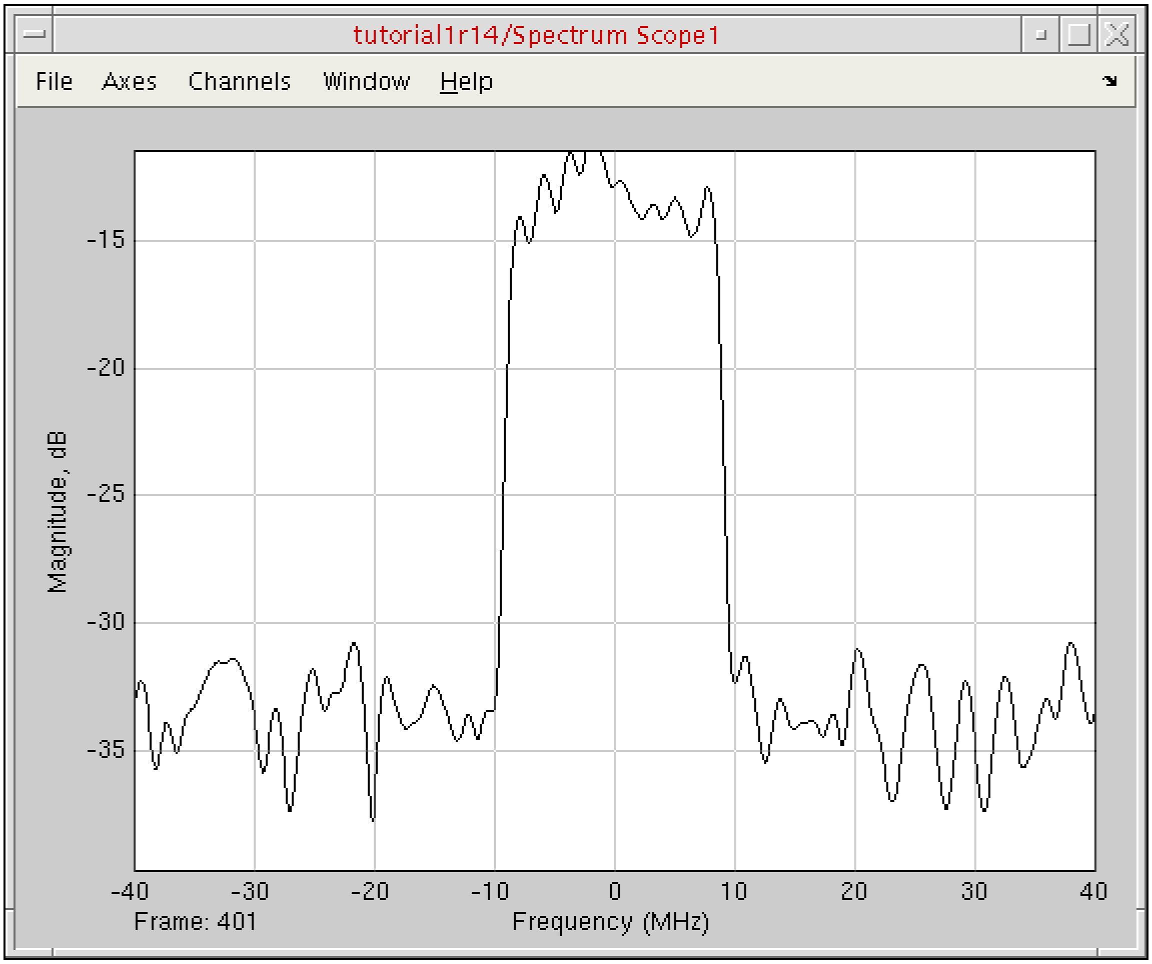
   The Function Block Parameters form appears.

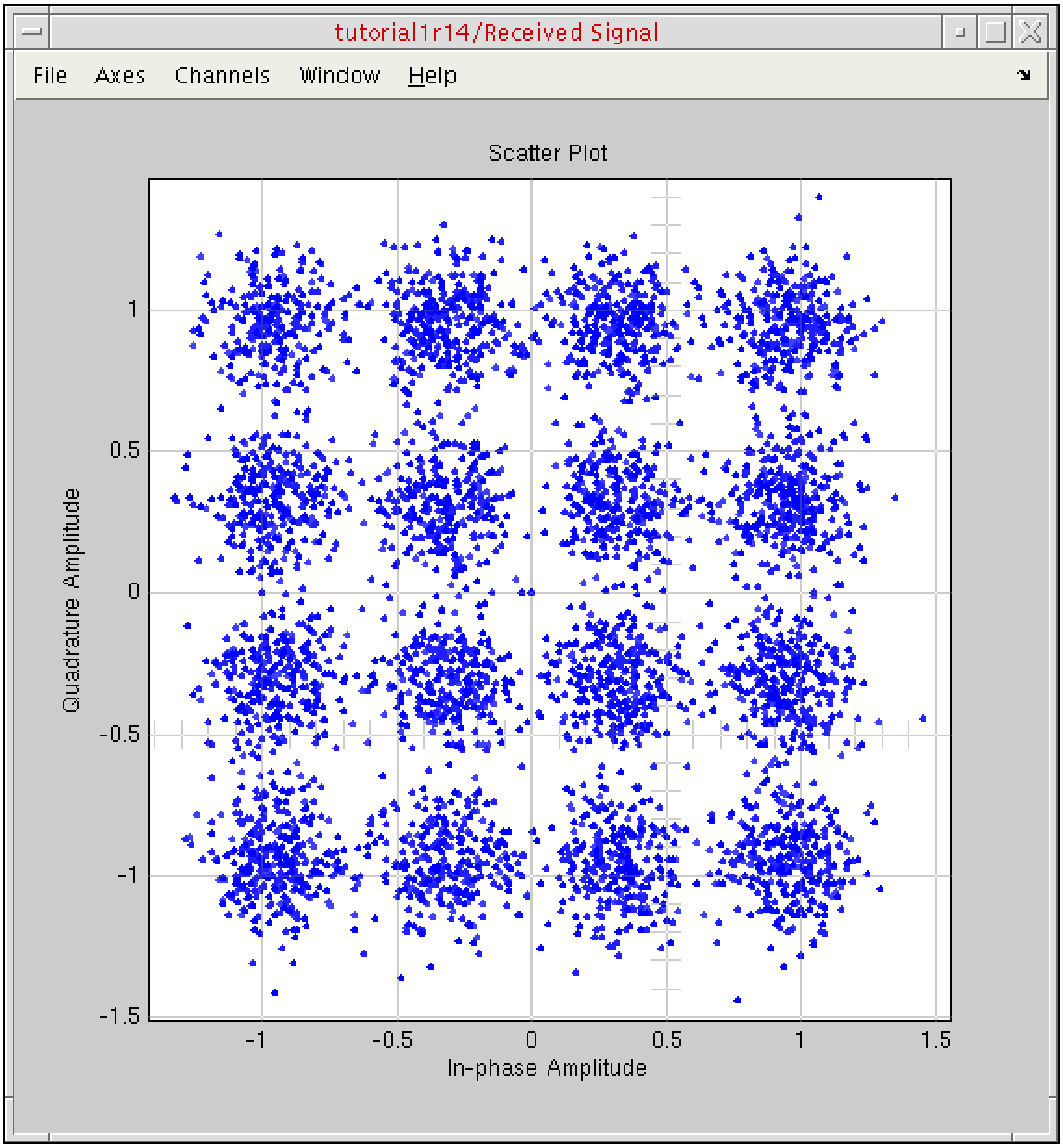
2. In the *SNR (dB)* field, change the signal-to-noise ratio from *20* to `13`.



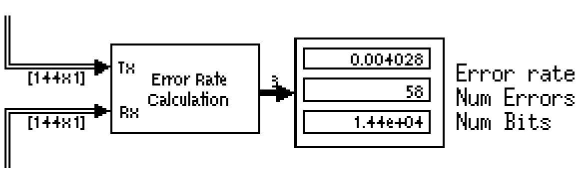

3. Click *Apply*.
4. Choose *Simulation – Start* to start the simulation again.

The final plots look like this:



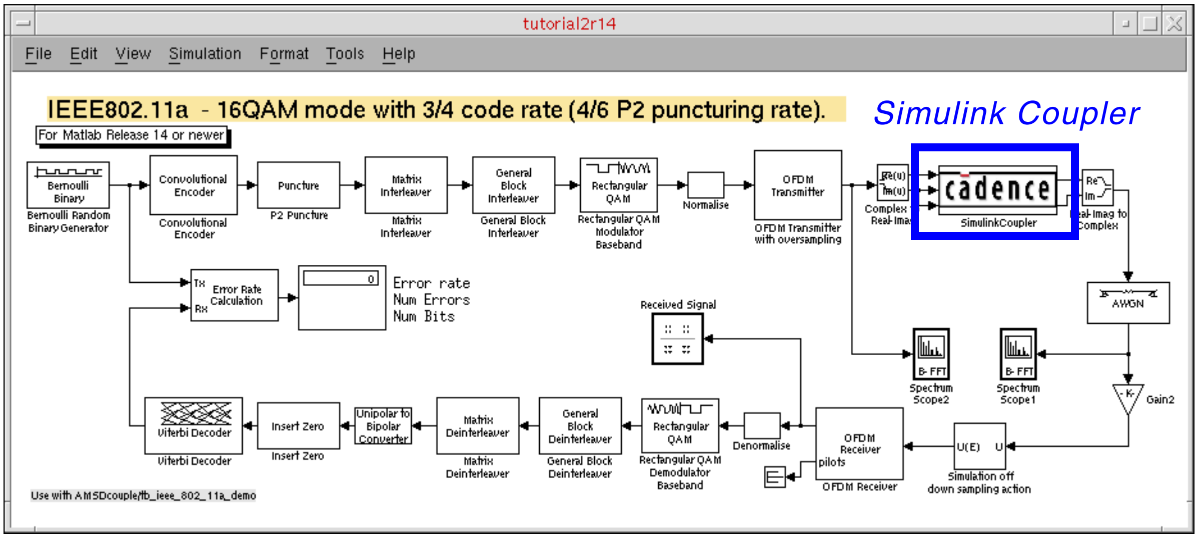



The scatter diagram shows some distortion. After some seconds, bit errors do occur.




**5.** Change the SNR back to 20.

# Adding the Simulink Coupler Module to the Testbench




To add the Simulink coupler module to the testbench, do the following:

1. Drag-and-drop the *SimulinkCoupler* block from the Library window (see “Modifying the Simulink Model” on page 11) to the testbench window (see above for placement position).

   We place the coupler block between the *OFDM Transmitter* and the *AWGN* block to include the analog transmitter RF front end in this system testbench.

2. Double-click the *SimulinkCoupler* block.

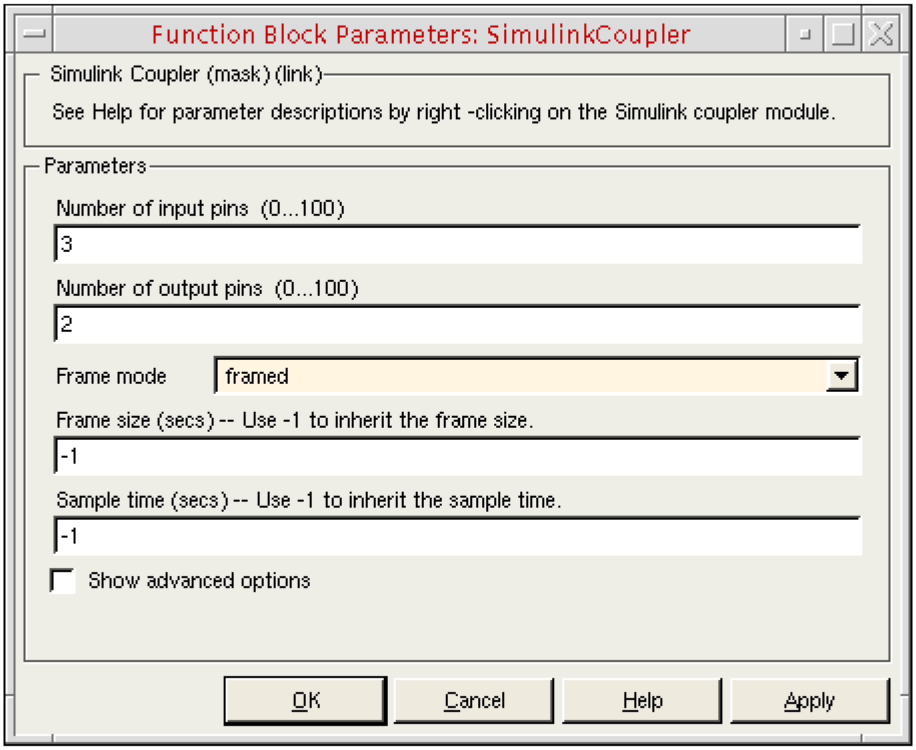
   The Function Block Parameters window appears.

3. In the *Number of input pins* field, type `3`.

   While the third input pin is not necessary in this case, it illustrates how signals flow and how we will create the coupler in the Virtuoso® design environment.

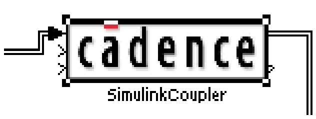
4. In the *Number of output pins* field, type `2`.

5. In the *Frame mode* field, select *framed*.




6. Click *OK*.

   The *SimulinkCoupler* block has the correct number of pins.




   You can resize the *SimulinkCoupler* block so that it fits better with the signal lines.

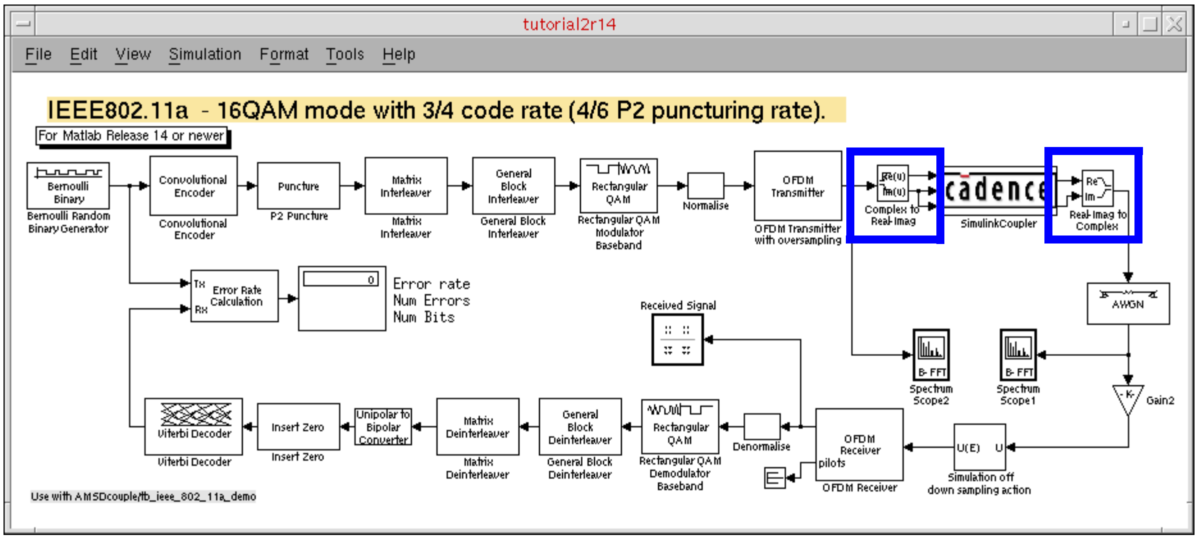
7. (Optional) To view details about the framed signals, choose *Format – Port/Signal Displays – Signal Dimensions* in the testbench window.

   Signal dimensions appear on the schematic.

   **Note:** Later sections contain descriptions of other *SimulinkCoupler* parameters.

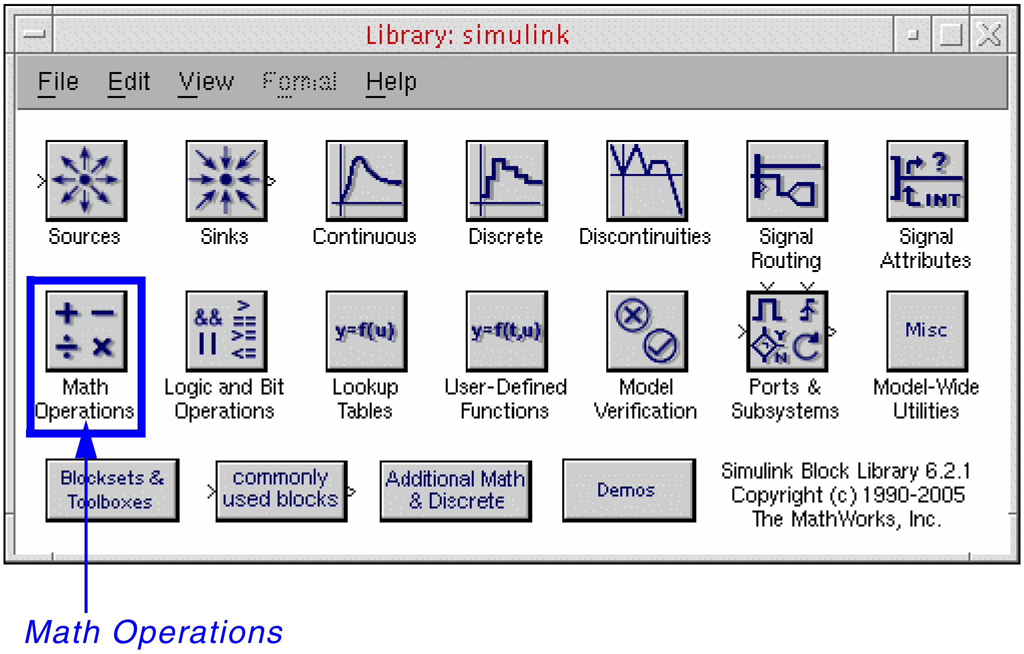
# Converting Complex Signals to Real and Imaginary Parts

The signals in this example are complex-valued signals. Before simulating with AMS Designer, you must split these complex signals into their real and imaginary parts. The Simulink library contains the converters we need for this purpose.




1. In the testbench window, choose *View – Simulink Library*.

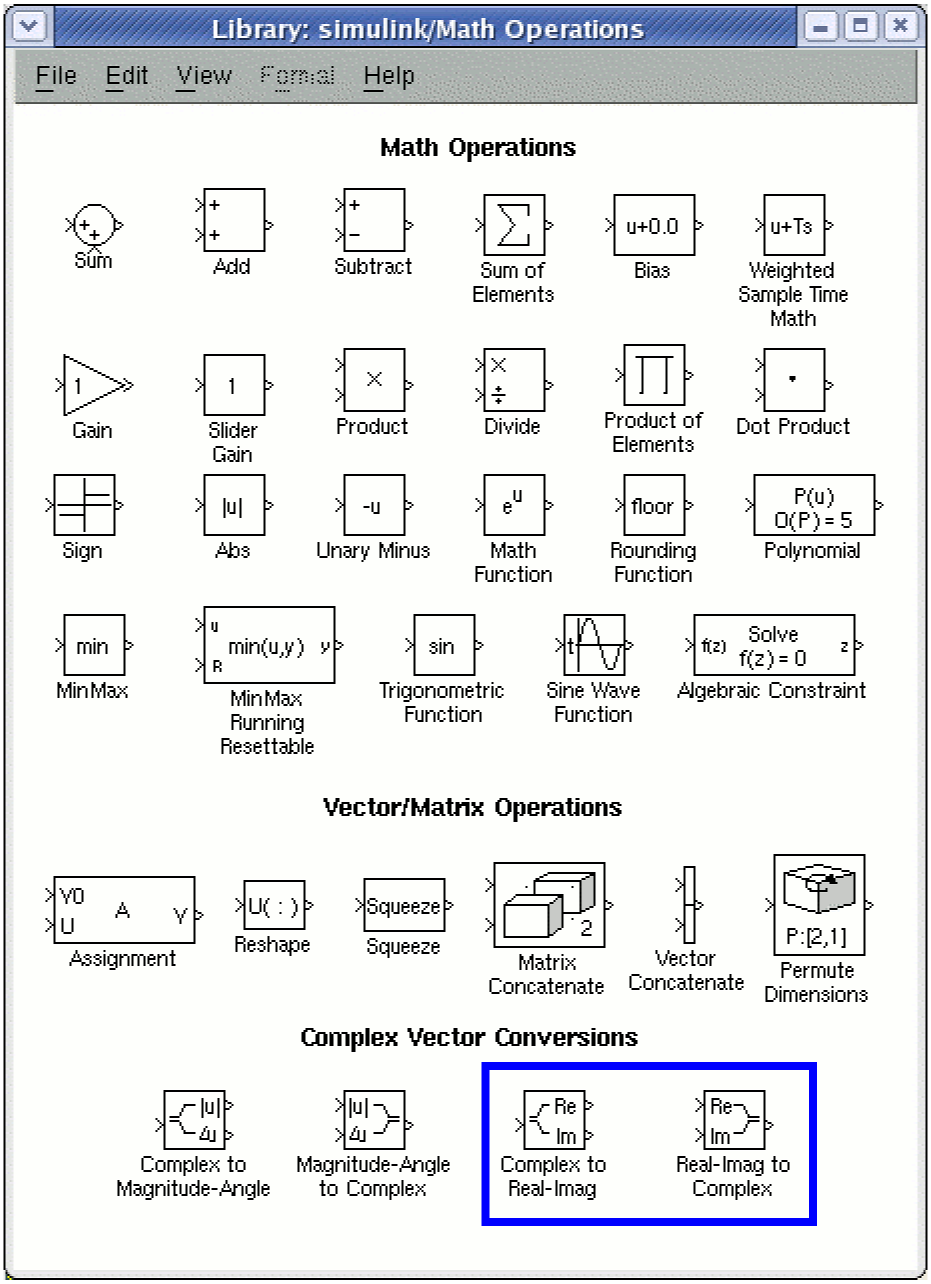
The Library window appears.




2. Double-click *Math Operations*.

   The math operator and conversion blocks appear.

The conversions we want for this tutorial appear in the bottom right corner.




3. Drag-and-drop the *Complex to Real-Imag* block from the Library window to the testbench window and place it on the left side of the *SimulinkCoupler* block.

4. Drag-and-drop the *Real-Imag to Complex* block from the Library window to the testbench window and place it on the right side of the *SimulinkCoupler* block.

5. Rewire the testbench schematic to accomodate the new blocks.

# Inserting an Ideal Gain Block in the Testbench Schematic

Because the analog/RF subsystem will eventually change the signal level, we will insert an ideal gain block at the output of the coupler block so that we can adapt the signal level properties of the digital base-band receiver.

1. In the testbench window, choose *View – Simulink Library*.

   The Library window appears.

2. Double-click *Math Operations*.

   The math operator and conversion blocks appear.

3. Drag-and-drop the *Gain* block from the Library window to the testbench window and place it near the output of the *SimulinkCoupler* block.

4. To rotate the block, right-click and choose *Format – Rotate Block*.

   See the picture in "Adding the Simulink Coupler Module to the Testbench" on page 20.

5. Double-click the *Gain* block.

   The Function Block Parameters form appears.

6. In the *Gain* field, type `0.04`.

7. Click *OK*.

8. Rewire the testbench schematic to accomodate the *Gain* block.

# Rewiring the Testbench Schematic

*Tip*

If you do not want to rewire the testbench schematic, you can double-click *Step 2* (*tutorial2r14.mdl*, the green box on the right) instead. See the picture in "Modifying the Simulink Model" on page 11.

You can rewire the design as follows:

1. Right-click the wire you want to remove and select *Cut*.

   For example, you will cut the wire between the *OFDM Transmitter* and *AWGN* blocks to fit the *SimulinkCoupler* block, and you will cut the wire between the *AWGN* and *Simulation off down sampling action* blocks to fit the *Gain* block.

2. Hover the mouse pointer over the module pin you want to connect until it changes to a cross.
3. Click-and-drag from the beginning connection point to the destination connection point and release the mouse button.

*Tip*

You can connect the blocks quickly by clicking the first block, then holding down the *Ctrl* key while clicking the second one.

Once you rewire the testbench schematic, it should look like the picture in "Adding the Simulink Coupler Module to the Testbench" on page 20. Using two *Spectrum Scope* blocks—one before and one after the *SimulinkCoupler* block—we can compare the before and after signals.

To add the second *Spectrum Scope* block before the *SimulinkCoupler* block, do the following:

1. Right-click *Spectrum Scope1* and select *Copy*.
2. Right-click near where you want to place the second one and select *Paste*.
3. To flip the block so that the port is on the left, right-click it and choose *Format – Flip Block*.
4. Hover the mouse pointer over the port until it changes to a cross.
5. Click-and-drag to connect it to the output of the *OFDM Transmitter* block.

The Simulink model is ready for cosimulation. The Simulink model automatically detects information about the frame size and sampling time.

