Allegro® SI SigXplorer User Guide

Product Version 16.5
May 2011

Document Last Updated On: May 14, 2012

Contents

1

Introduction to SigXplorer

Topics in this chapter include

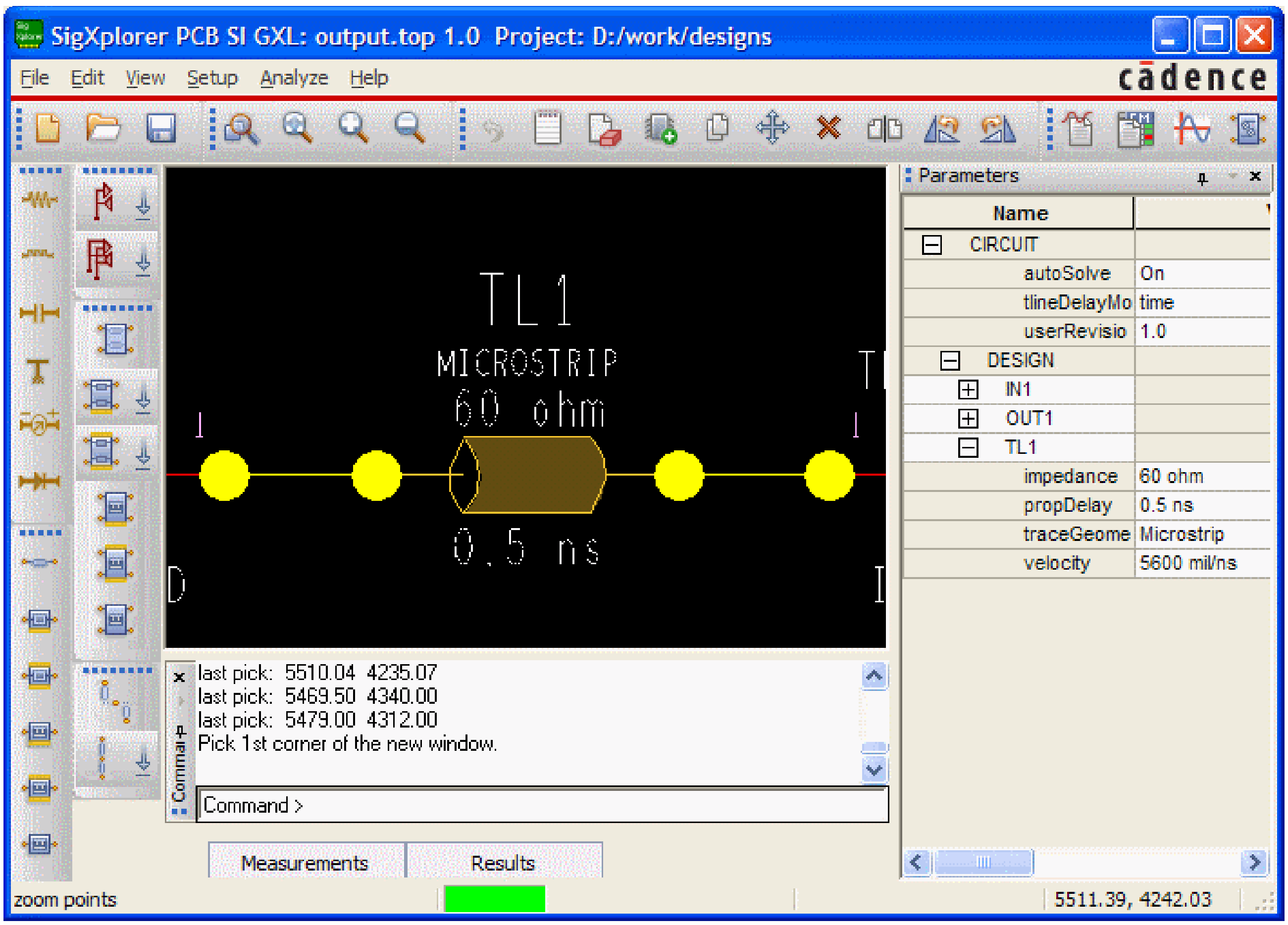
Finding Information about SigXplorer

The SigXplorer documentation set consists of online books accessible from Cadence Help in both HTML and PDF formats. You access documentation from SigXplorer's *Help* menu.