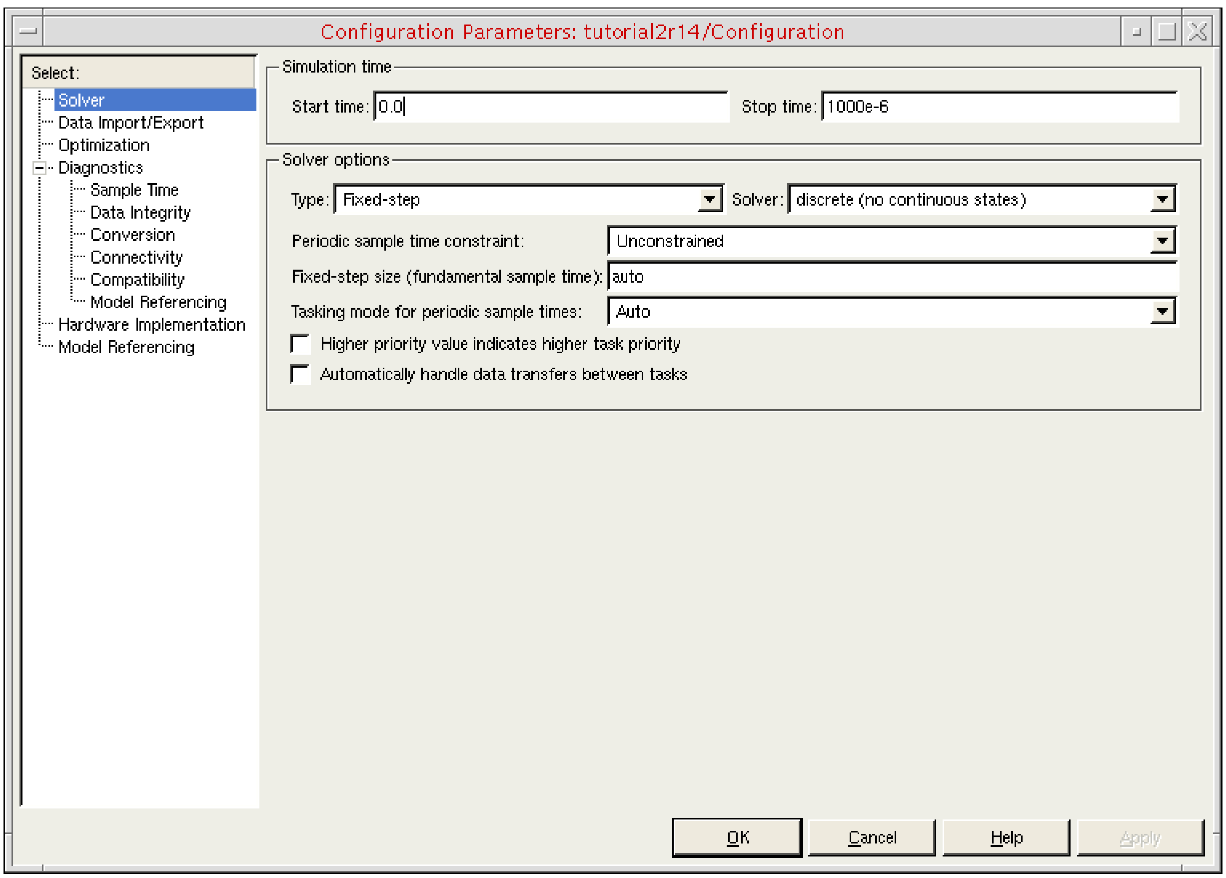
# Setting the Simulink Stop Time

To set the stop time of the Simulink simulation, do the following:

1. In the testbench window, choose *Simulation – Configuration Parameters*.

   The Configuration Parameters form appears.

2. In the *Stop time* field, type `1000e-6` (to match the `1m` stop time in the ADE setup later).




3. Click *OK*.

# Opening the Virtuoso Schematic

To open the schematic for this tutorial, do the following:

1. Start the Virtuoso® software:

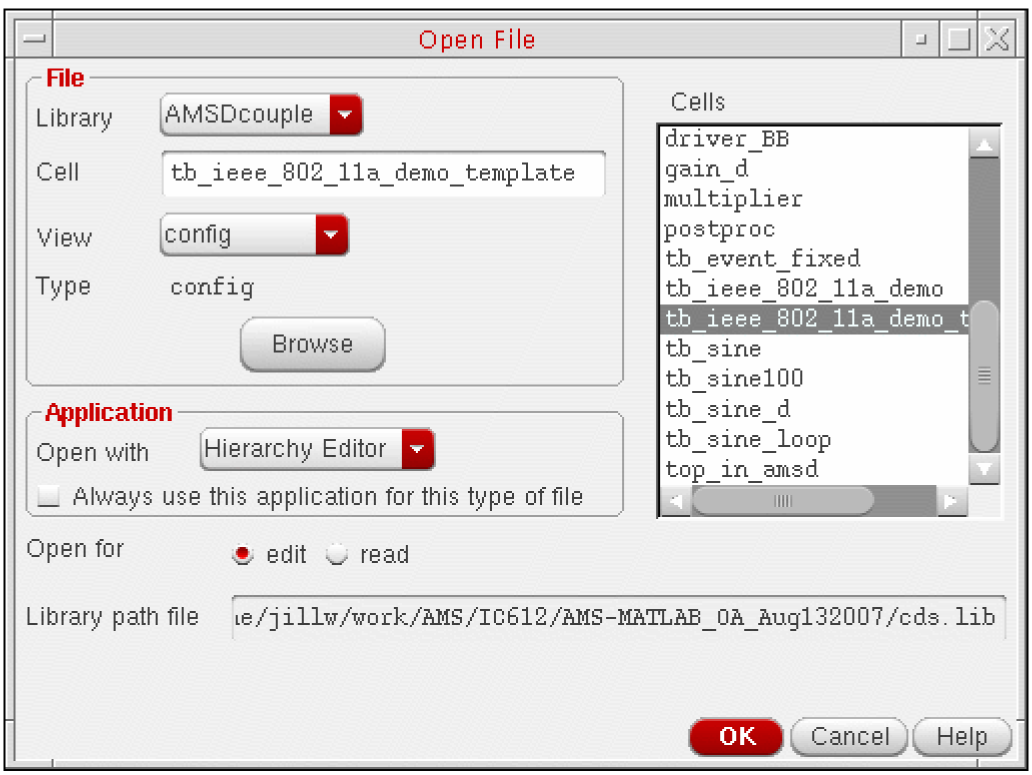
   `virtuoso &`

2. In the command interpreter window (CIW), choose *File – Open*.

   The File Open form appears.

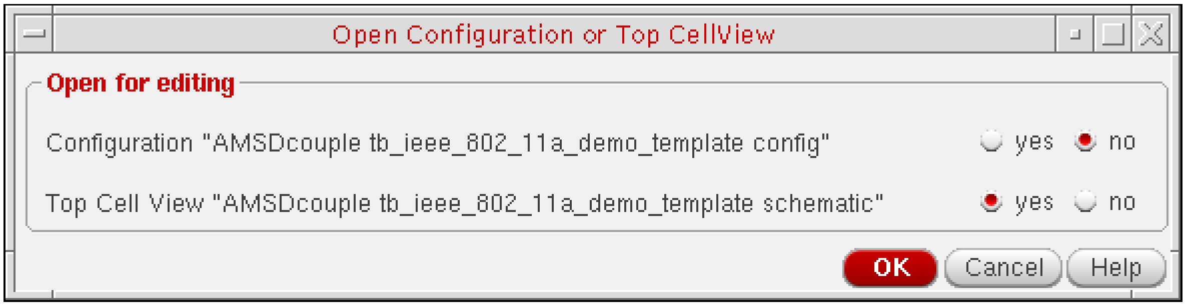
3. In the *File* group box, select the following:

| **Field** | **Selection** |
|---|---|
| *Library* | *AMSDcouple* |
| *Cell* | *tb_ieee_802_11a_demo_template* |
| *View* | *config* |



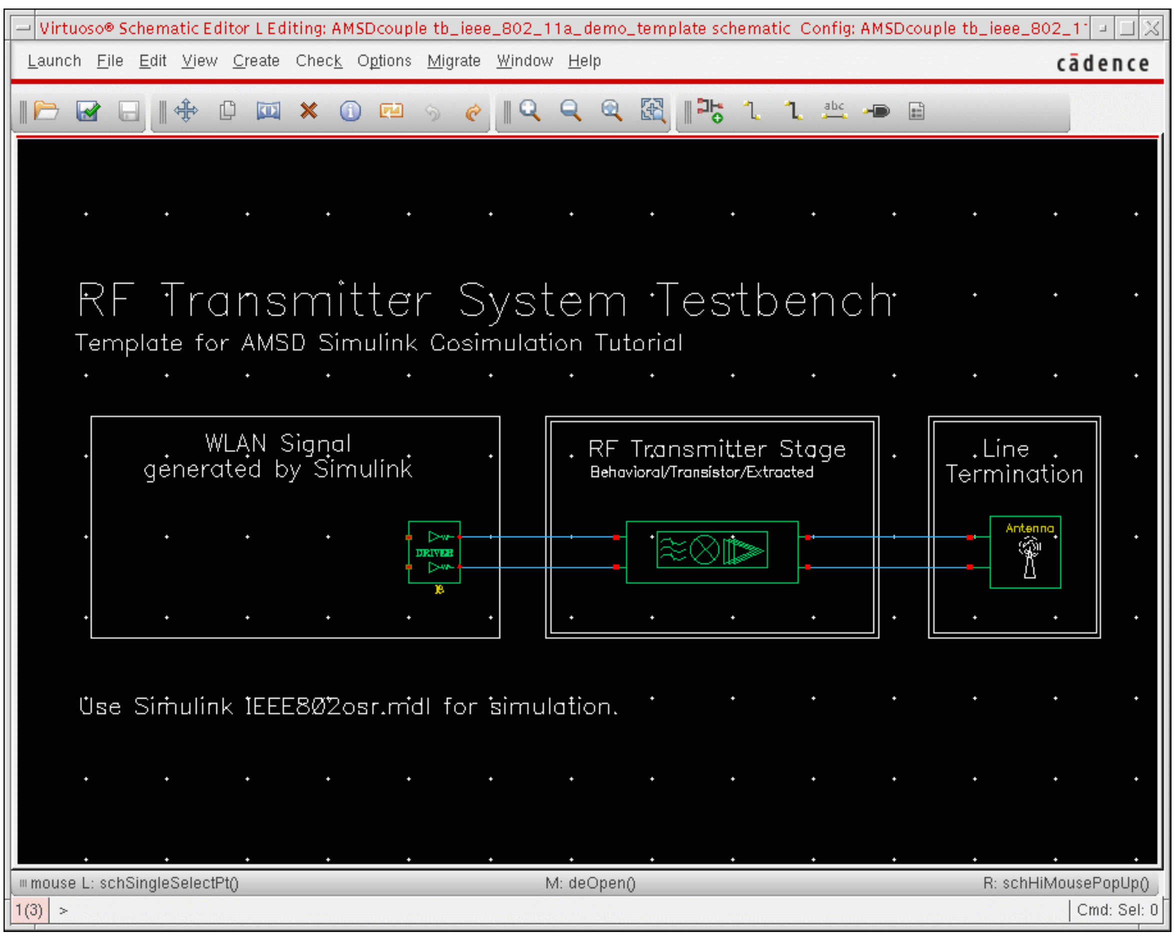

4. Click *OK*.

The Open Configuration form appears.




**5.** Click *OK*.

The *RF Transmitter System Testbench* schematic appears.




The schematic contains

- A driver module to scale the coupler output

- The RF transmitter model
- A simple line termination using resistors

The pieces that the MATLAB/Simulink design provides are

- A signal source for an 802.11a system
- The related post-processing algorithms

**Note:** The *AMSDcouple* library contains additional examples—such as *tb_sine*, *tb_sine100*, and *tb_event_fixed*. These examples have corresponding designs in MATLAB/ Simulink. You can use these examples to explore cosimulation. See also Appendix A, “Learning More about the Cosimulation Interface.”

# Creating the Coupler in the Schematic Window

You can create a coupler in the schematic window using either of two methods. The first method is the one we recommend.

- Creating a Coupler Using the Fixed-Cell Method on page 31

  Using the fixed-cell coupler method (the recommended method), you create the specific couplers that you need and save them in your library. You determine the number of coupler pins before you generate the coupler. This coupler is simple and works for all three cosimulation flows. Use this method to create a coupler instance for your design specifically.

  **Note:** We have already created a fixed coupler with two inputs and three outputs for this tutorial (`coupler_2_3_a`) and made it available in the `AMSDcoupler` library. See “Running Cosimulation from ADE Using the Fixed-Cell Coupler” on page 39 for more information.

- Creating a Coupler Using the simulinkCoupler Method on page 32

  Using the `simulinkCoupler` method, you can change the pins at any time. The simulinkCoupler is flexible and allows you to generate Verilog-AMS code automatically in the Virtuoso® Analog Design Environment (ADE). You can also modify the generated Verilog-AMS code.

  **Note:** You can find the `simulinkCoupler` in the `analogLib` library. The `simulinkCoupler` is a pcell coupler. See “Running Cosimulation from ADE Using the simulinkCoupler” on page 55 for more information.

How you perform cosimulation depends on which method you use.

## Creating a Coupler Using the Fixed-Cell Method

*Important*

The following steps show you how to create a fixed coupler with two inputs and three outputs. Because we have already created such a fixed coupler (`coupler_2_3_a`) and made it available in the `AMSDcoupler` library, so you need not perform these steps.

To create a coupler using the fixed-cell method, do the following:

**1.** In the schematic window, choose *Launch – Mixed Signal Options – AMS*.

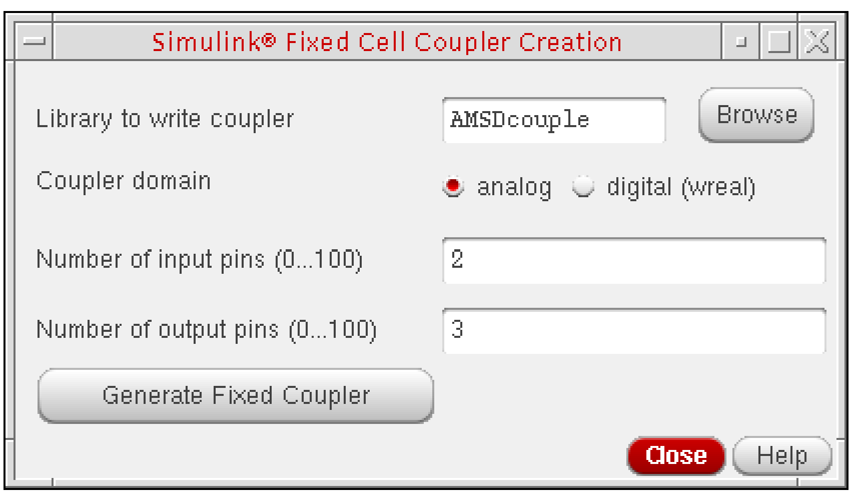
*AMS* appears on the menu banner.


**2.** Choose *AMS – Simulink® Coupler Creation*.

The Simulink® Fixed Cell Coupler Creation form appears.

**3.** In the *Number of input pins* field, type `2`.

**4.** In the *Number of output pins* field, type `3`.




**5.** Click *Generate Fixed Coupler*.

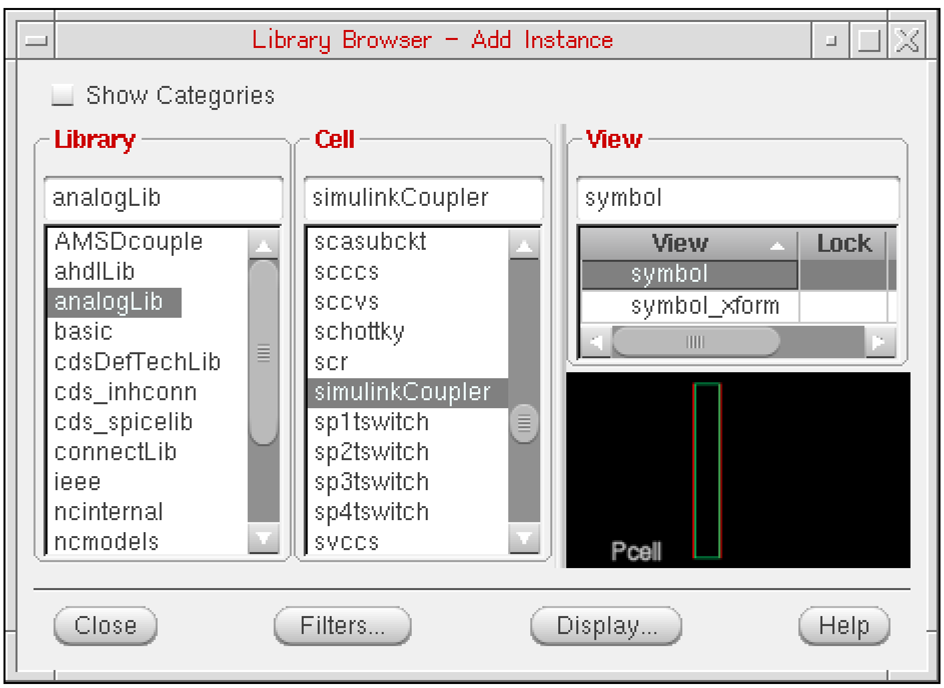
The software creates a coupler with two inputs and three outputs with the name `coupler_2_3_a`.

**6.** Click *Close*.

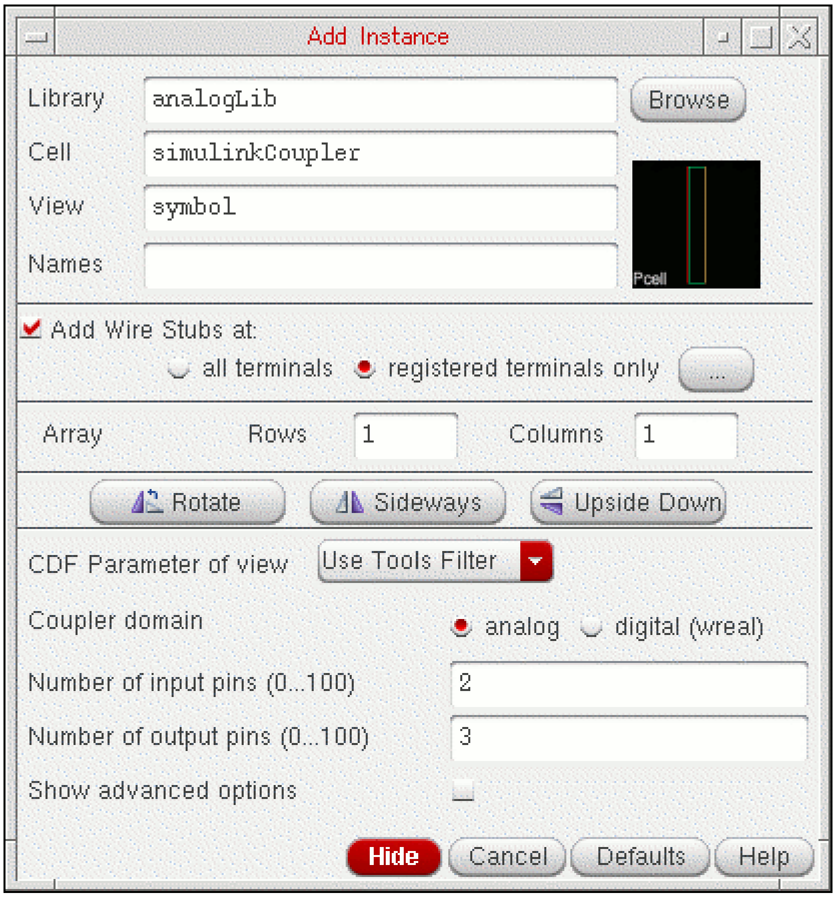
## Creating a Coupler Using the simulinkCoupler Method

The `simulinkCoupler` is a parameterized cell (pcell) that does not have a fixed number of pins: You can change the number of pins to fit different designs. To select, configure, and place a `simulinkCoupler` instance, do the following:

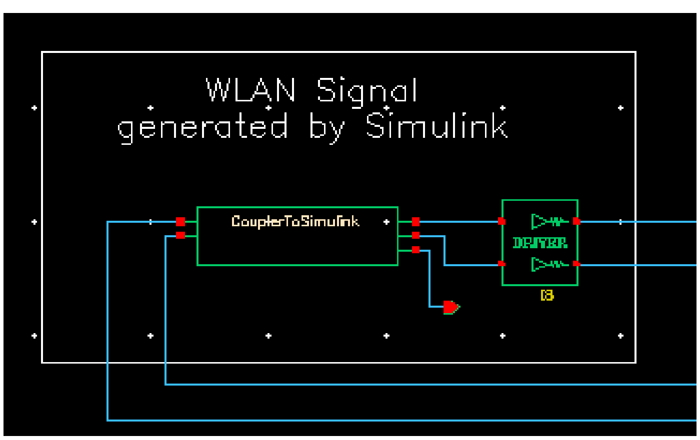
1. In the schematic window, type `i` to open the Add Instance form.
2. Click *Browse* to open the Library Browser.
3. In the *Library* column, select *analogLib*.
4. In the *Cell* column, scroll down and select *simulinkCoupler*.



5. On the Add Instance form, change the number of input pins and the number of output pins to whatever you need.




6. In the schematic window, click to place the `simulinkCoupler` instance.




7. Choose *File – Check and Save*.

# Placing the coupler_2_3_a Instance on the Schematic

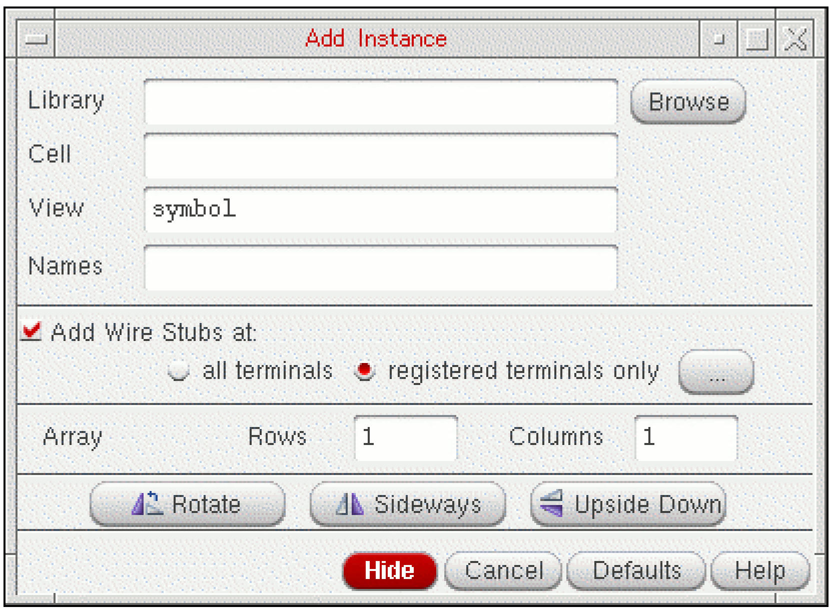
*Tip*

You can skip this section if you choose to open the finished schematic that we ship with this tutorial: `AMSDcoupler/tb_ieee_802_11a_demo/schematic`.

To place the `coupler_2_3_a` instance on the schematic, do the following:

1. In the schematic window, choose *Create – Instance* (or type `i`).

   The Add Instance form appears.

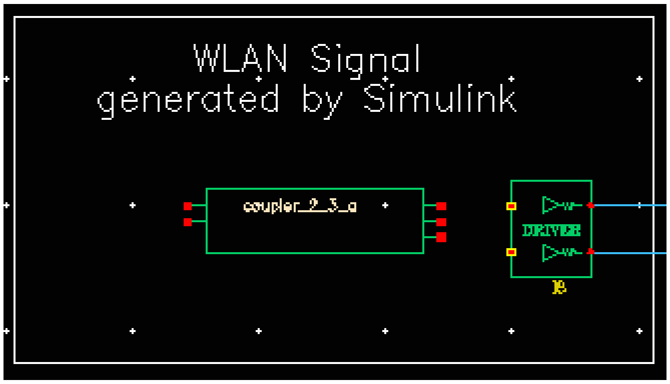



2. Click *Browse*.
3. In the Library Browser window that appears, select the following:

| | |
|---|---|
| *Library* | *AMSDcouple* |
| *Cell* | *coupler_2_3_a* |
| *View* | *symbol* |

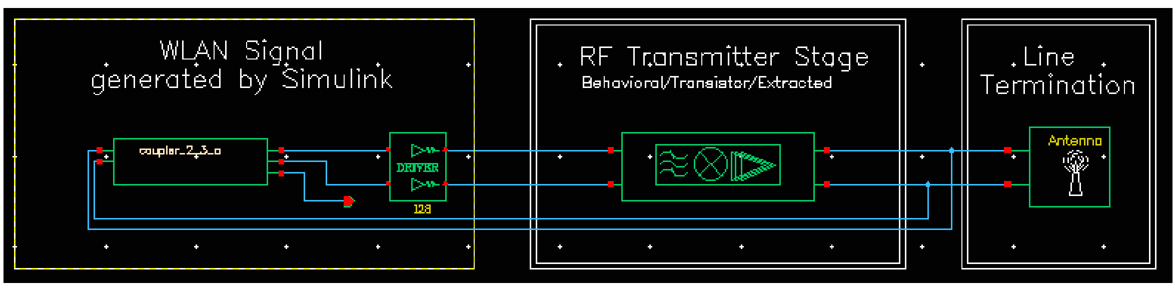
   This library/cell/view appears in the appropriate fields on the Add Instance form.

4. Move your mouse cursor over the *WLAN Signal* block on the schematic and click to place the coupler instance to the left of the driver instance.




5. Press *Esc*.

6. Wire up the instance according to the following picture (or you can open the finished schematic that we provide: `AMSDcoupler/tb_ieee_802_11a_demo/schematic`).



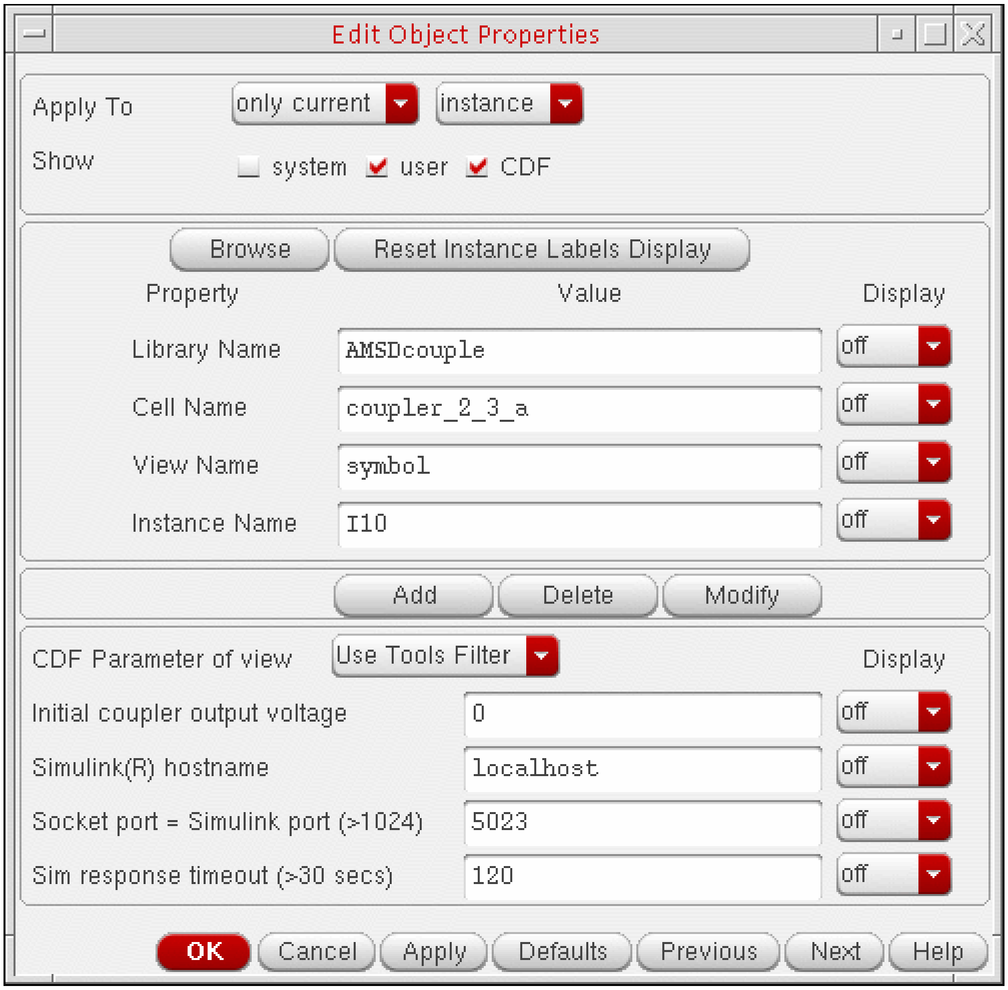

**Note:** The bottom right pin on the coupler instance connects to a pin whose name is `out` and whose direction is *output*.

7. Choose *File – Check and Save*.

You can view object properties for the coupler instance by doing the following:

1. Select the coupler instance.
2. Choose *Edit – Properties – Objects* (or type `q`).

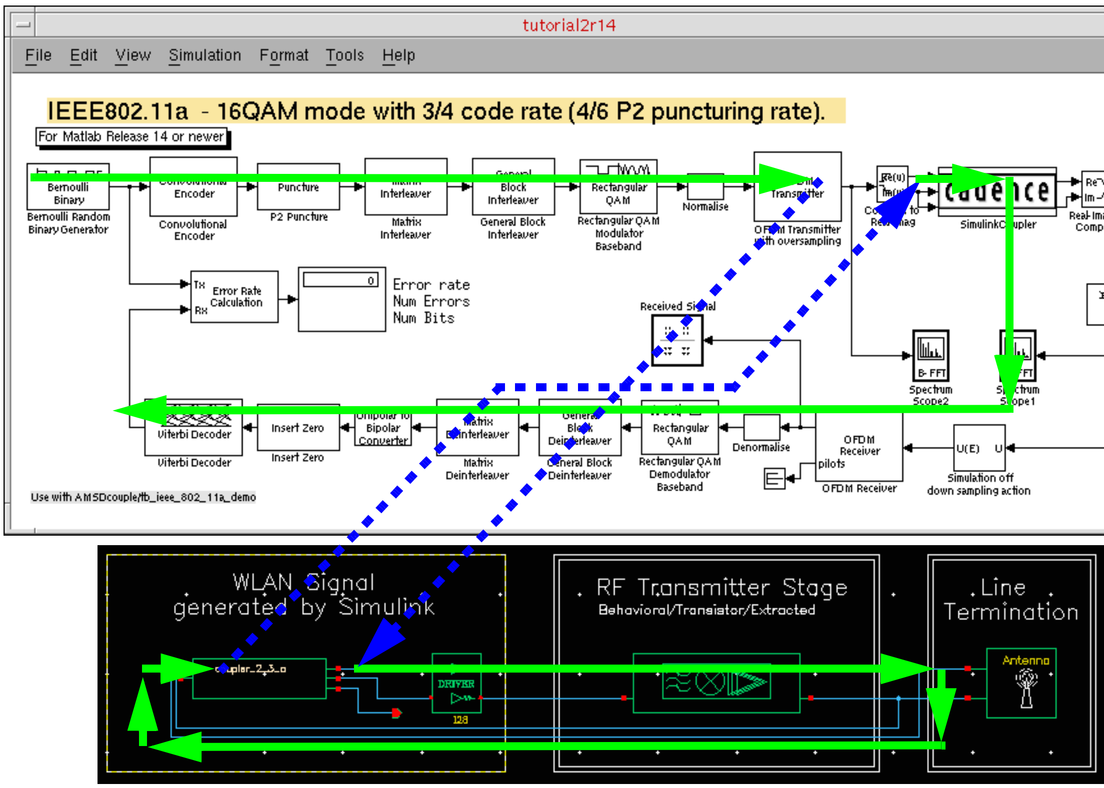
The Edit Object Properties form appears.




**3.** When you are finished viewing the object properties for the coupler, click *Cancel*.

# Viewing the Entire AMS-MATLAB/Simulink Signal Flow

The following diagram shows the entire signal flow through the MATLAB®/Simulink® and Cadence® Virtuoso® environments.




Each sink (coupler input pin) acts as a signal source (output pin) in the other environment. The signal flows into the three input pins on the Simulink schematic, out the three output pins on the Virtuoso schematic, out the two output pins on the Simulink schematic, and through the rest of the design on the Simulink schematic.

# Exiting MATLAB

To exit MATLAB, do the following:

➤ In the testbench schematic window, choose *File – Exit MATLAB*.

You are ready to run cosimulation in the Virtuoso Analog Design Environment (ADE). See "Running Cosimulation from ADE Using the Fixed-Cell Coupler" on page 39.

# 2

# Running Cosimulation from ADE Using the Fixed-Cell Coupler

**Note:** *AMS-MATLAB/Simulink cosimulation* stands for cosimulation using the Virtuoso® AMS Designer simulator and The MathWorks MATLAB®/Simulink®. *ADE* stands for the Virtuoso Analog Design Environment.

*Important*

Use IC 6.1.3, IUS 8.1 or later, and MATLAB704R14 or MATLAB R2007b or MATLAB R2008a for this tutorial. The estimated time to complete this tutorial is about one hour.

For information about the fixed-cell coupler, see "Creating the Coupler in the Schematic Window" on page 30.

To run cosimulation from ADE using the fixed-cell coupler, do the following:

1. If you have not already done so, open the RF Transmitter System Testbench config.

   Shortcut: You can open the finished config/schematic that we provide with this tutorial: `AMSDcouple/tb_ieee_802_11a_demo`.

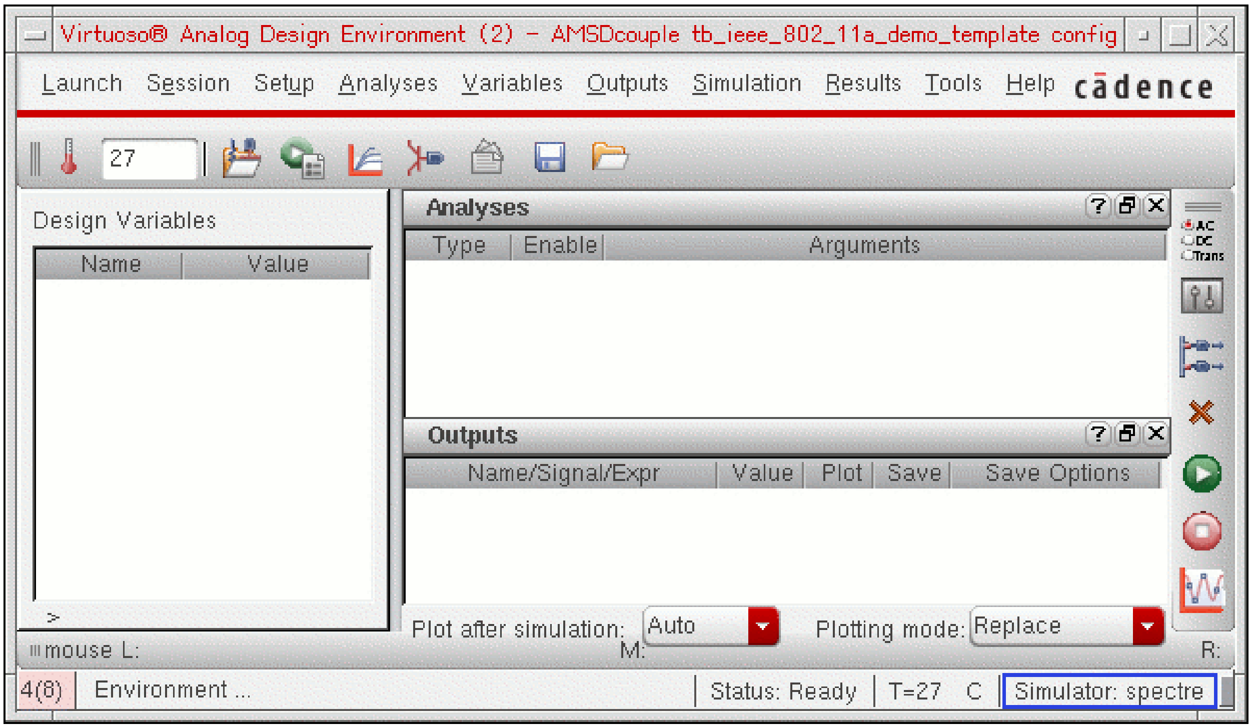
2. Open the Virtuoso Analog Design Environment (ADE) and set up the analysis.
3. Choose irun for simulation.
4. View AMS simulator options.
5. Specify values for the design variables.
6. Set up the MATLAB/Simulink simulation to start before the AMS Designer simulation.
7. Run cosimulation from ADE.
8. Rerun the cosimulation after changing a design variable value.
9. Run cosimulation by starting two simulations separately.

# Opening ADE and Setting Up the Analysis

To open ADE and set up the analysis, do the following:

1. In the schematic window, choose *Launch – ADE L*.

   *Simulator: ams(Spectre)* appears on the status bar.

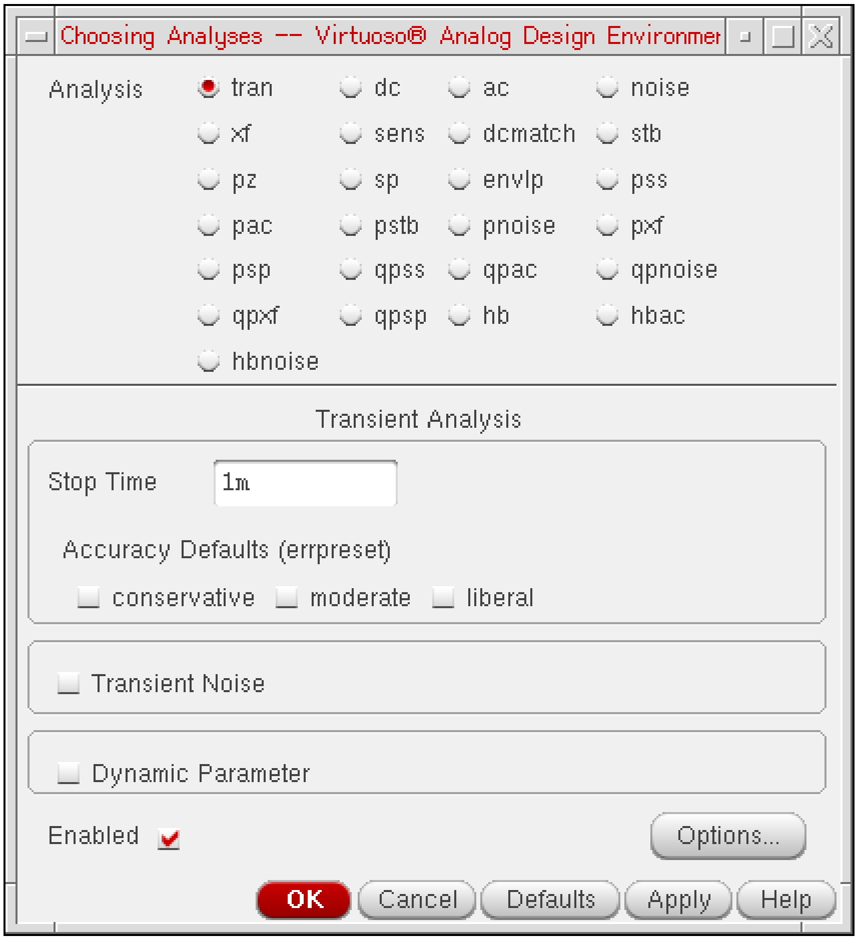



2. Choose *Analyses – Choose*.

   The Choosing Analyses form appears.

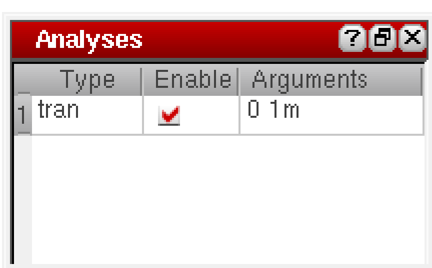
3. In the *Stop Time* field, type `1m`.

**4.** Turn on the *Enabled* check box.




**5.** Click *OK*.

The analysis setup appears in the *Analyses* section of the ADE window.



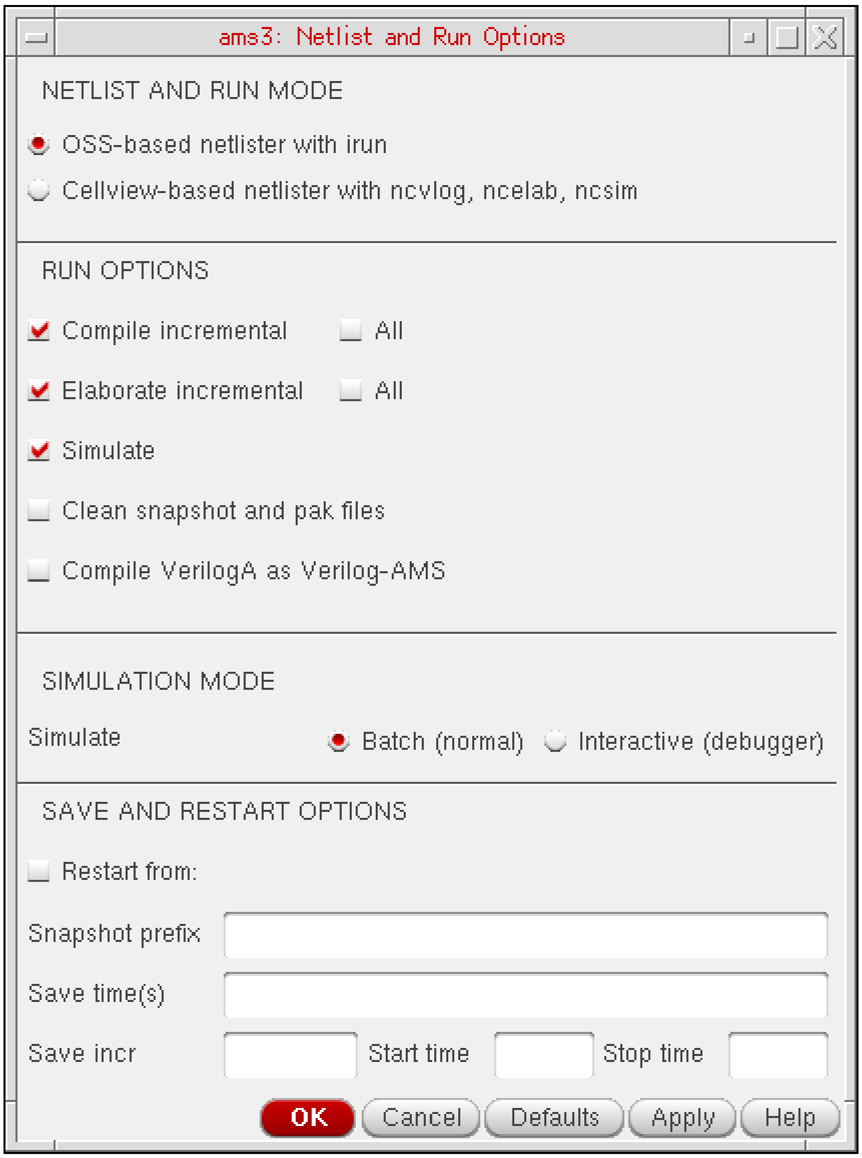
## Choosing irun for Simulation

To choose `irun` for simulation, do the following:

1. In ADE, choose *Simulation – Netlist and Run Options*.

   The Netlist and Run Options form appears.

2. In the *NETLIST AND RUN MODE* section at the top of the form, select *OSS-based netlister with irun*.




3. Click *OK*.

# Viewing AMS Simulator Options

To view AMS simulator options, do the following:

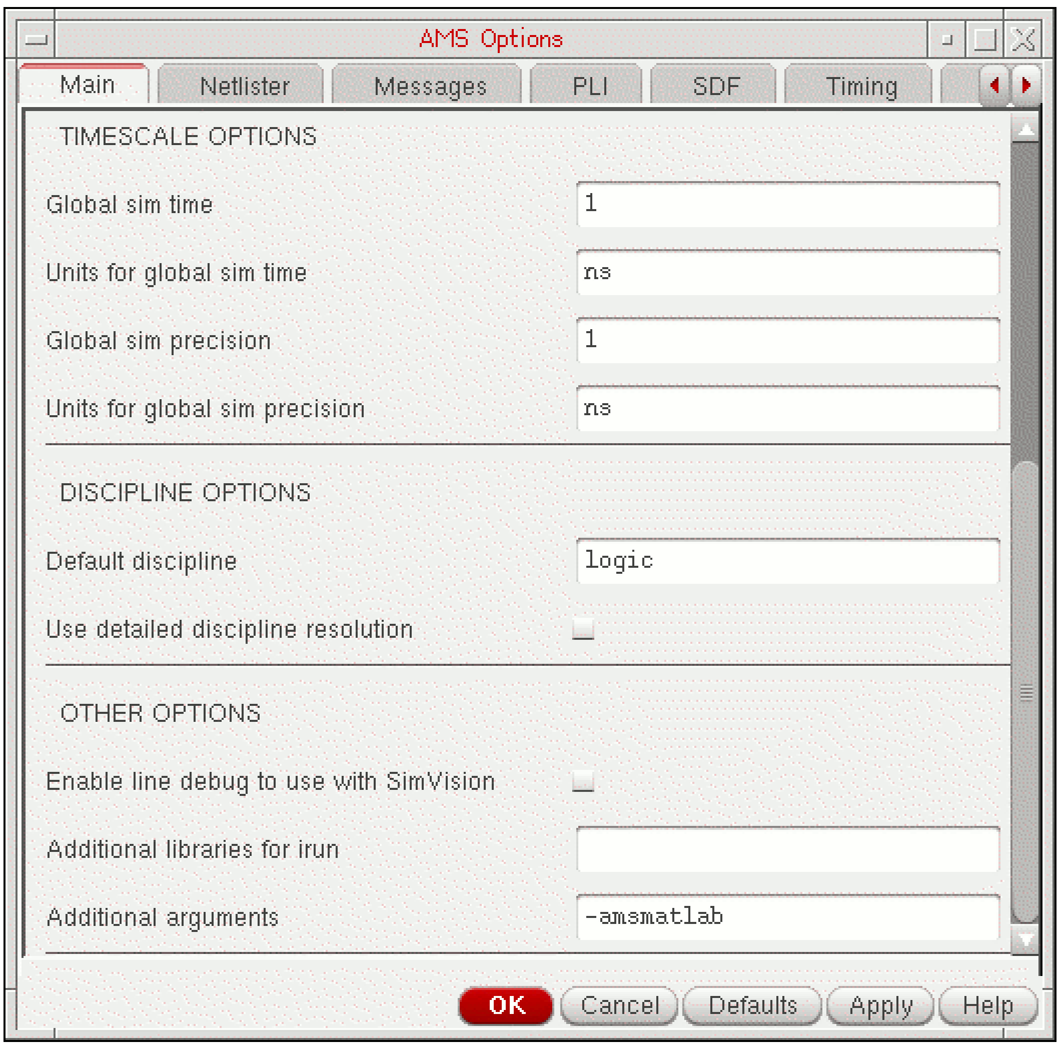
1. In ADE, choose *Simulation – Options – AMS Simulator*.

   The AMS Options form appears.

2. To link dynamically with the VPI code required for AMS-MATLAB cosimulation, do the folloing in this form:

   a. On the *Main* tab, scroll down to the *OTHER OPTIONS* section.

   b. In the *Additional arguments* field, type `-amsmatlab`.




3. (Optional) You can scroll through the tabs of this form to view other AMS options.

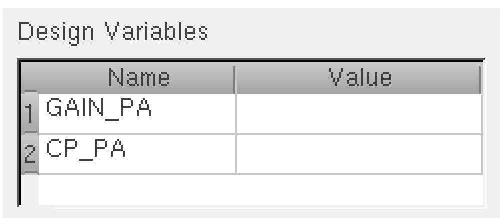
4. Click *OK*.

# Setting Up Design Variables

To set up design variables, do the following:

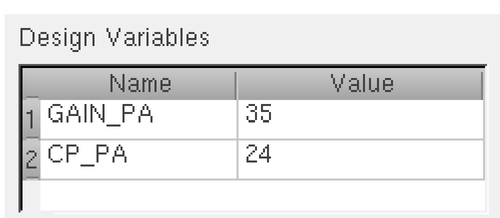
1. Copy the variables from the schematic by choosing *Variables – Copy from Cellview* in the ADE window.

   Design variables from the schematic appear in the *Design Variables* section of the ADE window.




   *GAIN_PA* controls the gain of the RF power amplifier. *CP_PA* controls the compression point of the RF power amplifier. The compression point is a measurement of the amplifier's linearity/nonlinearity: The smaller the number, the larger the amplifier's nonlinearity.

2. In the *Value* column for *GAIN_PA*, click and type `35`.
3. In the *Value* column for *CP_PA*, click and type `24`.



# Starting MATLAB before AMS Starts

To set up MATLAB to start before AMS starts, do the following:

1. In ADE, choose *Setup – MATLAB/Simulink – Start*.

   The Setup MATLAB form appears. By default *Start MATLAB* is *no*.

2. For *Start MATLAB*, select *before AMS starts*.

   This setting establishes a connection between the MATLAB/Simulink simulator and the AMS Designer simulator. The other fields on the form become active.

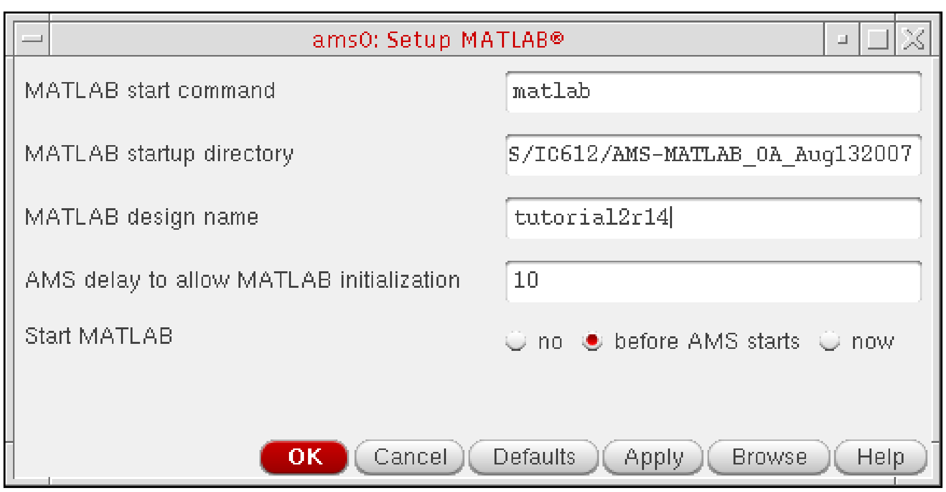
   The default MATLAB start command is `matlab`.

   The default MATLAB start-up directory is your current directory (where you started your Cadence software).

   The default delay to allow MATLAB initialization is 10 seconds. This delay allows MATLAB/Simulink to be ready and running when the AMS simulation starts. For successful cosimulation, you must coordinate start-up times such that MATLAB/Simulink is ready and running before AMS.

   **Note:** You would set *Start MATLAB* to *now* to use the flow that runs the AMS Designer simulator and environment separate from the MATLAB/Simulink simulation. See "Running Cosimulation by Starting the Two Simulations Separately" on page 52 for more information.

3. In the *MATLAB design name* field, type `tutorial2r14`.




4. Click *OK*.

   You are ready to run cosimulation from ADE.