Refer to . . .	**for this level of information**
Allegro SI SigXplorer User Guide (this book)	This book is for users who want to know how to use SigXplorer in the high-speed design flow. It complements the information in the *Allegro SI SigXplorer Command Reference*.
Allegro SI SigXplorer Command Reference	This book contains descriptions and procedures for all commands, organized by menu-sequence. If you click *Help* in a dialog box, or highlight a menu command and press F1, the command description from this book appears. It complements the information in the *Allegro SI SigXplorer User Guide*.
Channel Analysis Introductory Tutorial.	Provides a quick hands-on experience with the Channel Analysis functionality.
Allegro PCB SI User Guide	This book contains reference information about TLsim, the analysis engine used by SigXplorer.

What is SigXplorer?

SigXplorer is an enhanced, SPICE-based simulation tool that aids you (signal integrity engineers) in exploring, identifying, and solving the adverse analog effects of a digital system. You can use SigXplorer to design the latest high-speed computer interfaces, including serial link, common clock, and source-synchronous designs.



SigXplorer provides access to signal models through a common interface to native Device Modeling Language (DML), and seamless access to IBIS, generic SPICE, HSPICE, and Spectre signal models. The Touchstone and Quad signal models are also supported.

The easy-to-use interface of the tool lets you visually:

- construct (or extract) interconnect topology of a circuit,
- set and sweep a range of circuit parameters for devices (buffers, differential pairs, discretes, and vias), characterize them,
- and generate interconnect (actual or ideal).

This enables you to execute *what-if* scenarios on critical high-speed signals in your board, package, or system-in-package design.

You can define parameters for ideal transmission-line models (faster, but less accurate), trace models (slower, but more accurate), vias, and circuit elements that you add to your topology. You can also define IO cell stimulus to drive simulations and specify what to measure.

There are two 2D transmission line solvers, the quasi-static bem2d and the full-wave ems2d. Another solver, FSVia characterizes vias (narrowband, wideband, and s-parameter). The simulation output appears in the *Results* window of the SigXplorer interface, as a comprehensive set of reports, and as a graphical waveform rendition.

You can capture the constraints that you set in a topology file, and then export it to Constraint Manager for importing as an Electrical Constraint Set (ECSet). This ECSet can then be applied to similar nets in your design, such as members of a bus.

Invoking SigXplorer

This section describes the different methods you can use to invoke SigXplorer.

You can invoke SigXplorer (also called Topology Editor) with a clean canvas or by extracting a topology from a layout tool.

Important

The version of SigXplorer that runs depends on the license you have.

Invoking SigXplorer with a Clean Canvas

➡ In Windows,

- ❑ Choose *Start – Run*, and type *sigxp.*

  -or-

- ❑ Choose *Start – Programs – Cadence – Release 16.5 – SigXplorer.*

  -or-

➡ In UNIX, type `sigxp` in a Shell window.

-or-

➡ From a layout tool, choose *Tools – Topology Editor*.

SigXplorer opens with an empty canvas.

Invoking SigXplorer by Extracting a Topology

From Constraint Manager

1. In Constraint Manager, select a worksheet in the Net folder.
2. Right-click on the net of the topology you want to extract and from the pop-up menu, choose *SigXplorer*.

   SigXplorer launches and the topology appears on the SigXplorer Canvas.

From an SI layout tool

1. In your SI layout tool, choose *Analyze – Probe*.

   The *Signal Analysis* dialog box appears.

2. Select the net of the topology you want to extract.

3. Click *View Topology.*

   SigXplorer launches and the topology appears on the SigXplorer Canvas.