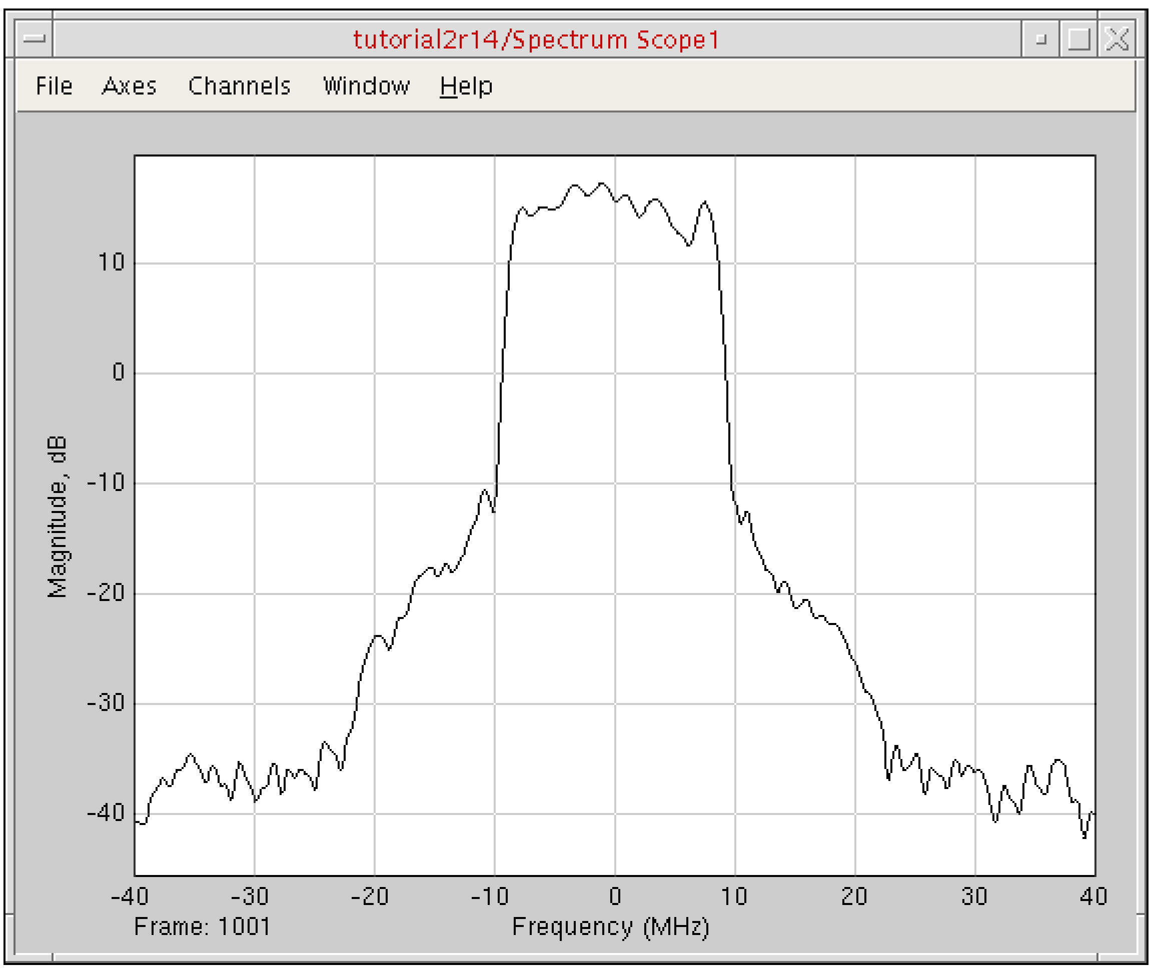
# Running Cosimulation from ADE

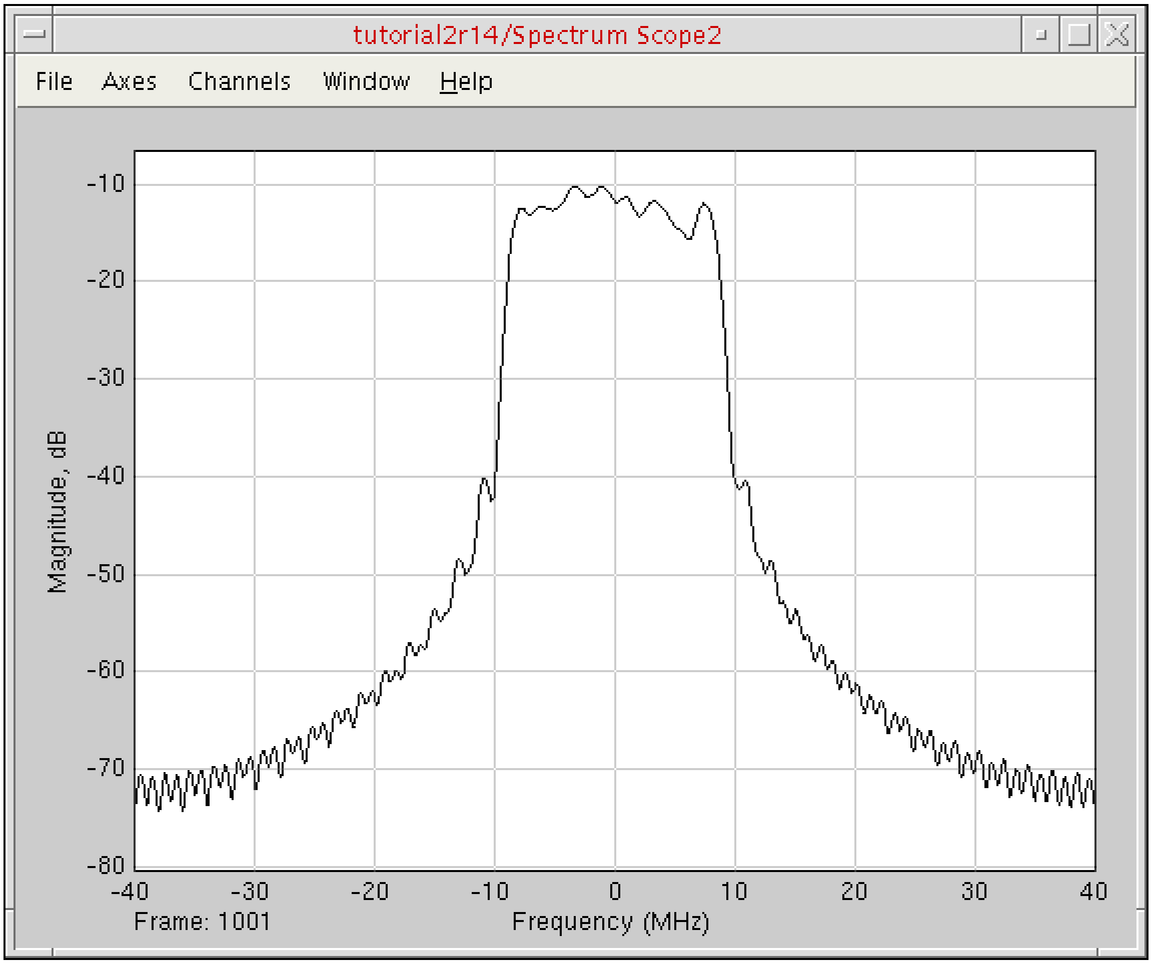
To start cosimulation from ADE, do the following:

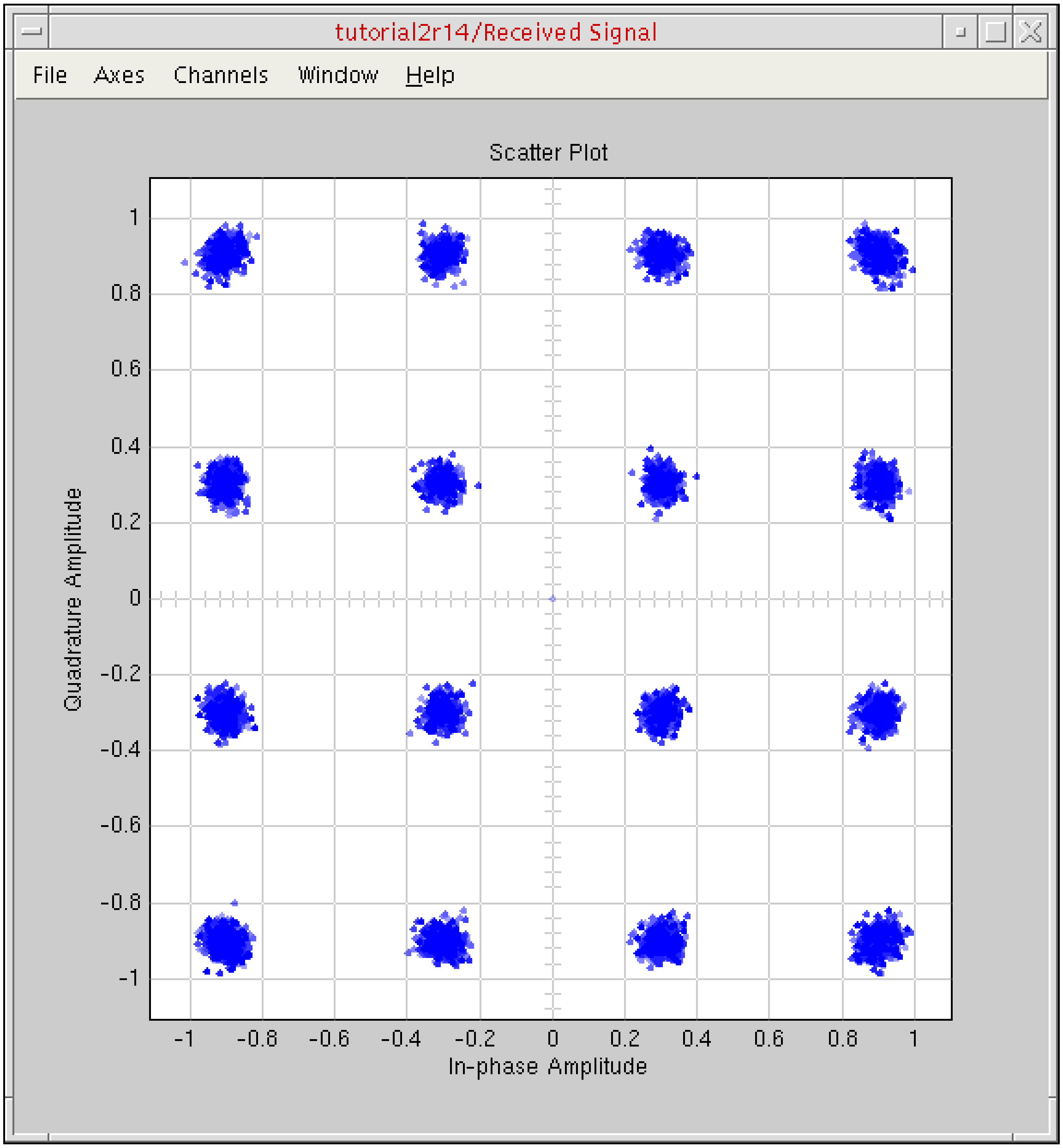
1. Choose *Simulation – Netlist and Run* (or click the green Netlist and Run button).

   The testbench schematic appears in Simulink. The MATLAB Command Window does not appear. The AMS Designer simulator starts running. The cosimulation proceeds. The `simulation.log` and `matlab_ade.log` files appear in separate windows. Simulation data appears in the Simulink *Spectrum Scope* and *Received Signal* windows.

   The input spectrum and output spectrum are different owing to of the non-ideal RF transmitter chain.




tutorial2r14/Spectrum Scope2
File Axes Channels Window Help
Magnitude, dB
-10
-20
-30
-40
-50
-60
-70
-80
-40 -30 -20 -10 0 10 20 30 40
Frame: 1001
Frequency (MHz)


tutorial2r14/Received Signal
File Axes Channels Window Help
Scatter Plot
Quadrature Amplitude
In-phase Amplitude
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

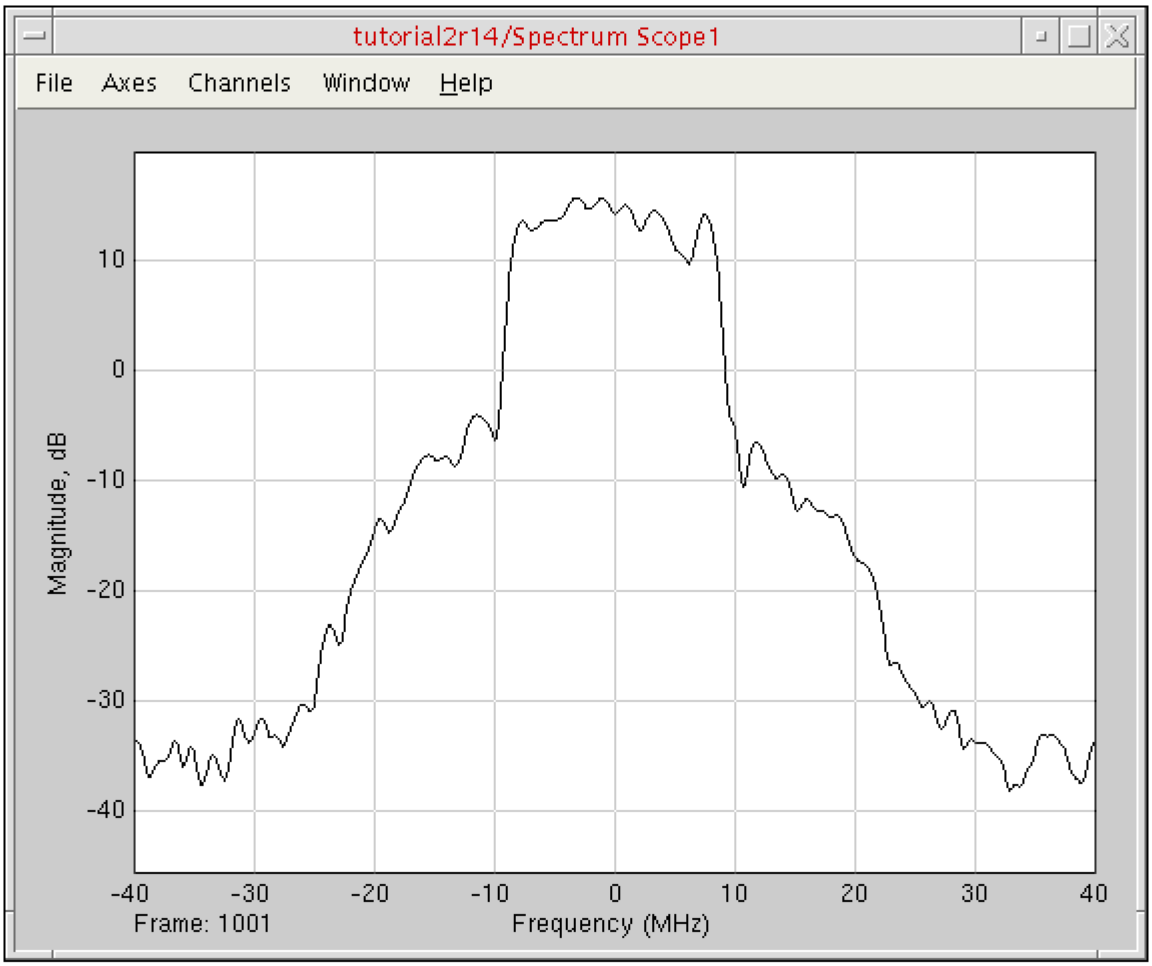
# Changing a Value and Rerunning Cosimulation

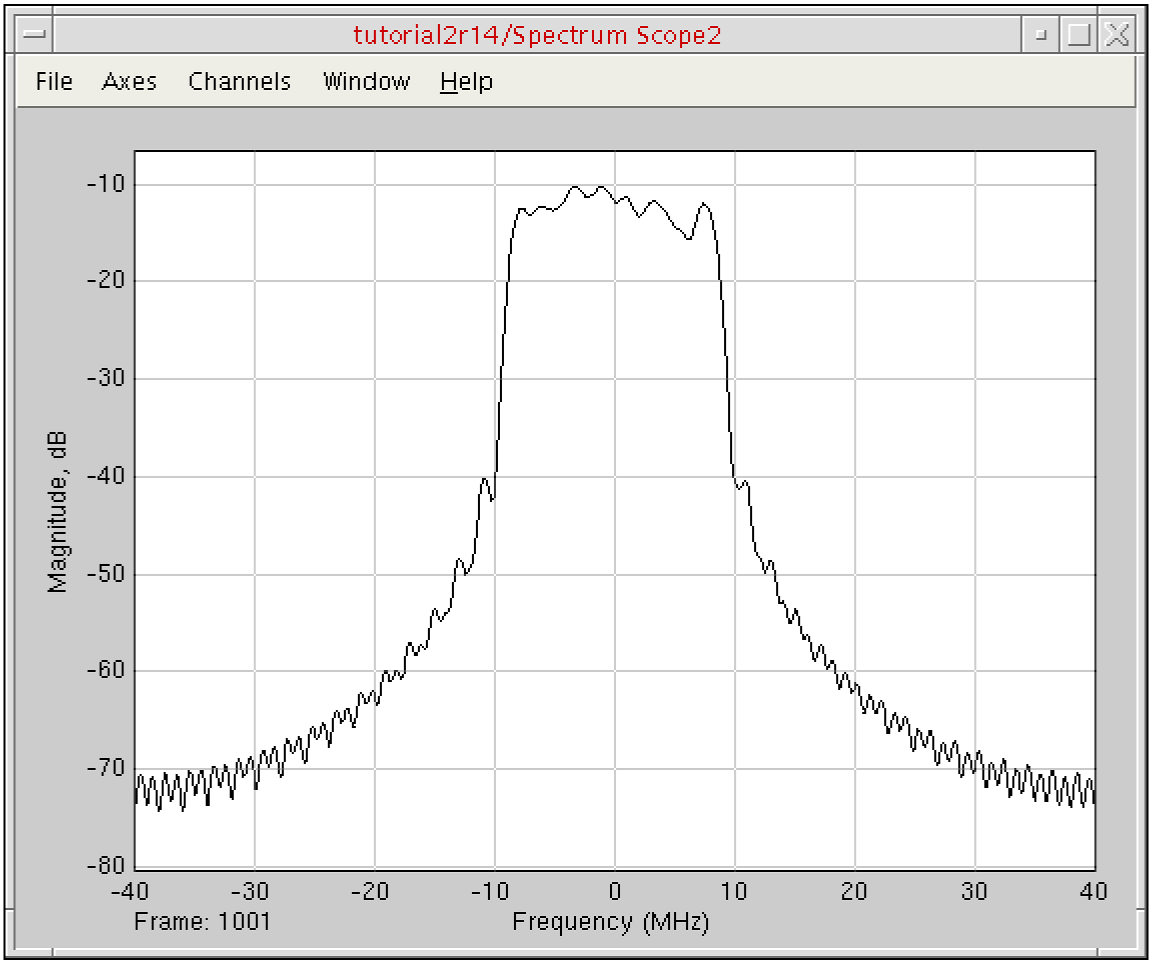
Once you are set up for cosimulation, changing values and rerunning the cosimulation is simple. To change a design variable value and rerun the cosimulation, do the following:

1. In the ADE window, double-click the value for *CP_PA* and change it to `18` (from 24).
2. Click the run button (or choose *Simulation – Netlist and Run*).

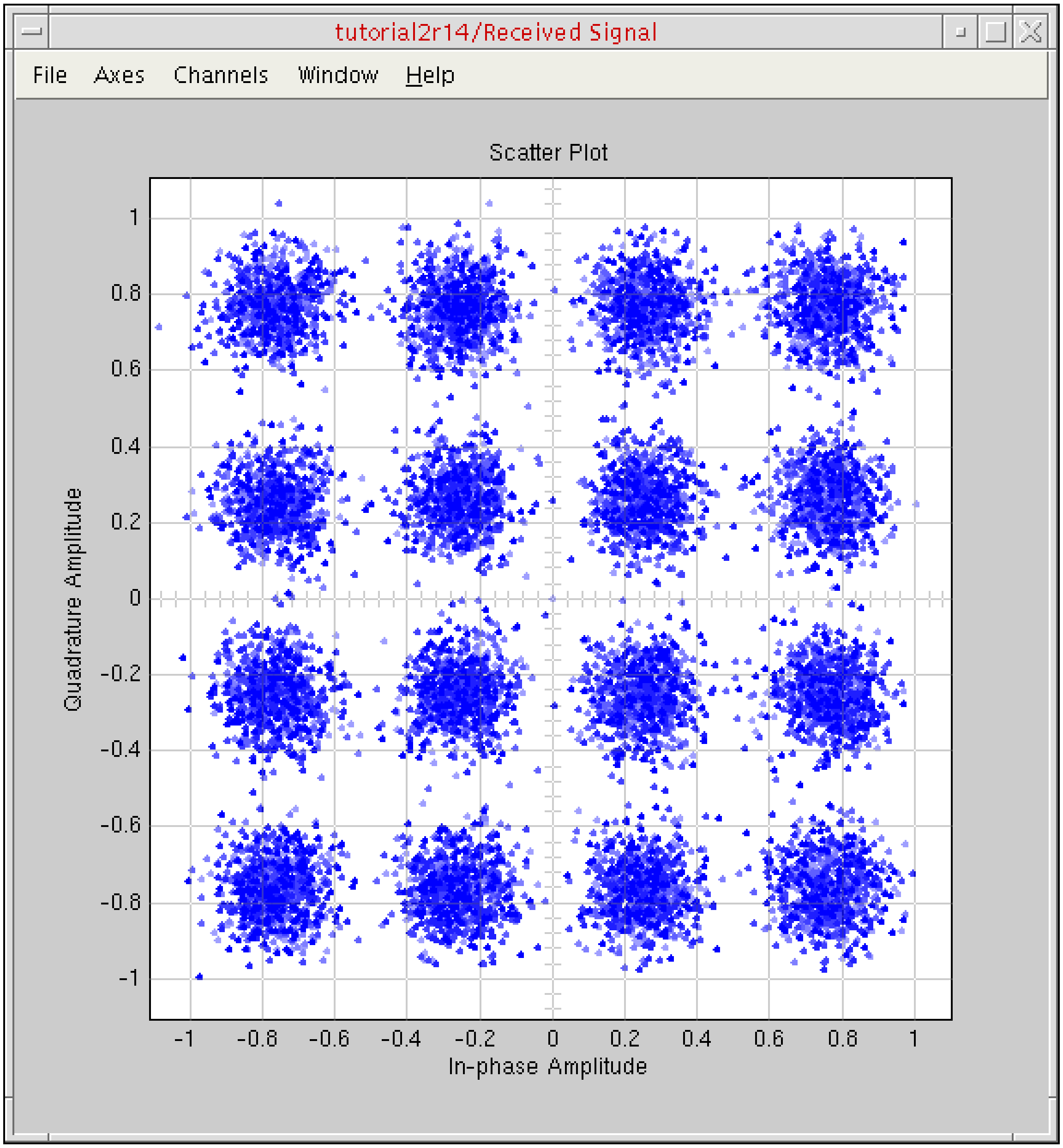
   The cosimulation runs again with the changed value.

   The number of bit errors increases (you can view this information on the testbench scheamtic). The overall system behavior is still reasonable.




tutorial2r14/Spectrum Scope2
File Axes Channels Window Help
Magnitude, dB
-10
-20
-30
-40
-50
-60
-70
-80
-40 -30 -20 -10 0 10 20 30 40
Frame: 1001
Frequency (MHz)

The scatter plot reflects the larger amplifier nonlinearity owing to the change to the value of the *CP_PA* design variable.



# Running Cosimulation by Starting the Two Simulations Separately

Previously, we demonstrated how you can run a cosimulation from ADE by setting the *Start MATLAB* option to *before AMS starts*. To leverage the full funtionality of MATLAB, you can start MATLAB from ADE and run cosimulation by starting the two simulations separately.

Before beginning this part of the tutorial, do the following:

➤ Choose *File – Exit MATLAB*.

To run cosimulation by starting the two simulations separately, do the following:

1. Start MATLAB from ADE.
2. Start the two simulations separately.

## Starting MATLAB Immediately

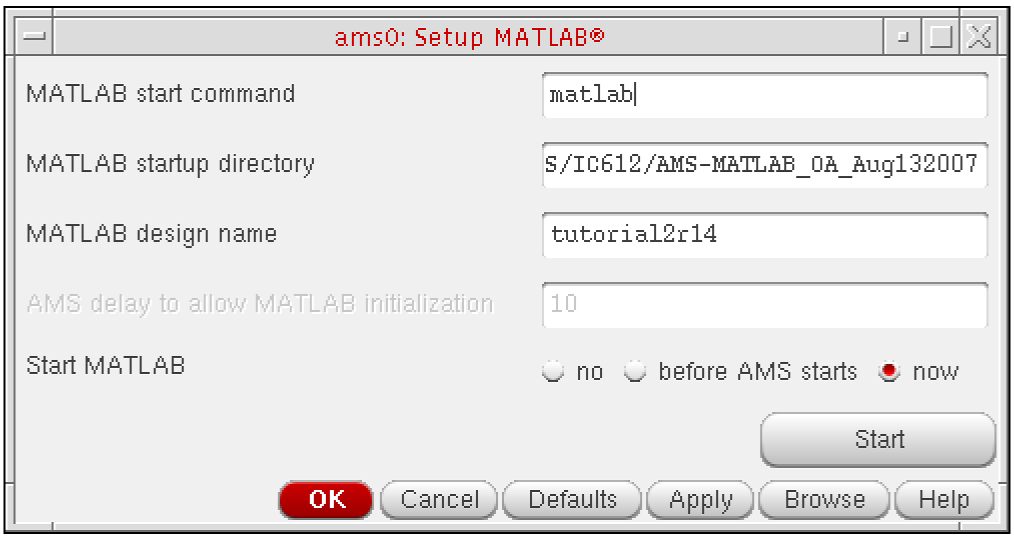
To set up MATLAB to start immediately, do the following:

1. In ADE, choose *Setup – MATLAB/Simulink – Start*.

   The Setup MATLAB form appears.

2. For *Start MATLAB*, select *now*.

   The *Start* button appears.




   **Note:** See “Starting MATLAB before AMS Starts” on page 45 for information about the *before AMS starts* selection.

3. Click *Start*.

   The MATLAB Command Window appears. The testbench schematic appears in Simulink. You are ready to cosimulate by starting the two simulations separately.

## Starting the Two Simulations Separately

To start the two simulations separately, do the following:

1. In the Simulink testbench window, choose *Simulation – Start*.

   Messages appear in the MATLAB Command Window. One of the messages is this:

   ```
   Waiting for incoming connection on port 5023, timeout: 120 sec ...
   ```

2. In the ADE window, choose *Simulation – Netlist and Run* (or click the Netlist and Run button) before this time interval expires.

   Once the `ncsim` simulation starts, the two applications establish a connection and the cosimulation proceeds. When the cosimulation finishes, the Spectrum Scope and Received Signal graph windows appear. You can view the number of bit errors on the testbench schematic.

   If you want to rerun the simulation, you will need to start Simulink and ADE again separately.

# 3

# Running Cosimulation from ADE Using the simulinkCoupler

**Note:** *AMS-MATLAB/Simulink cosimulation* stands for cosimulation using the Virtuoso® AMS Designer simulator and The MathWorks MATLAB®/Simulink®. *ADE* stands for the Virtuoso Analog Design Environment.

*Important*

Use IC 6.1.3, IUS 8.1 or later, and MATLAB704R14 or MATLAB R2007b or MATLAB R2008a for this tutorial. The estimated time to complete this tutorial is about one hour.

The `simulinkCoupler` is a parameterized cell (pcell) that does not have a fixed number of pins: You can change the number of pins to fit different designs.

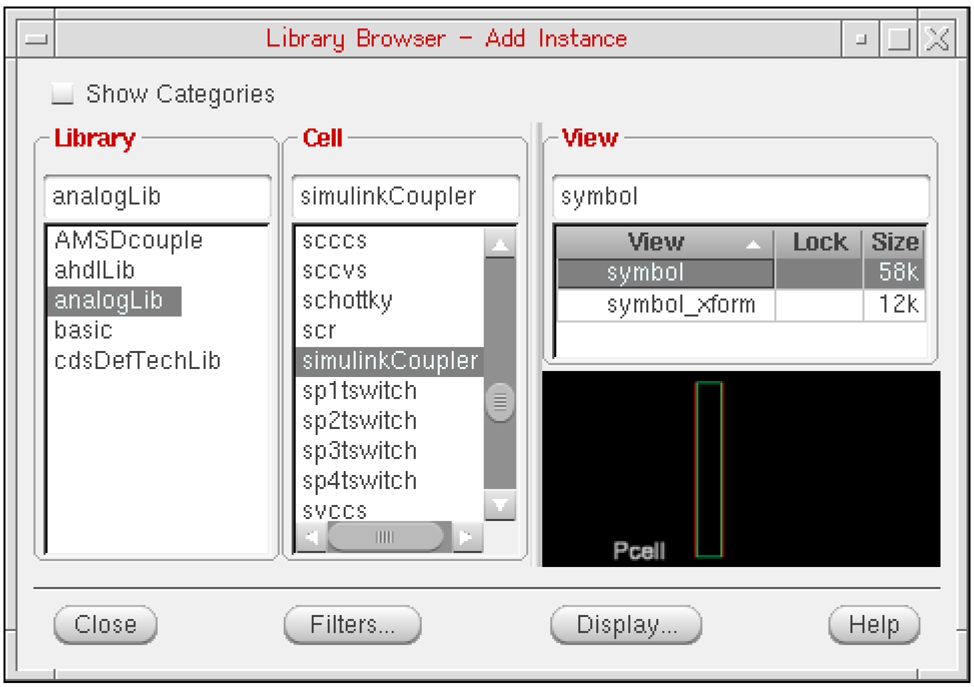
To run cosimulation from ADE using the `simulinkCoupler` block from the Cadence `analogLib` library, do the following:

1. Replace the coupler_2_3_a block with the simulinkCoupler block.
2. Specify automatic generation of the Verilog-AMS module.
3. Run cosimulation by starting two simulations separately.

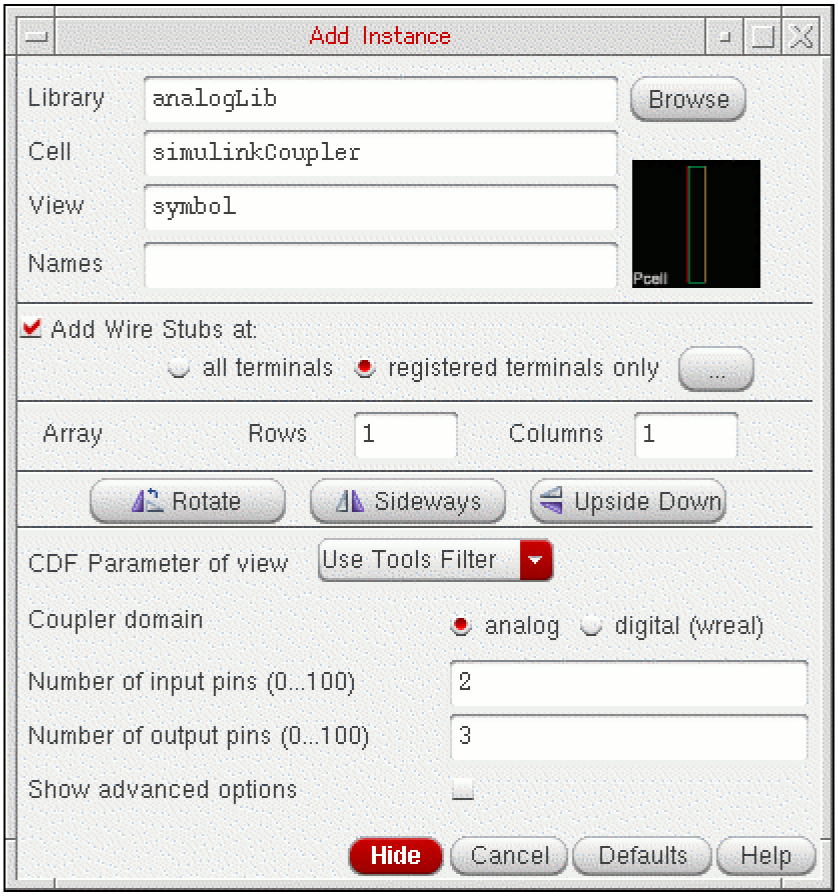
# Replacing the coupler_2_3_a with the simulinkCoupler

To replace the `coupler_2_3_a` coupler block on the Virtuoso schematic with the `simulinkCoupler` block from the Cadence `analogLib` Library, do the following:

1. In the schematic window, click the *coupler_2_3_a* instance and press *Delete*.
2. Type `i` to open the Add Instance form.
3. Click *Browse* to open the Library Browser.
4. In the *Library* column, select *analogLib*.
5. In the *Cell* column, scroll down and select *simulinkCoupler*.



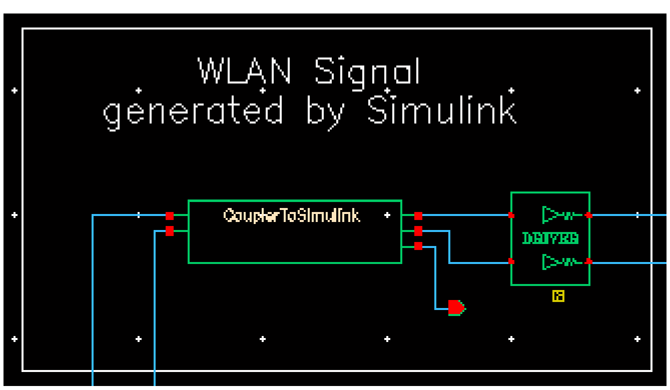
6. On the Add Instance form, change the number of input pins to 2 and the number of output pins to 3 as follows:




   a. In the *Number on input pins* field, type `2`.

   b. In the *Number of output pins* field, type `3`.

7. In the schematic window, place the `simulinkCoupler` instance.



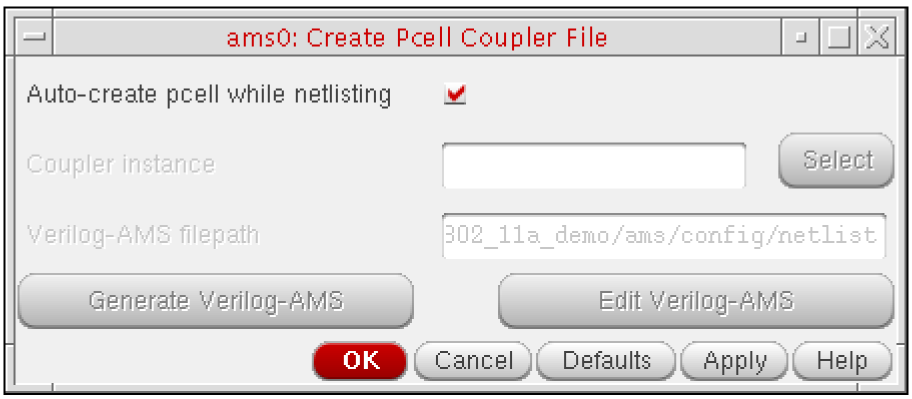
8. Choose *File – Check and Save*.

# Specifying Automatic Generation of the Verilog-AMS Module

To specify automatic generation of a Verilog-AMS module for the `simulinkCoupler` block, do the following:

1. In ADE, choose *Setup – MATLAB/Simulink – Create pcell coupler file*.

   The Create Pcell Coupler File form appears.




2. Verify that *Auto-create pcell while netlisting* is turned on.

   When *Auto-create pcell while netlisting* is turned on, ADE generates the Verilog-AMS module automatically.

3. Click *OK*.

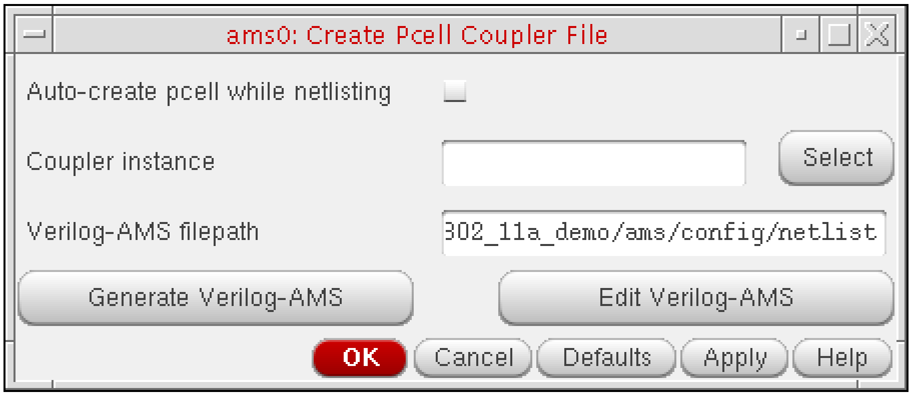
   You are ready to run cosimulation by starting two simulations separately.

# Viewing and Modifying the Verilog-AMS Module

To view and modify the Verilog-AMS module for the `simulinkCoupler` that ADE generates, do the following:

1. In ADE, choose *Setup – MATLAB/Simulink – Create pcell coupler file*.

2. On the Create Pcell Coupler File form, turn off *Auto-create pcell while netlisting*.

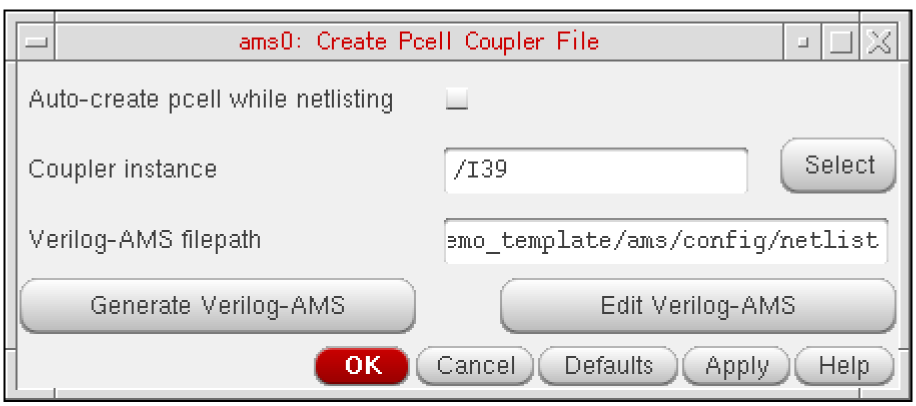



3. Click *Select* twice.

   The RF Transmitter System Testbench schematic appears in the foreground.

4. Select the *CouplerToSimulink* instance.

   The instance name appears in the *Coupler instance* field on the Create Pcell Coupler File form.




5. Click *Generate Verilog-AMS* to create the Verilog-AMS code for the pcell coupler.

   The following success message appears in the output area of the CIW:

   `Generated Verilog-AMS file successfully.`

6. On the Create Pcell Coupler File form, click *Edit Verilog-AMS*.

   The Verilog-AMS code for the `simulinkCoupler` cell appears in a text editing window.

7. Quit the text editor window.

8. (Optional) You can run cosimulation by starting two simulations separately.

**9.** To exit Cadence software and MATLAB, choose *File – Exit* in the CIW.

# 4

# Running Cosimulation from MATLAB/ Simulink

**Note:** *AMS-MATLAB/Simulink cosimulation* stands for cosimulation using the Virtuoso® AMS Designer simulator and The MathWorks MATLAB®/Simulink®. *ADE* stands for the Virtuoso Analog Design Environment.

*Important*

Use IC 6.1.3, IUS 8.1 or later, and MATLAB704R14 or MATLAB R2007b or MATLAB R2008a for this tutorial. The estimated time to complete this tutorial is about one hour.

When AMS Designer netlists a design for simulation, it generates a run script called `runSimulation` in the `simulation/`*`designName`*`/ams/config/netlist` directory. You can view the contents of this file.

See the following topics for tutorial details:


- Opening the Step 2 Tutorial Schematic on page 62
- Specifying the Run Script on page 62
- Running the Cosimulation from MATLAB/Simulink on page 66

# Opening the Step 2 Tutorial Schematic

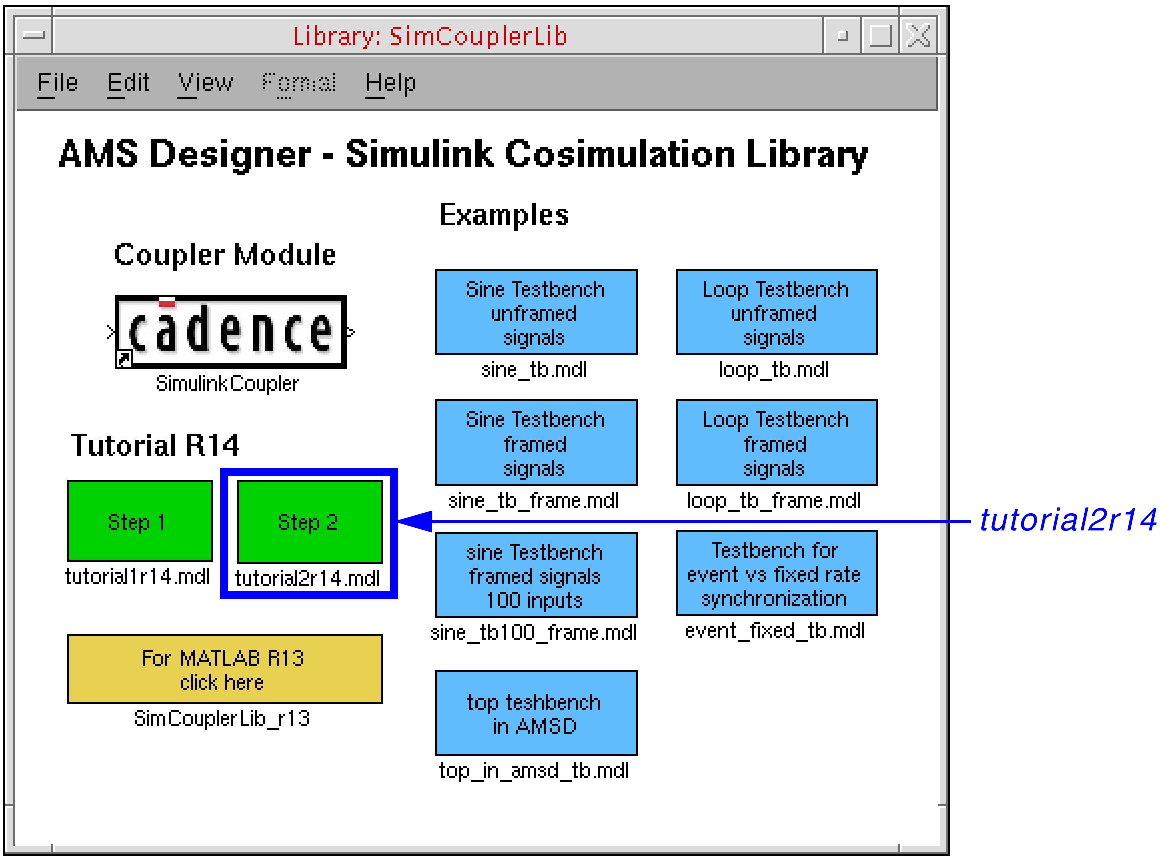
To open the *tutorial2r14* testbench schematic, do the following:

1. Start MATLAB software:

   ```
   matlab &
   ```

2. In the MATLAB Command Window, type the following command:

   ```
   open SimCouplerLib.mdl
   ```




3. Double-click *Step 2* (*tutorial2r14.mdl*, the green box on the right).

   The tutorial schematic appears in MATLAB/Simulink.

# Specifying the Run Script

To specify the run script for the cosimulation, do the following:

1. In the testbench schematic window, double-click *SimulinkCoupler*.

The Function Block Parameters form appears.

2. Turn on *Show advanced options*.

   Advanced options appear on the form.

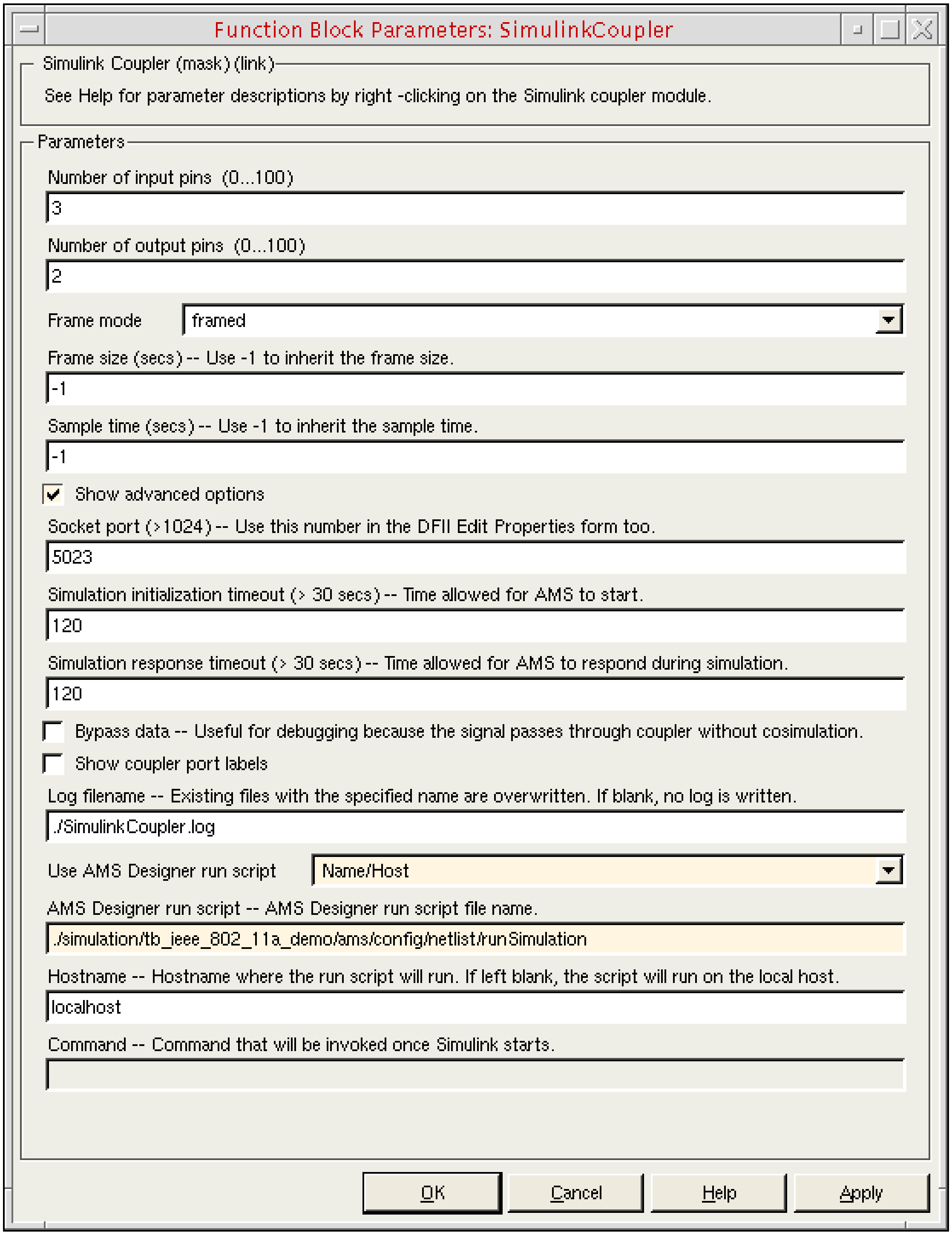
3. Scroll down to the *Use AMS Designer run script* field and select *Name/Host*.

   `./runSimulation` appears in the *AMS Designer run script* field.

   `localhost` appears in the *Hostname* field.

4. In the *AMS Designer run script* field, type the relative path to the `runSimulation` script that ADE generated earlier in this tutorial:

```
./simulation/tb_ieee_802_11a_demo/ams/config/netlist/runSimulation
```


Function Block Parameters: SimulinkCoupler
Simulink Coupler (mask) (link)
See Help for parameter descriptions by right -clicking on the Simulink coupler module.
Parameters
Number of input pins (0...100)
3
Number of output pins (0...100)
2
Frame mode
framed
Frame size (secs) -- Use -1 to inherit the frame size.
-1
Sample time (secs) -- Use -1 to inherit the sample time.
-1
Show advanced options
Socket port (>1024) -- Use this number in the DFII Edit Properties form too.
5023
Simulation initialization timeout (> 30 secs) -- Time allowed for AMS to start.
120
Simulation response timeout (> 30 secs) -- Time allowed for AMS to respond during simulation.
120
Bypass data -- Useful for debugging because the signal passes through coupler without cosimulation.
Show coupler port labels
Log filename -- Existing files with the specified name are overwritten. If blank, no log is written.
./SimulinkCoupler.log
Use AMS Designer run script
Name/Host
AMS Designer run script -- AMS Designer run script file name.
./simulation/tb_ieee_802_11a_demo/ams/config/netlist/runSimulation
Hostname -- Hostname where the run script will run. If left blank, the script will run on the local host.
localhost
Command -- Command that will be invoked once Simulink starts.
OK
Cancel
Help
Apply

**5.** Click OK.

Assuming you have specified a valid script that runs without incident, you are now ready to run the cosimulation from MATLAB/Simulink.

# Running the Cosimulation from MATLAB/Simulink

To run the cosimulation from MATLAB/Simulink using the run script you specified, do the following:

➤ In the testbench schematic window, choose *Simulation – Start*.

The cosimulation begins. The Spectrum Scope and Received Signal graphs appear. You can watch the cosimulation progress. When the cosimulation finishes, the `AMSSimulink.log` file appears in a text window.

# 5

# Running Cosimulation from the AMS Designer Environment

**Note:** *AMS-MATLAB/Simulink cosimulation* stands for cosimulation using the Virtuoso® AMS Designer simulator and The MathWorks MATLAB®/Simulink®. *ADE* stands for the Virtuoso Analog Design Environment.

*Important*

Use IC 6.1.3, IUS 8.1 or later, and MATLAB704R14 or MATLAB R2007b or MATLAB R2008a for this tutorial. The estimated time to complete this tutorial is about one hour.

You can run cosimulation from the Virtuoso® AMS Designer environment. You do not need to start or use ADE.

**Note:** This flow supports the fixed-cell coupler method only. For information about the fixed-cell coupler, see “Creating the Coupler in the Schematic Window” on page 30.

See the following topics for tutorial details:

- Reversing Changes on page 68
- Opening the Virtuoso Schematic on page 28

## Reversing Changes

If you are starting with the design as you finished with it in the previous section, you need to reverse the changes you made on the Function Block Parameters form for the *SimulinkCoupler*. To reverse your changes, do the following:

1. In the testbench schematic window, double-click *SimulinkCoupler*.

   The Function Block Parameters form appears. *Show advanced options* is turned on.