Exploring SigXplorer

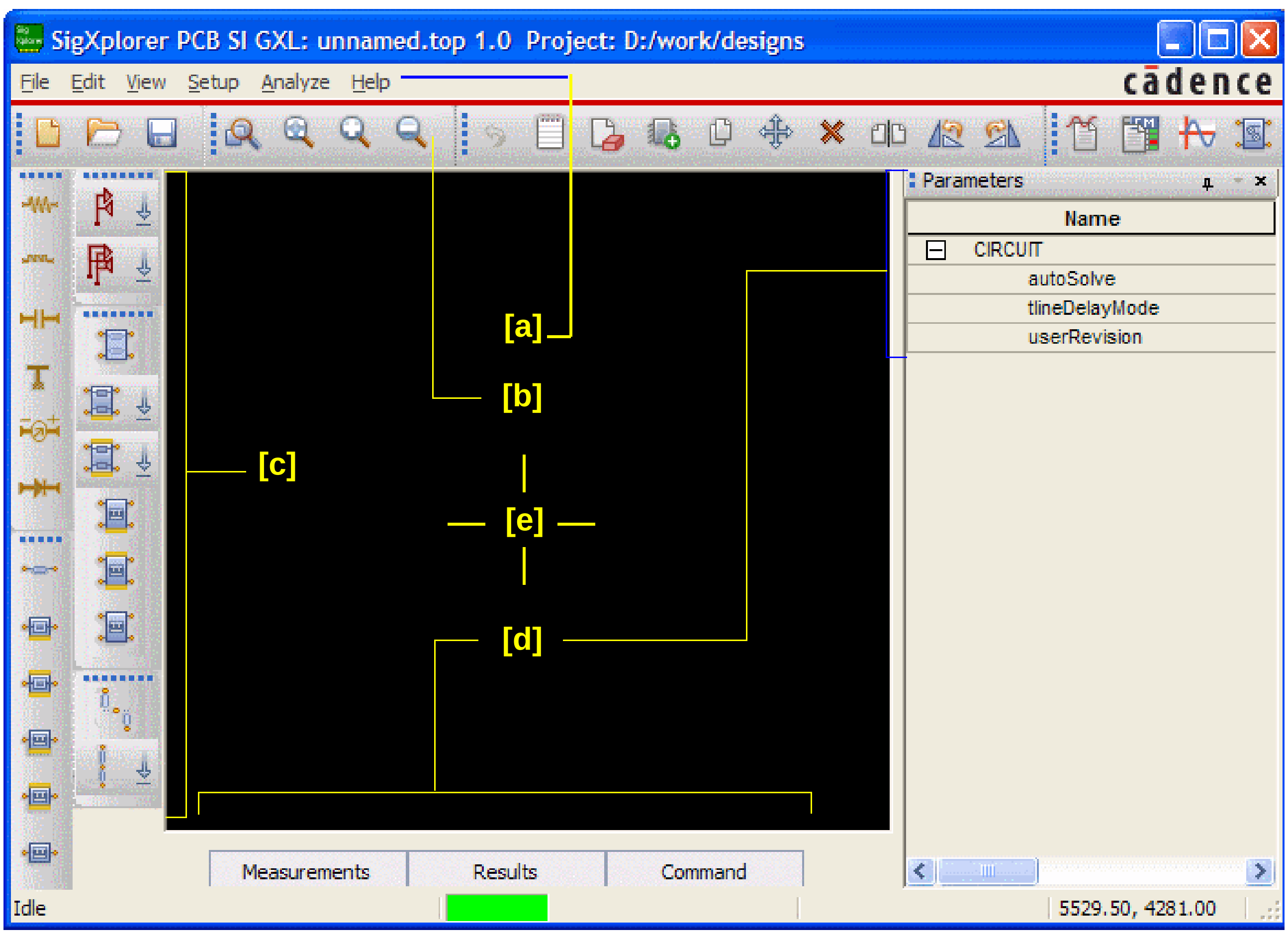
As depicted in Figure 1-1, SigXplorer includes the canvas `[e]`, where you graphically construct your topology, and supporting windows `[d]` for working with *Measurements, Results*, *Commands,* and *Parameters*.

Tip

See "Working with Topologies" on page 25 and "Preparing for Simulations" on page 53 for information on the *Parameters* window.

See"Preparing for Simulations" on page 53 for information on the *Commands*, *Measurements*, and *Results* windows.

Figure 1-1 The SigXplorer Canvas



Command access is provided through menus `[a]` and icons `[b]`. You also have direct access to signal models (discretes, traces, buffers, transmission lines, and vias) through icons `[c]`. Passing the cursor over any of the tabs at the bottom of the screen, *Command*, *Measurements*, and *Results* `[d]`, unfolds the window for viewing or editing. The Parameters window can also be dragged to the bottom as a tab to create more space on the canvas `[e]`.

Measurements Tab

The Measurements tab contains four sections, EMI, Reflection, Crosstalk, and Custom. There is a pull-down menu for each section which includes all standard measurements available for the first three sections. All the currently available custom measurement expressions are available under the Custom section. User-defined custom measurements appear alphabetically, following the standard measurements.

Note: The Measurements tab is grayed out for SiP Layout XL.