2. Scroll down to the *Use AMS Designer run script* field and select *No*.
3. Click *OK*.

## Opening the Virtuoso Schematic and Configuration

To open the Virtuoso schematic and configuration, do the following:

1. Start the Virtuoso® software:

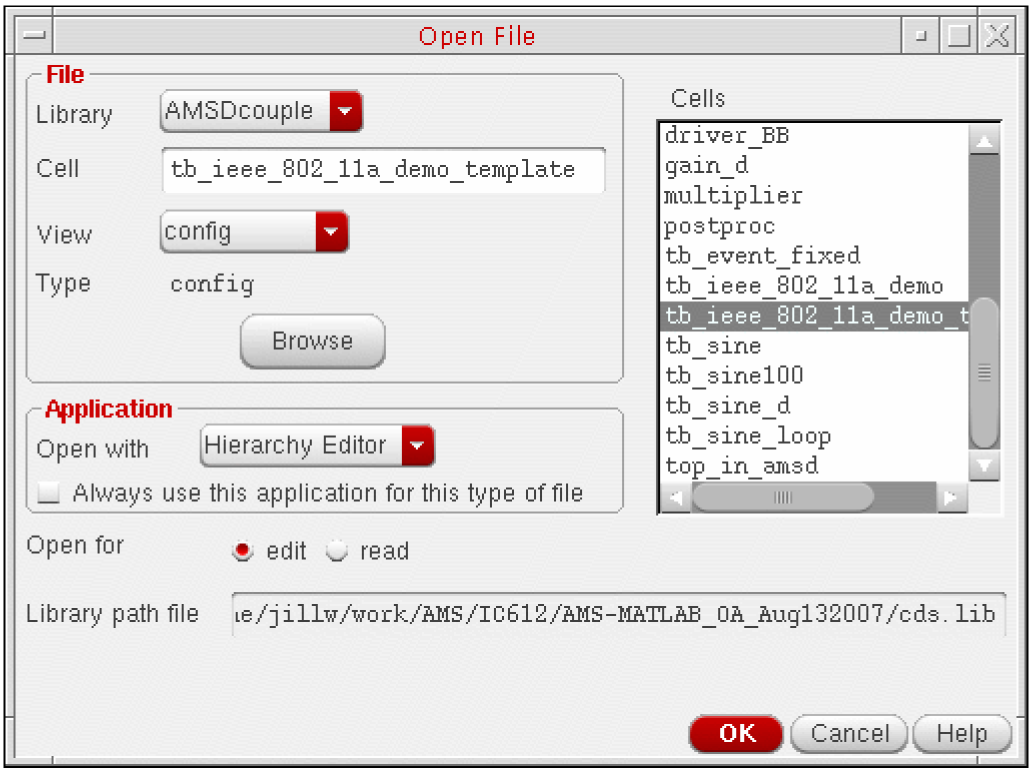
   ```
   virtuoso &
   ```

2. In the command interpreter window (CIW), choose *File – Open*.

   The File Open form appears.

3. In the *File* group box, select the following:

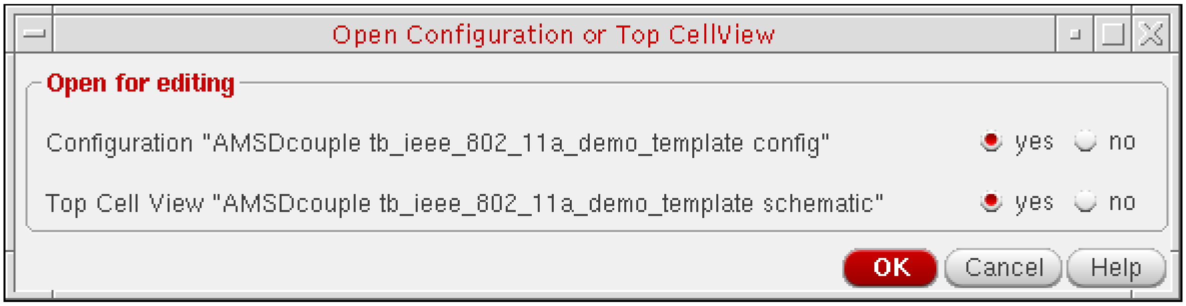
| **Field** | **Selection** |
|---|---|
| *Library* | *AMSDcouple* |
| *Cell* | *tb_ieee_802_11a_demo_template* |
| *View* | *config* |




4. Click *OK*.

   The Open Configuration form appears.

**5.** Select *yes* for both *Configuration* and *Top Cell View*.




**6.** Click *OK*.

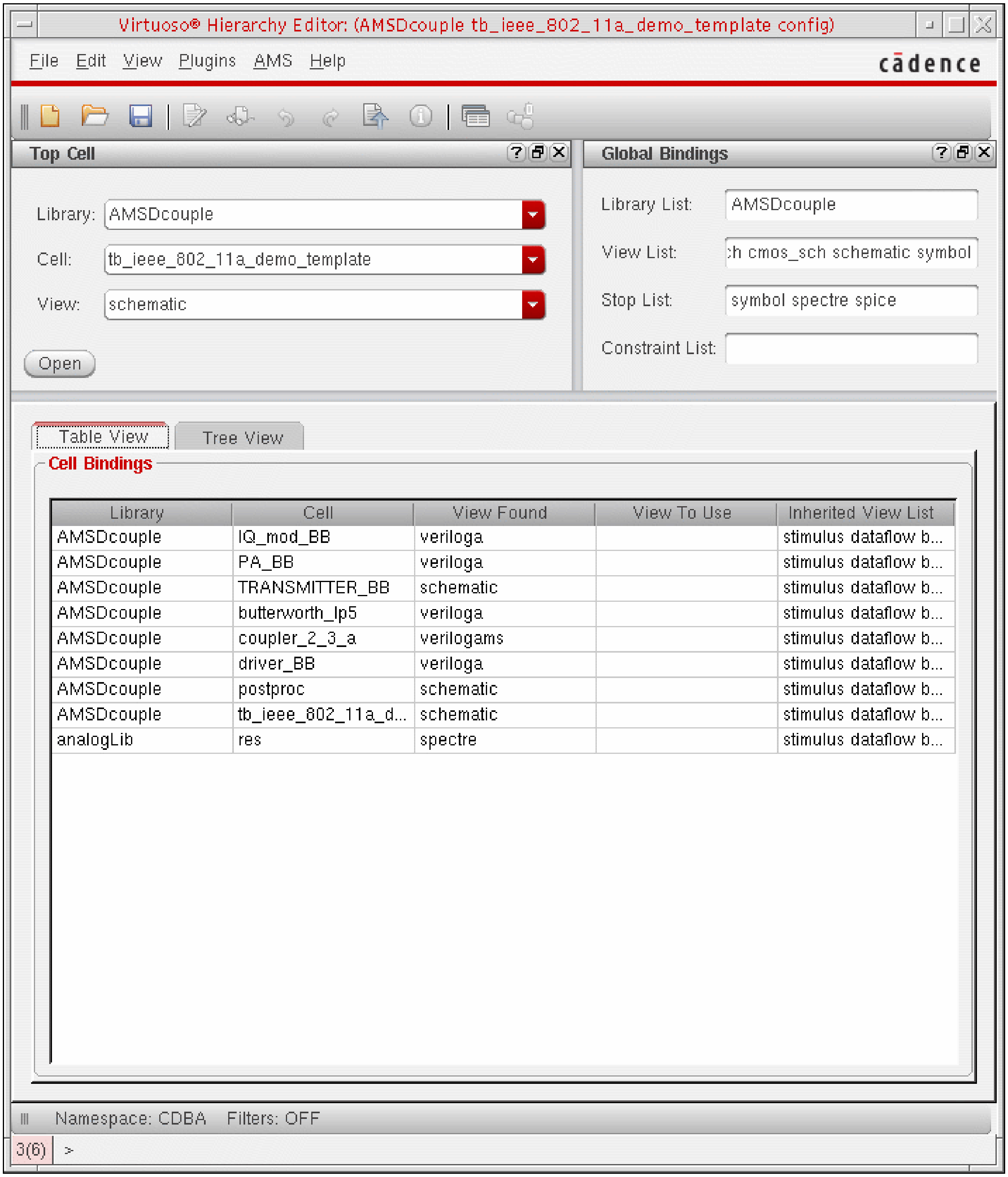
The configuration appears in the Virtuoso® Hierarchy Editor. The *RF Transmitter System Testbench* appears in a schematic window. The *RF Transmitter System Testbench* schematic contains the fixed-cell coupler you added in "Placing the coupler_2_3_a Instance on the Schematic" on page 34.

# Launching the AMS Environment from the Hierarchy Editor

After you have opened the Virtuoso schematic and configuration, you can launch the AMS environment from the hierarchy editor as follows:

➤ In the Virtuoso® Hierarchy Editor, choose *Plugins – AMS*.

*AMS* appears on the menu banner.

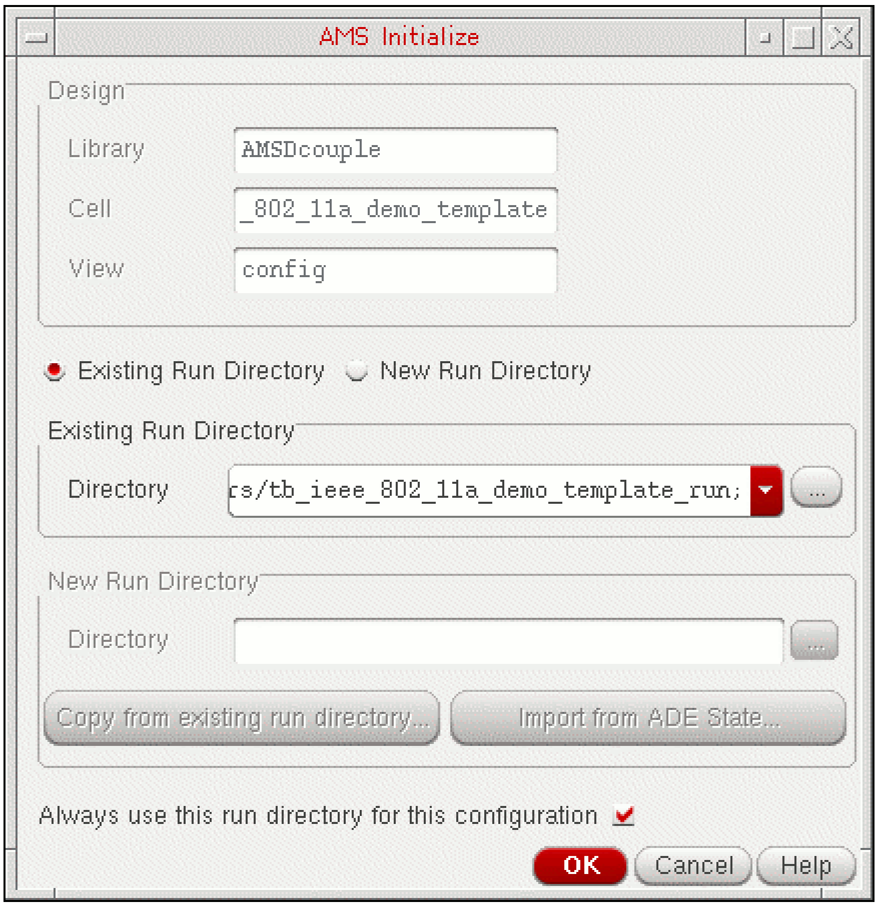



Initially, the only item available on the AMS menu is *Initialize*.

# Initializing the Run Directory for AMS

The AMS Designer environment requires that you initialize a run directory. The tutorial files you installed include a run directory. To initialize AMS to use this tutorial run directory, do the following:

1. In the Virtuoso® Hierarchy Editor, choose *AMS – Initialize*.




   The `tb_ieee_802_11a_demo_template_run` directory appears in the *Directory* field in the *Existing Run Directory* group box. You created this directory when you installed the tutorial files.

2. Click *OK*.

   The AMS Designer environment uses this run directory. Other items on the *AMS* menu become available for selection.

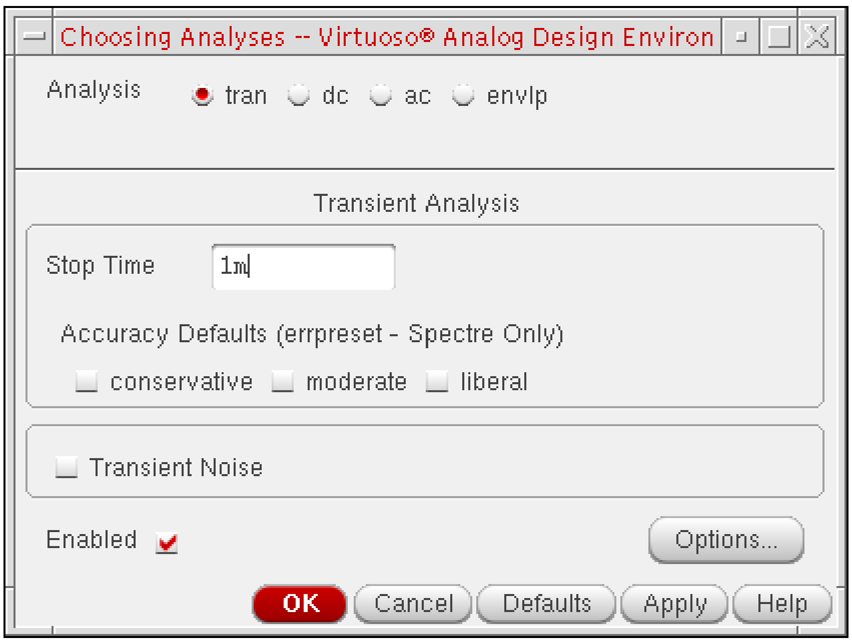
# Specifying the Transient Stop Time

To specify the transient stop time, do the following:

1. In the Virtuoso® Hierarchy Editor, choose *AMS – Detailed Setup – Analyses*.

   The Choosing Analyses form appears.

2. In the *Stop Time* field, type `1m` to match the same simulation time in MATLAB.




3. Click *OK*.

# Specifying Values for Design Variables

To specify values for the design variables in this design in the AMS environment, do the following:

1. In the Virtuoso® Hierarchy Editor, choose *AMS – Detailed Setup – Design Variables*.

   The Editing Design Variables form appears.

2. If the *GAIN_PA* (gain) and *CP_PA* (compression point) design variables (for the RF power amplifier) do not already appear in the *Design Variables* table on this form, click *Copy From*.

3. For each of the design variables (*GAIN_PA* and *CP_PA*), do the following:

   a. Click the design variable name.

      The design variable name appears in the *Name* field on the left side of the form.

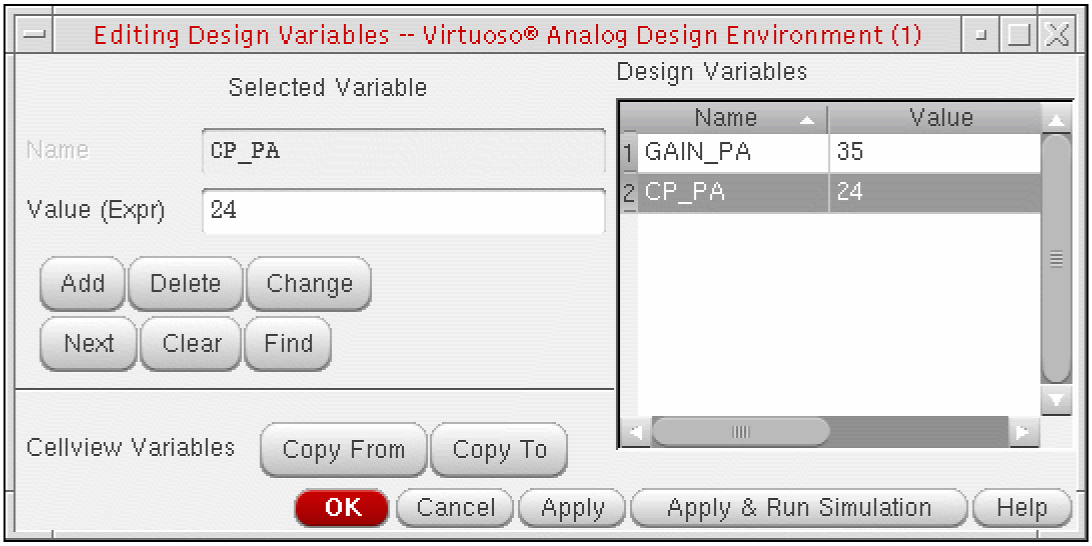
   b. In the *Value (Expr)* field, type a value:

      - For *GAIN_PA*, type `35`.
      - For *CP_PA*, type `24`.

      **Note:** The compression point design variable controls the power amplifier's linearity/nonlinearity: The smaller the number, the larger the amplifier's nonlinearity.

   c. Click *Change*.

The design variable and its value appear in the *Design Variables* table on the right side of the form.



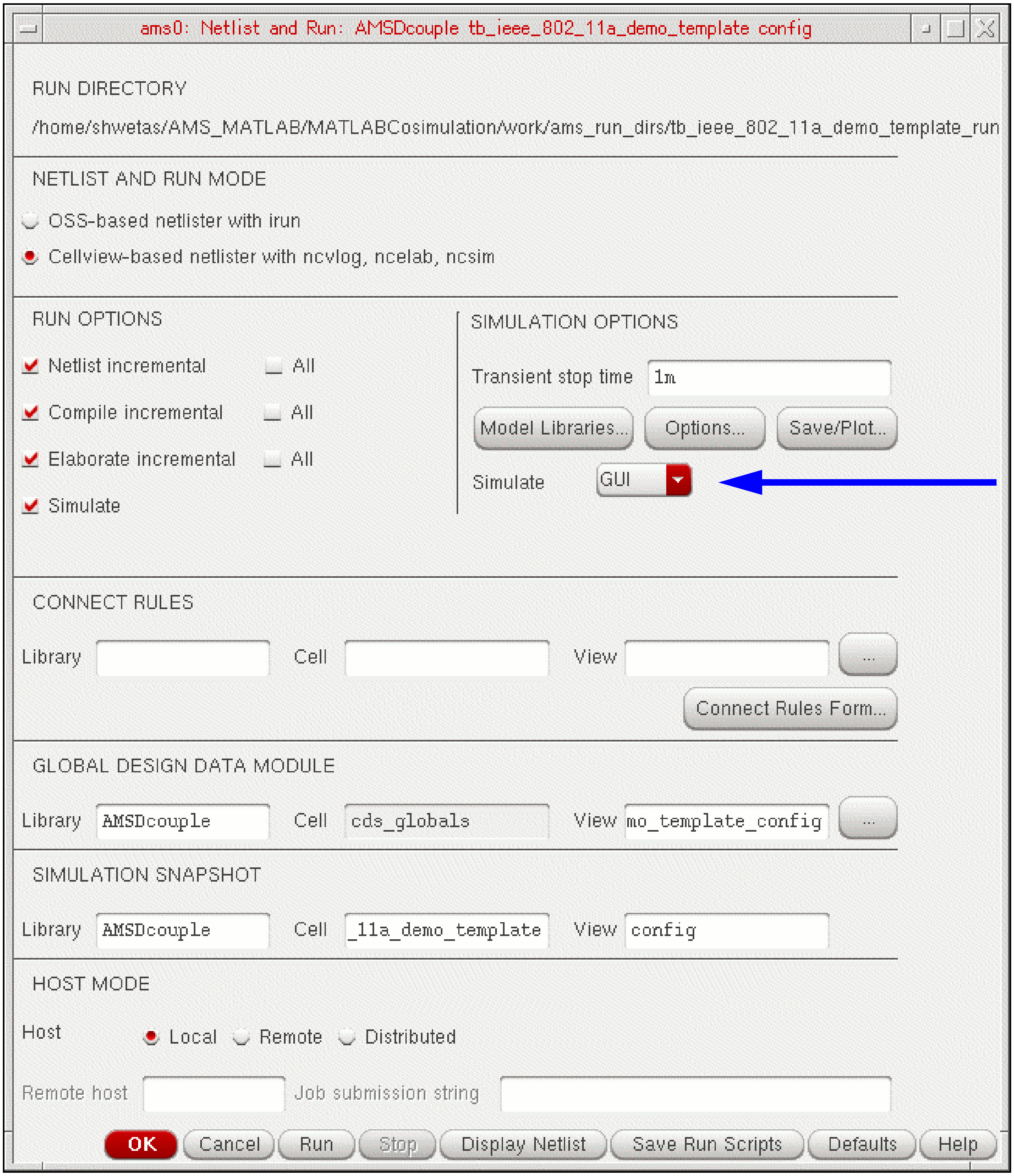

**4.** Click *OK*.

# Starting the AMS Simulation

To start the AMS simulation, do the following:

**1.** Choose *AMS – Netlist and Run*.

The Netlist and Run form appears. In the *SIMULATION OPTIONS* section of the form, notice that the *Simulate* option is *GUI*.



2. In the *CONNECT RULES* section, remove the contents of the *Library*, *Cell*, and *View* fields.

   This tutorial does not require any connect rules.

3. Click *Run*.

   The simulation starts.

# A

# Learning More about the Cosimulation Interface

**Note:** *AMS-MATLAB/Simulink cosimulation* stands for cosimulation using the Virtuoso® AMS Designer simulator and The MathWorks MATLAB®/Simulink®. *ADE* stands for the Virtuoso Analog Design Environment.

The following topics provide additional information about the AMS-MATLAB/Simulink cosimulation interface:


- Using Framed and Unframed Signals on page 80
- Running Event-Based and Fixed-Rate Simulation on page 87
- Using the Coupler Module in Feedback Loops on page 92
- Running AMS-MATLAB/Simulink Cosimulation on Other Platforms on page 98

# Using Framed and Unframed Signals

Simulink provides different testbenches for framed and unframed data.

To open the `tb_sine` configuration, do the following in the CIW:

1. Choose *File – Open*.

   The File Open form appears.

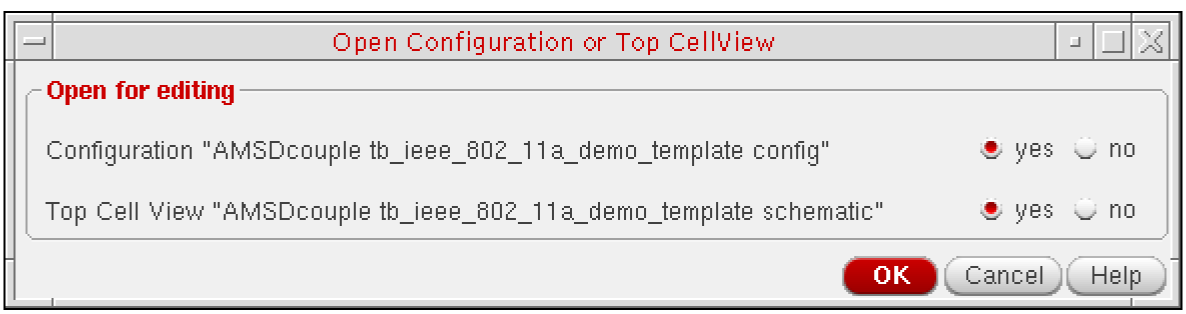
2. In the *File* group box, select the following:

   | **Field** | **Selection** |
   |---|---|
   | *Library* | *AMSDcouple* |
   | *Cell* | *tb_sine* |
   | *View* | *config* |

3. Click *OK*.

   The Open Configuration form appears.

4. In the *Open for editing* group box, selet *yes* for both *Configuration* and *Top Cell View*.




5. Click *OK*.

   The configuration appears in the Viruoso® Hierarchy Editor. The schematic appears in the schematic editor.

To start the AMS Designer simulation, do the following in the Hierarchy Editor:

1. Choose *Plugins – AMS*.

   *AMS* appears on the menu banner. You must initialize a run directory before you can access any of the *AMS* menu selections.

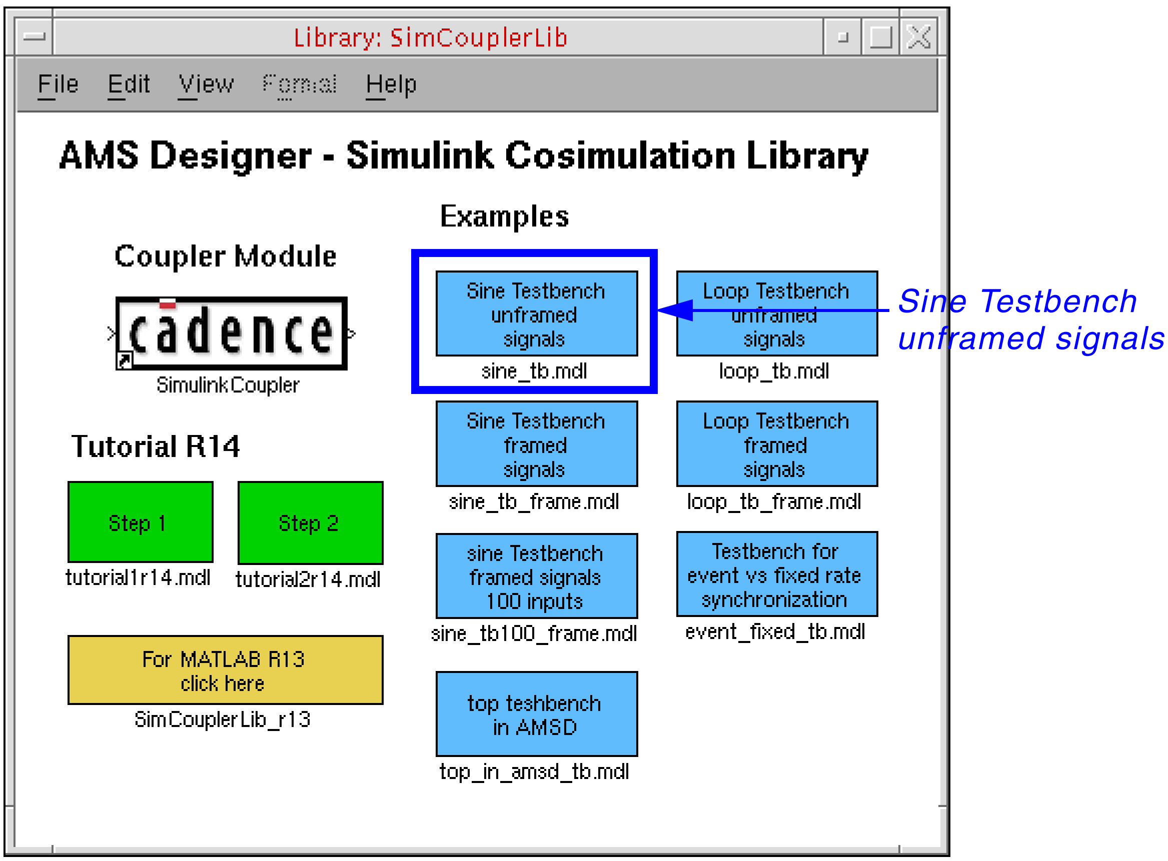
2. Choose *AMS – Initialize*.
3. Fill out the form and click *OK*.
4. Choose *AMS – Netlist and Run*.
5. (Optional) On the Netlist and Run form, click *Save/Plot* to select signals to plot.
6. On the Netlist and Run form, click *Run*.

## Using the Sine Testbench with Unframed Coupling

To open the *Sine Testbench* (*unframed signals*) from the Simulink library, do the following:

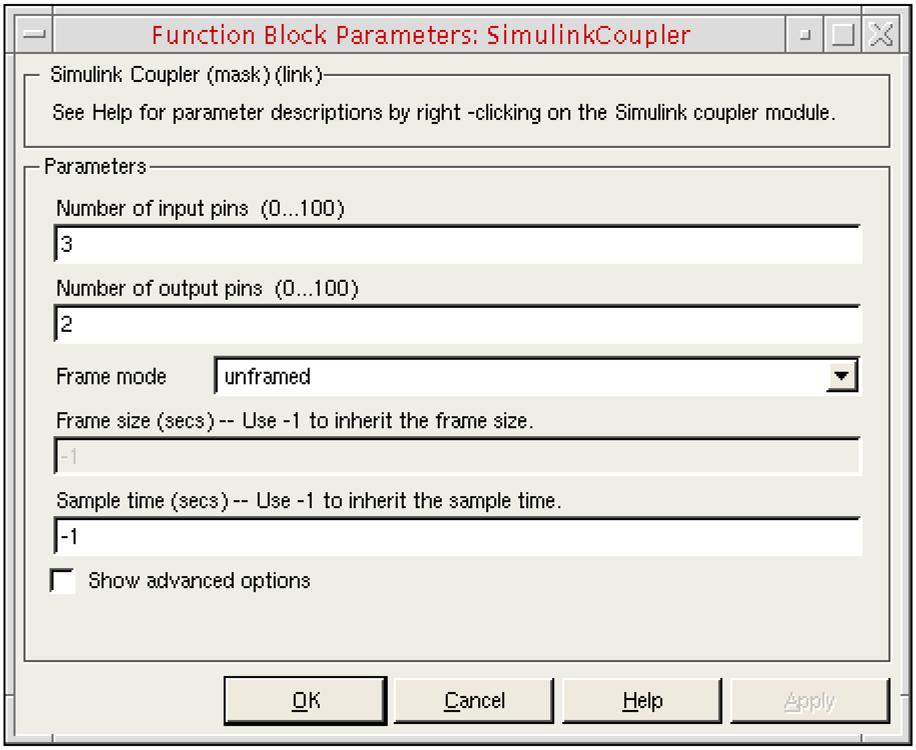
1. In the MATLAB Command Window, type the following command to open the library:

```
open SimCouplerLib.mdl
```




2. Double-click *Sine Testbench (unframed signals)*.

3. In the testbench schematic window, double-click the *SimulinkCoupler* to view its parameters.




This *SimulinkCoupler* supports event-driven simulation and fixed-rate synchronization.

For event-based simulation:

- The *Frame mode* is *unframed* and the *Frame size* field is inactive.
- The coupler module sends data to AMS Designer after each change of its input signal.
- In SimVision, you would see the sine wave exactly as the sources generate it.
- The sampling time of all sine sources in Simulink is 1e-6, by default. You can change the sampling time on the Function Block Parameters form (by typing a value in the *Sample time* field) and view the changes in synchronization.

For fixed-rate synchronization:

1. Specify a *Sample time* of `10e-6` to sample the signal at 100 kHz.

   This value satisfies the sampling theorem since the highest sine frequency is 20 kHz.

2. Choose *Simulation – Start* to start the Simulink simulation.
3. Start the AMS Designer/`ncsim` simulation (click *Run* on the Netlist and Run form).

You do not need to make any changes on the AMS Designer side.

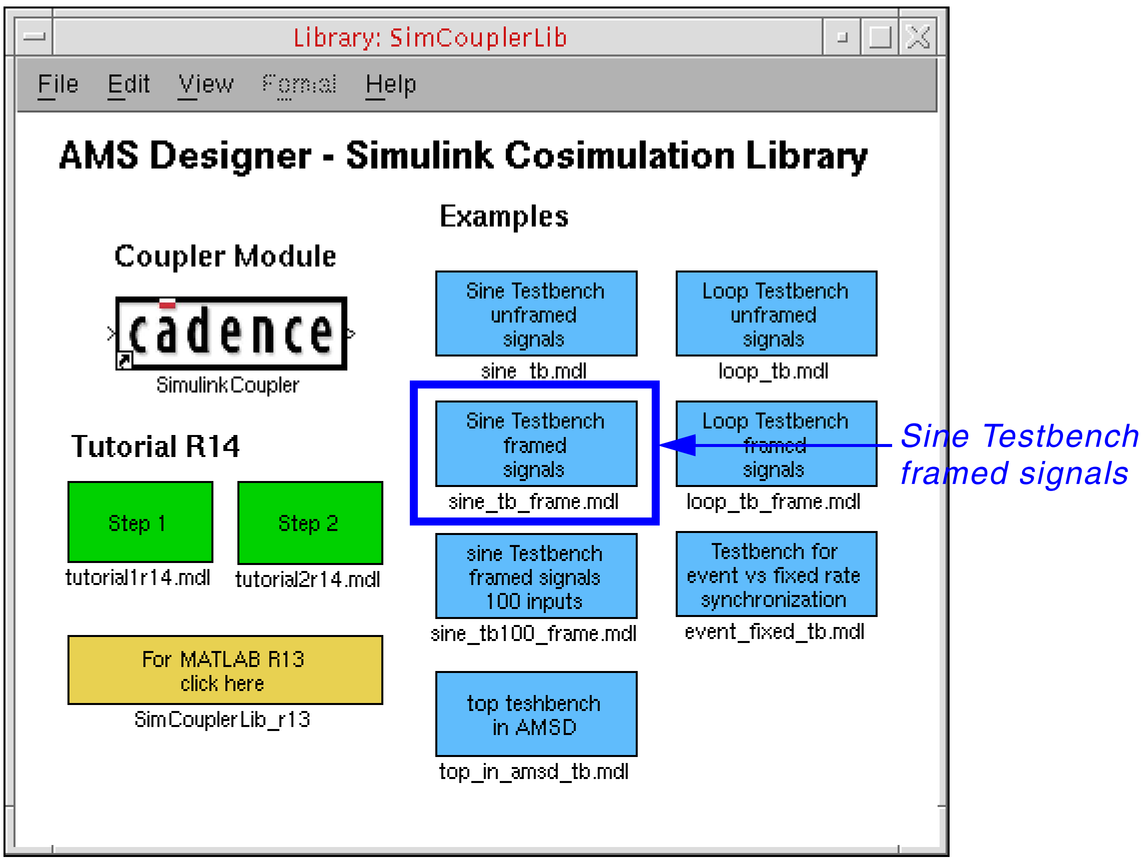
You will notice that this simulation runs faster.

4. You can view the sampled signals in SimVision and Simulink.
5. You can run this simulation using reduced values for *Sample time* (down to 1e-6) for more accurate sampling.

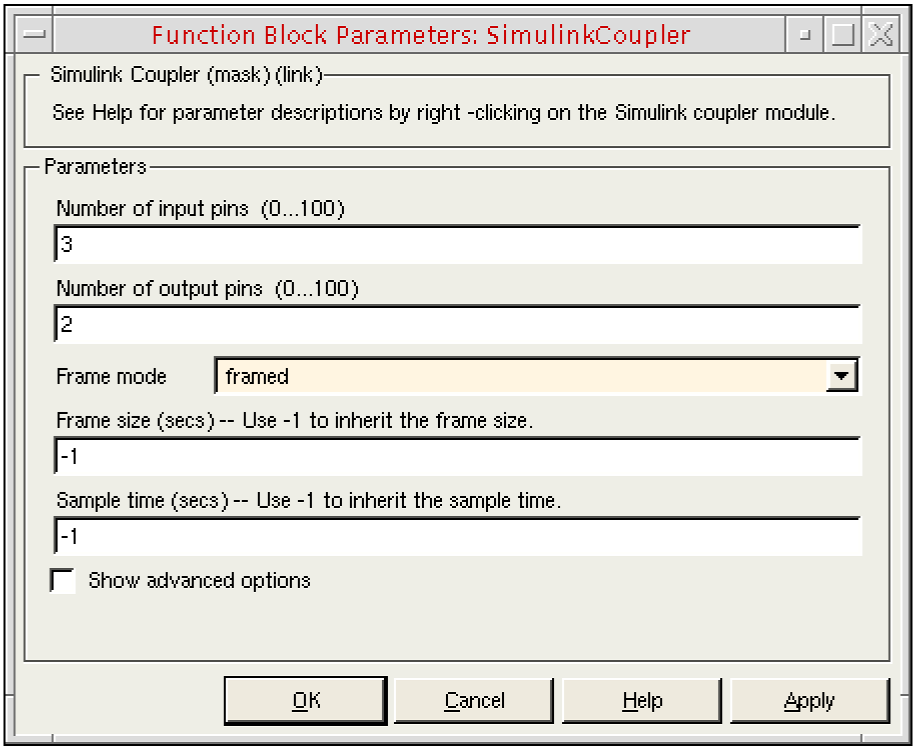
## Using the Sine Testbench with Framed Coupling

Some telecommunication simulations work with a data-stream-driven simulation. Some DSP algorithms process a data frame instead of single samples. Simulink provides framed signals for this purpose. To increase the simulation performance, cosimulation also supports framed signals. To investigate framed signals, do the following:

1. In the Library: SimCouplerLib window, double-click *Sine Testbench (framed signals)*.



**2.** In the testbench schematic window, double-click the *SimulinkCoupler* to view its parameters.




   **a.** You can specify the number of input and output ports.

   **b.** If you turn on *Show advanced options*, you can also specify the socket port.

   **c.** Notice that the *Frame mode* is *framed*.

   **d.** Simulink determines the frame length from the connected input signals.

**3.** If you simulated the unframed example before, rerun the simulator and view the signals in Simulink.

**4.** In the *Frame size* field, type `10`.

**5.** Rerun the AMS Designer simulator.

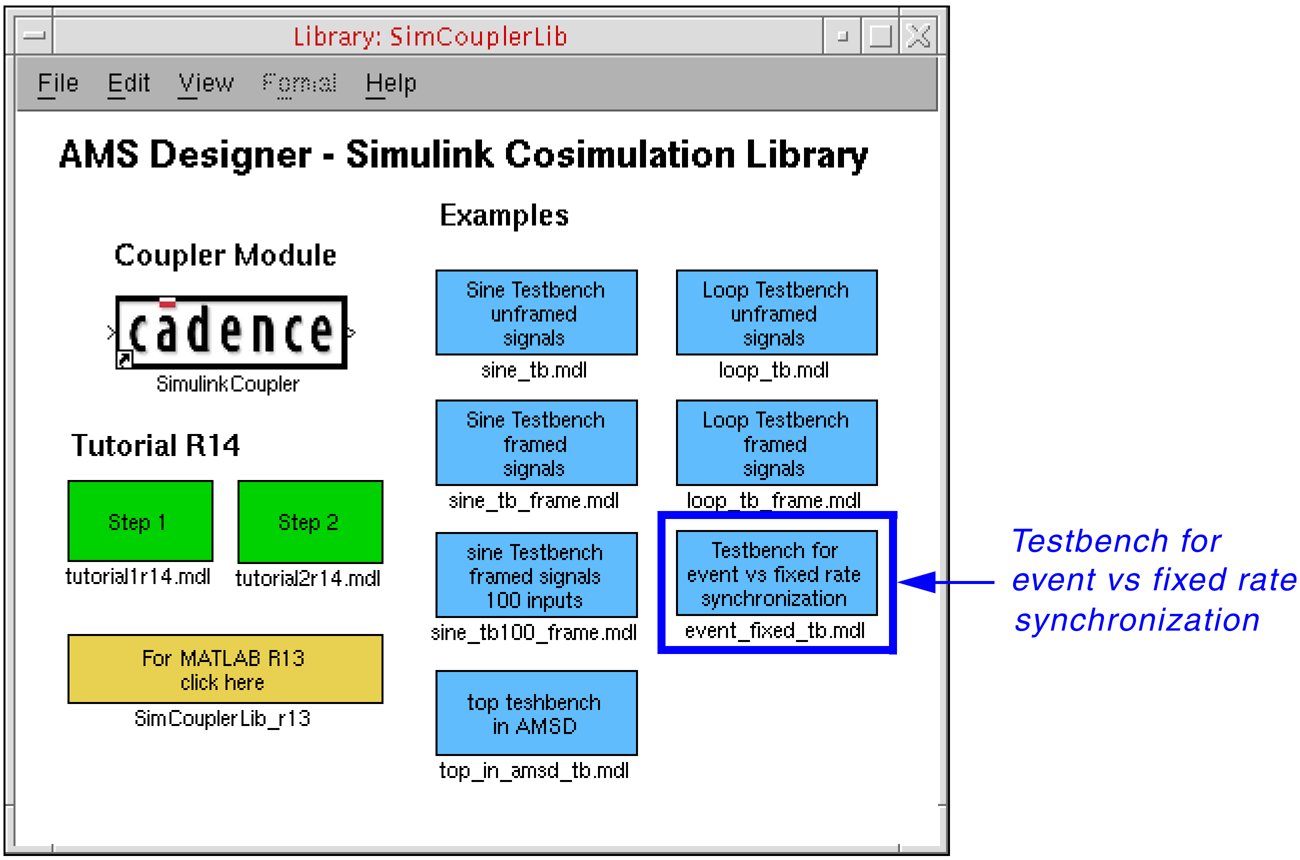
The result is the same. The simulation takes a bit more time.

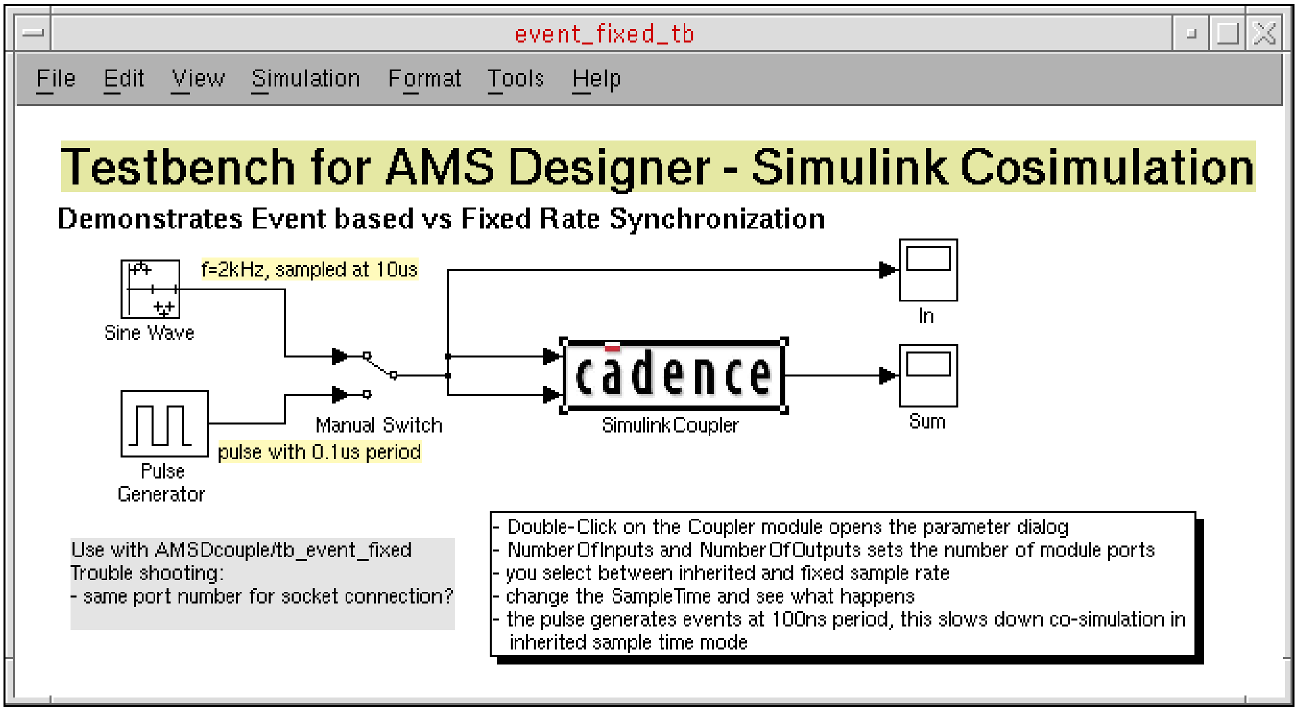
You can explore several different frame sizes. The maximum valid frame length is 10000.

## Running Event-Based and Fixed-Rate Simulation

Fixed-rate synchronization can be useful if the model contains signals with different sampling rates (high and low) and the interface connects only to a low sampling rate block. If the signals exchanged between AMS Designer and Simulink are at the lower rates, fixed-rate synchronization with a dedicated sample time can improve the simulation performance significantly. Here is an example:

1. In the Library: SimCouplerLib window, double-click *Testbench for event vs fixed rate synchronization*.






- The sine wave generator (*Sine Wave*) produces a 2 KHz sine wave sampled at 10 us. Simulink transmits the sine wave signal to the AMS Designer simulator.
- The pulse generator (*Pulse Generator*) produces a high-frequency pulse with a period of 100 ns (which is 100 times faster than the sampling rate of the sine wave).

**2.** In the Virtuoso® command interpreter window (CIW), choose *File – Open*.

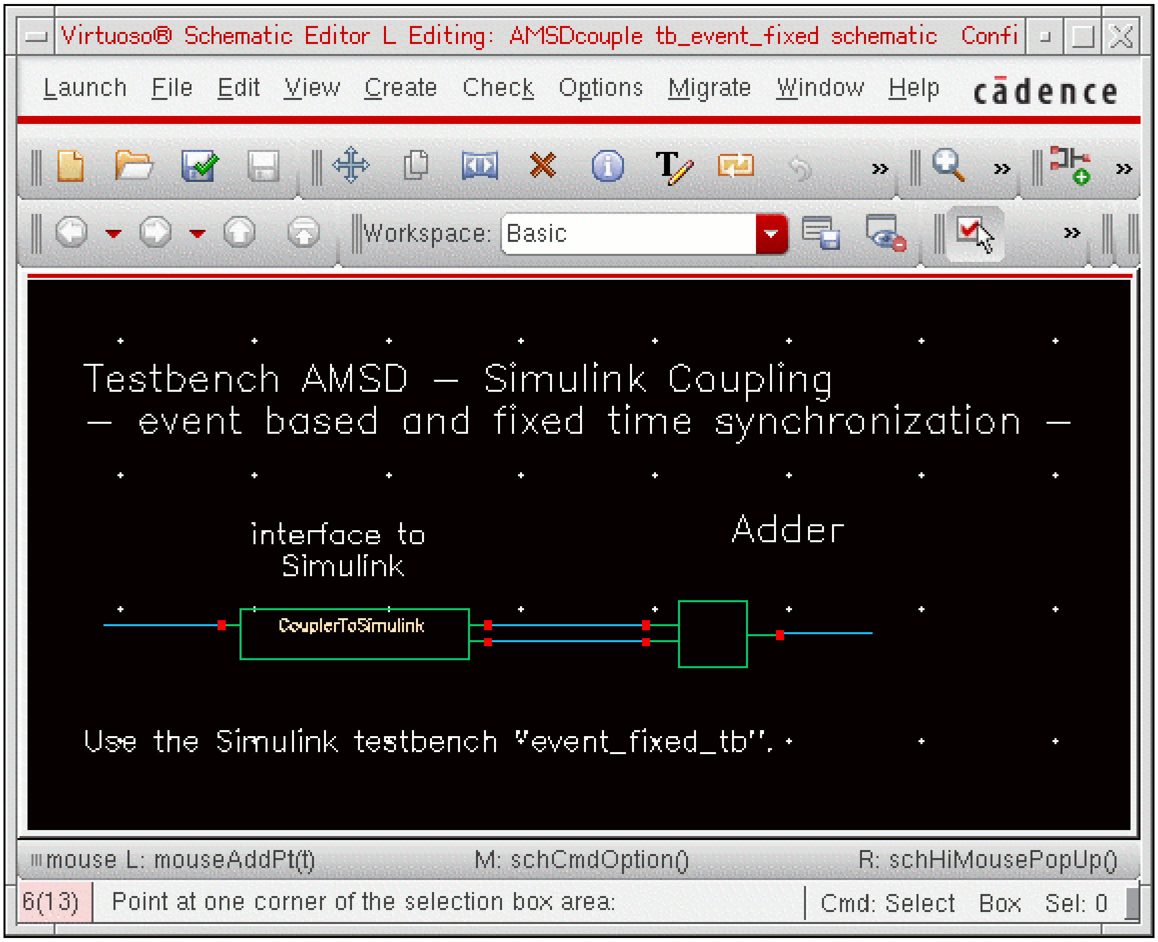
The File Open form appears.

**3.** In the *File* group box, select the following:

| **Field** | **Selection** |
|---|---|
| *Library* | *AMSDcouple* |
| *Cell* | *tb_event_fixed* |
| *View* | *config* |

**4.** On the Open Configuration form, select *yes* for *Top Cell View* to open the schematic.

**5.** Click *OK*.




The *Adder* component adds the two signals from Simulink.

**6.** To start the simulation in event-based mode, do the following:

**a.** Double-click the coupler module on the Simulink schematic.

**b.** Verify that the *Sample time* is `-1` to turn on the event-based mode.

**c.** Click *OK* to close the form.

**d.** Choose *Simulation – Start*.

**7.** In SimVision, display *Signal_1*, *Signal_2*, and *Sum*.

**8.** Start the AMS Designer simulation.

The simulation takes about 30 seconds. The sine waves appear in Simulink and in SimVision. You can mark the computed data points.

9. After finishing the run, go back to Simulink and change the sync mode in the coupler module to fixed rate with a sample time of 0.00001 (equal to the sampling rate of the sine source).
10. Run the Simulink simulation again.
11. Go back to SimVision and restart the simulator.
12. Start the AMS Designer simulation.

    The simulation finishes after about 5 seconds. The high-frequency pulse no longer influences data exchange between the simulators.

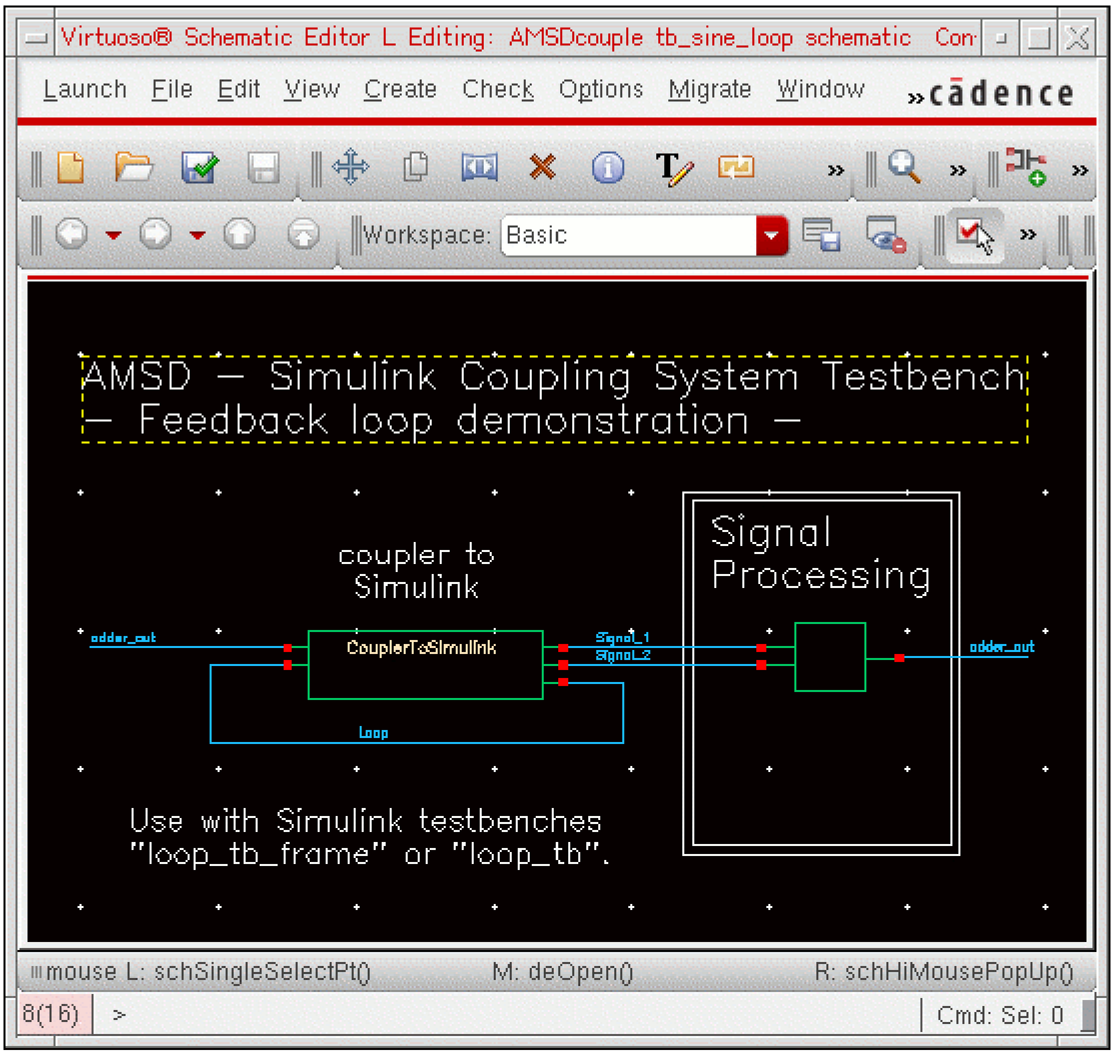
You can resimulate this example using different sample times.

In other examples where the sampling rate changes over time, the event-based synchronization might be the better choice. Decide carefully what synchronization scheme fits best with your design.

**Note:** If you look at the computed signal points in the SimVision waveform plot, you will see that the values are not necessarily updated at equidistant points even if you choose fixed-rate in the Simulink coupler. AMS Designer's analog solver controls the timing of these data points and can introduce more simulation points if necessary between two synchronization points.

# Using the Coupler Module in Feedback Loops

You can use coupling in feedback loops. In this example testbench (`tb_sine_loop`), Simulink generates two sine waves and transmits them to AMS Designer and AMS Designer feeds one of the signals back to Simulink. The second signal is unused.




Simulink compares the feed-through sine wave with the original signal. The coupler has additional input and output pins and there is another feedback loop inside AMS Designer. Simulink adds a constant of 0.1 to the coupler output and the signal connects to the third coupler input. After each cycle, Simulink increases the signal by 0.1. You can observe this behavior in the Simulink or in the AMS Designer window. At the beginning of simulation, the program initializes the signal to zero.

To open the Virtuoso® schematic, do the following:

1. In the command interpreter window (CIW), choose *File – Open*.

   The File Open form appears.

2. In the *File* group box, select the following:

   | **Field** | **Selection** |
   |---|---|
   | *Library* | *AMSDcouple* |
   | *Cell* | *tb_sine_loop* |
   | *View* | *config* |

3. On the Open Configuration form, select *yes* for *Configuration* and for *Top Cell View*.
4. Click *OK*.
5. In the Virtuoso® Hierarchy Editor, choose *Plugins – AMS*.
6. Once you have initialized the run directory for AMS, choose *AMS – Netlist and Run* and click *Run*.
7. In SimVision, display the *Loop* signal.

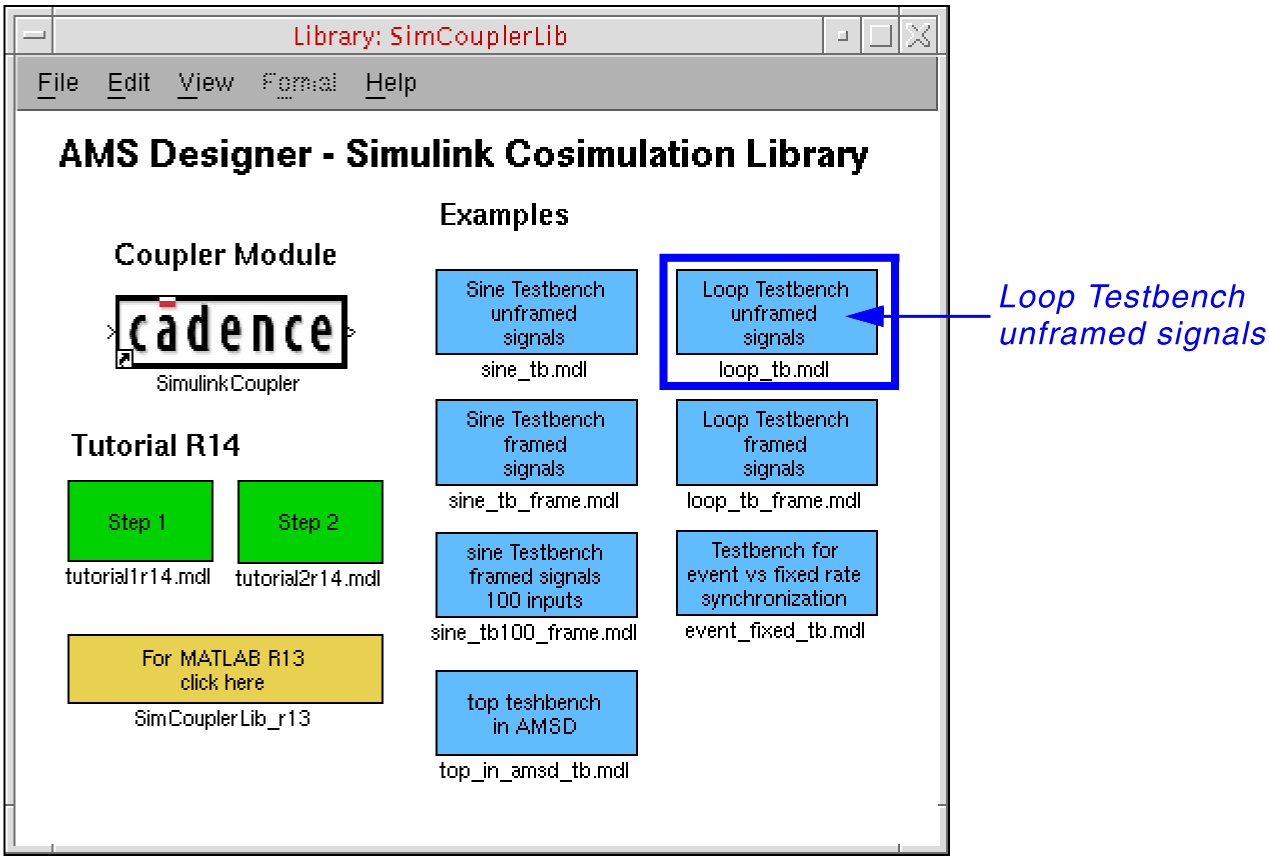
You can decide whether to use the *Loop Testbench* with unframed or framed signals in Simulink:

- Using the Sine Testbench with Unframed Coupling on page 82
- Using the Sine Testbench with Framed Coupling on page 85

## Using the Loop Testbench with Unframed Coupling

To use the loop testbench with unframed coupling, do the following:

1. In the Library: SimCouplerLib window, double-click *Loop Testbench (unframed signals)*.



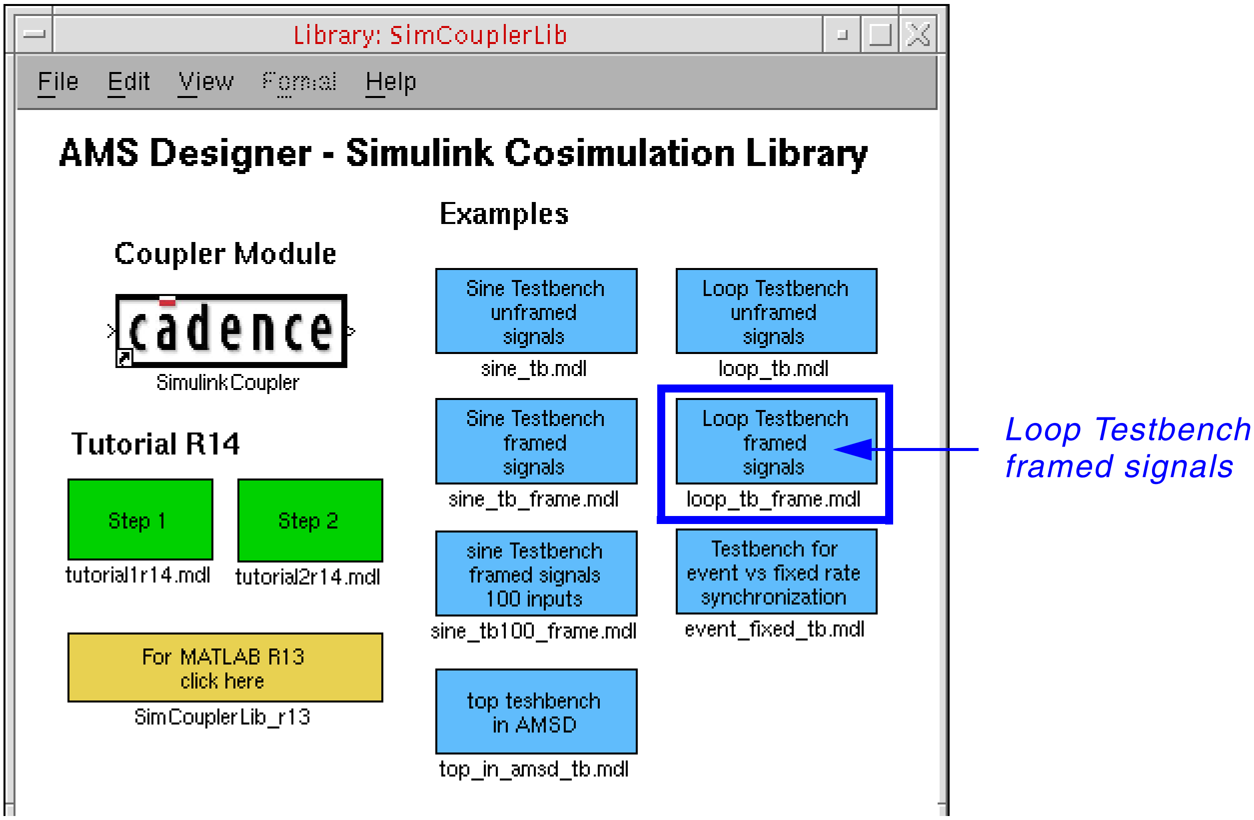

2. On the testbench schematic, double-click *SimulinkCoupler* to review its parameters.
3. Choose *Simulation – Start*.
4. Start the AMS Designer simulation.
5. After each cycle, increase the signal by 0.1.

   A ramp appears in the plot windows. The two sine waves match.
6. Switch the coupling block to fixed port rate and view the changes on the ramp and the two sine waves.

## Using the Loop Testbench with Framed Coupling

To use the loop testbench with framed coupling, do the following:

1. In the Library: SimCouplerLib window, double-click *Loop Testbench (framed signals)*.




2. On the testbench schematic, double-click *SimulinkCoupler* to review its parameters.
3. Choose *Simulation – Start*.
4. Start the AMS Designer simulation.

   The signal increases once for each frame.

   The signal values are equal within each frame.

   The signal has steps, depending on the frame size.

   The frame size in the feedback loop path equals the frame size of the sine waves (defined in the buffer blocks).
5. Change the frame size and view the changes in the results.

**Note:** You must handle feedback loops in cosimulation very carefully. In this example, you can see that the behavior of feedback signals can vary depending on different simulation settings. Review your design very carefully around possible loops and decide how to integrate the loops in the cosimulation. With the default settings, Simulink displays a warning:

```
Warning: Block diagram 'loop_tb' contains 1 algebraic loop(s).
```

For more information about loops, use type the following command in the MATLAB Command Window:

```
sldebug loop_tb
```

The Simulink debugger command-line prompt appears:

```
(sldebug @0): >>
```

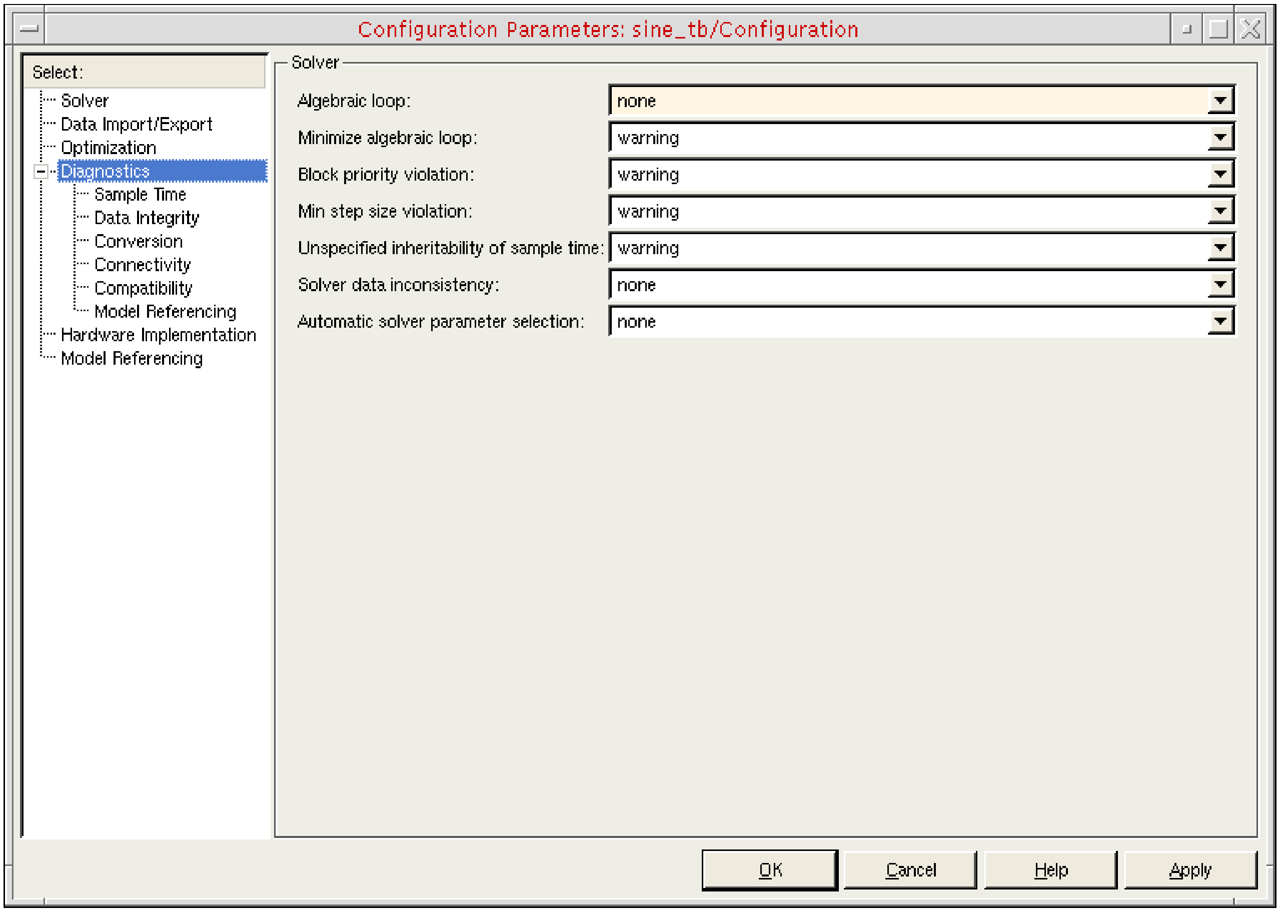
To eliminate the warning message, do the following:

1. In the Simulink testbench window, choose *Simulation – Configuration Parameters*.

   The Configuration Parameters form appears.

2. In the *Select* tree, select *Diagnostics*.

**3.** In the *Algebraic loop* field, select *none*.




**4.** Click *OK*.

# Running AMS-MATLAB/Simulink Cosimulation on Other Platforms

You can run AMS-MATLAB/Simulink cosimulation on Linux, Solaris, HP, IBM, and Windows platforms. If you are running cosimulation from MATLAB under Windows, you must add the path to the coupler in the Cadence® IUS installation hierarchy in MATLAB using the `addpath` command. For example:

```
addpath('/grid/cadence/install/ius57/lnx86/tools/affirma_ams/etc/matlab');
addpath('/home/cadence/tutorial/AMS-MATLAB/matlab');
addpath('/home/cadence/tutorial/AMS-MATLAB/matlab/tutorial');
```

# B

# Troubleshooting

**Note:** *AMS-MATLAB/Simulink cosimulation* stands for cosimulation using the Virtuoso® AMS Designer simulator and The MathWorks MATLAB®/Simulink®. *ADE* stands for the Virtuoso Analog Design Environment.

The coupler module prints messages to the MATLAB main window and to the AMS Designer/ `ncsim` console. The UNIX version of the Simulink coupler block also writes messages to the `SimCoupler.log` file in the MATLAB run directory. These messages can help you troubleshoot problems.

## Possible Problems

Here are some possible problems you might encounter and what you can do.

### *SimVision does not come up, NC elaboration failed*

Netlist the design again and restart elaboration/simulation. Check for error messages in the output window or in the log files.

**Note:** If you are using the ADE flow, Simvision does not appear unless you have specified interactive mode.

### *The cosimulation cannot be established*

If AMS Designer/`ncsim` terminates immediately after starting the simulation run and the AMS Designer/`ncsim` console closes, go to the dedicated AMS run directory and review the `ncsim.log` file. Coupler module messages appear at the end of this file.

For example:

```
ncsim> run
initializing couple module 'tb_sine.I6'
ERROR: connecting to socket failed, sockfd=16, host=localhost, port=5023
       errno=111: Connection refused
```

```
ERROR: can't create new connection to 'localhost' at port 5023 (Master simulator
not running?)
```

The coupler module cannot connect to MATLAB. Here are some possible reasons and work-arounds:

1. The Simulink simulation did not start or has stopped as a result of the time-out settings.

   Try again. Start the Simulink simulation before the AMS Designer/`ncsim` simulation. Wait for the MATLAB initialization phase to finish before starting AMS Designer. Do not wait longer than the time-out before starting AMS Designer.

2. The simulators are using different socket ports.

   Check the settings in MATLAB and in the Virtuoso schematic editor. If there are differences, change the port numbers so that they match.

3. MATLAB does not run on the specified host.

   Check the `Hostname` parameter of the coupler module on the Virtuoso schematic. The default value is `localhost`.

4. Another service is using the socket port.

   Change to another port on both sides.

### *The simulation ends before the desired time*

The cosimulation terminates when one of the simulators reaches its stop time. On the AMS Designer side, you configure this setting on the ADE Choosing Analyses form or in the Virtuoso® Hierarchy Editor on the Netlist and Run form (or by choosing *AMS – Detailed Setup – Analyses*). In Simulink, choose *Simulation – Configuration Parameters* and type a simulation stop time in the *Stop time* field on the Configuration Parameters form.

### *When using framed signals the Simulink coupler module reports the error "lookup ports failed"*

The software limits the maximum frame size to 10,000 for each input signal. If your application requires a larger frame size, contact Cadence for support.

*Important*

All connected signals must have the same frame size.

# Possible Simulink Errors

Here are some possible Simulink error messages and how you might handle them.

***"Illegal rate transition found involving block 'sine_tb_frame/SimCoupler' at input port 2. A rate transition block must be inserted between the two blocks."***

Simulink generates this error by default only for the multitasking solver mode of its fixed-step solver when signals with different sample rates connect to the same block. In framed mode, the cosimulation produces wrong results when signals with different sample rates connect to the Simulink coupler module.

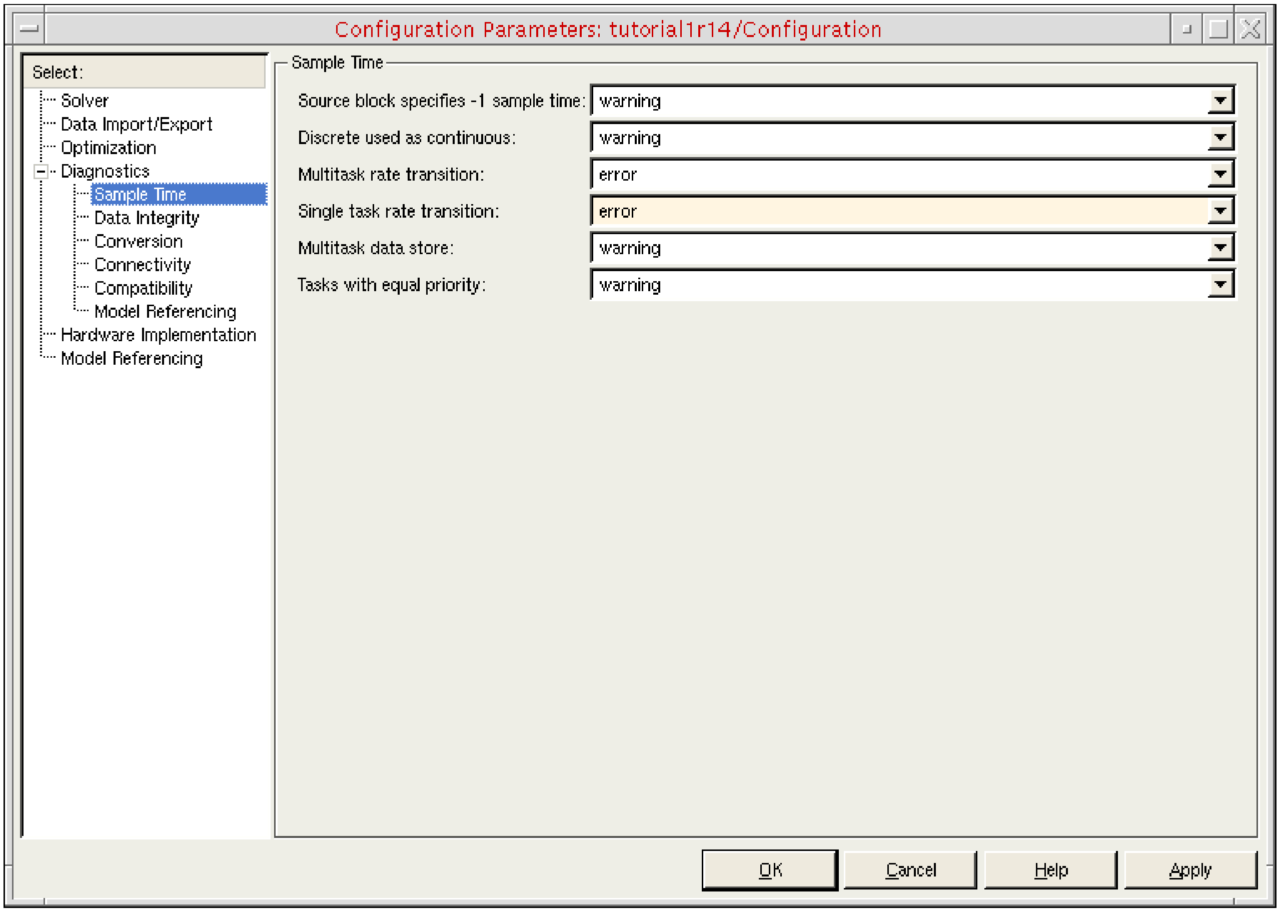
We recommend that you turn on this error message for single-task mode. All models in the AMS Designer/Simulink Cosimulation Library that use framed signals have this option turned on. To turn on this error message for single-task mode, do the following:

1. In the testbench schematic window, choose Simulation – Configuration Parameters.

   The Configuration Parameters form appears.

2. Select *Diagnostics*, *Sample Time*.

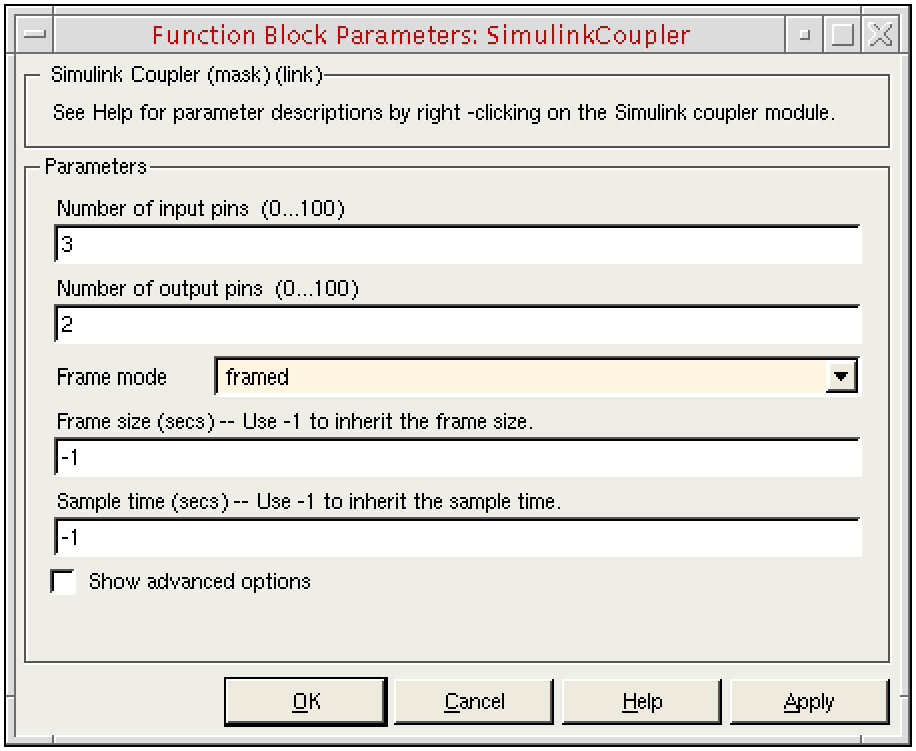
**3.** In the *Single task rate transition* field, select *error*.




**4.** Click *OK*.

***"Error reported by S-function 'SimCoupler' in block 'sine_tb_frame/SimCoupler': could not determine block sample rate, try using the fixed step solver"***

You can only use the Simulink coupler module whose *Frame mode* is set to *framed* in models that have fixed sample rates. (You specify the *Frame mode* on the Function Block Parameters form.)




You can use the variable step solver as long as the coupler module has a fixed sample rate. To turn on sample time coloring so that you can inspect the sample rates in your model, do the following:

➤ In the testbench schematic window, choose *Format – Port/Signal Displays – Sample Time Colors*.

Black and gray indicate a variable sample time, which you cannot have in framed mode. Also, you cannot have different sample times at the coupler module (see the previous Simulink error message).