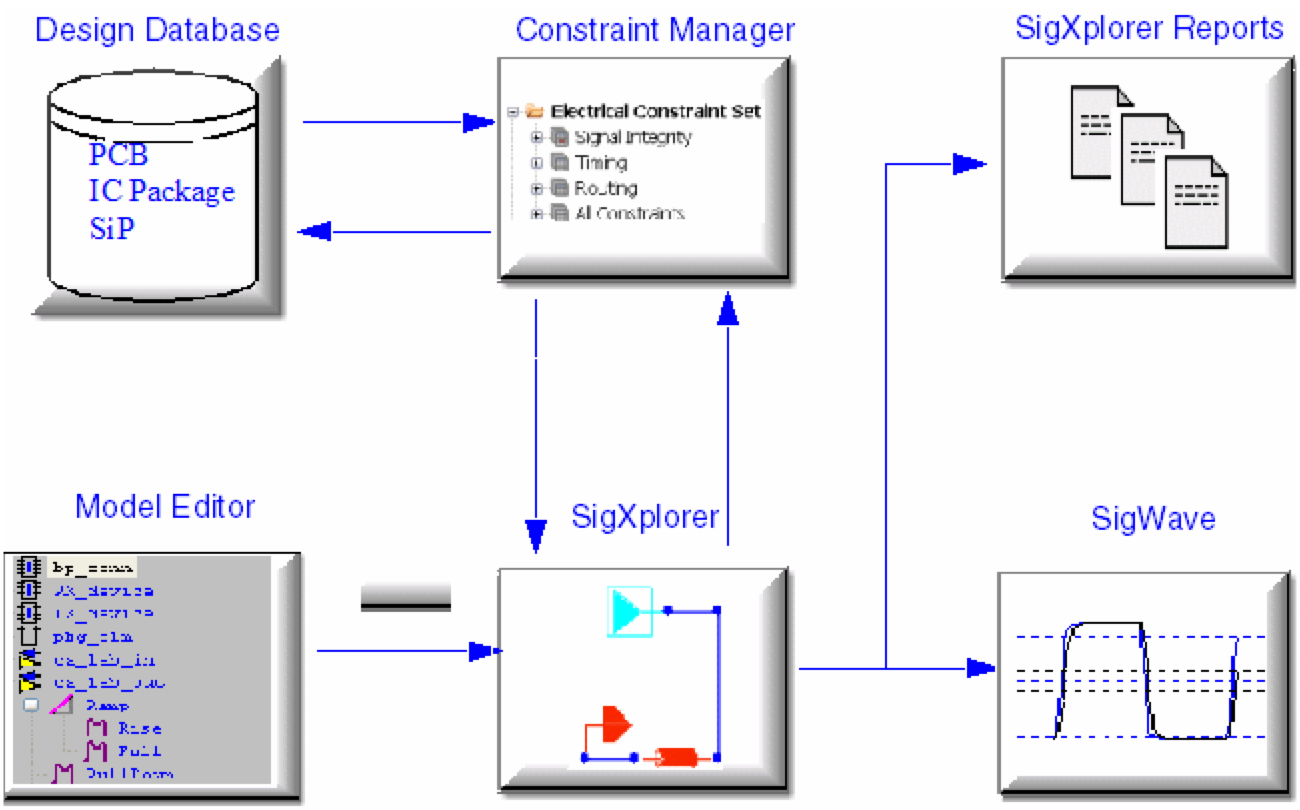
The status bar text at the bottom left of the window displays the active command. The middle of the status bar is colored red during simulation, and green when finished. The rightmost status bar text shows the horizontal and vertical coordinates of your mouse pointer.

SigXplorer makes extensive use of context-sensitive pop-up menu (right-click) for easy access to element parameters and commands. See Editing in Context on page 34 for more information.

SigXplorer Workflows

You can use SigXplorer in the flows as depicted in Figure 1-2 below:

Figure 1-2 The SigXplorer Flow



SigXplorer Flows

■ **Exploration**

In the exploration flow, your focus is on setting up access to libraries, developing signal models, and performing extensive *What-If* analysis. You may not have access to a design database, but you can speculate how a particular component and its interconnect will behave in your topology. You will simulate for *reflection*, *crosstalk*, and *EMI*, derive constraints, and then save them in a topology template file for later reuse.

Tools you use in the Exploration flow include *SigXplorer*, *SigWave*, and *Model Editor*.

■ **Pre-Route Analysis**

In the pre-route analysis flow, your focus is on extracting a signal from a placed component in a PCB, Package, or System-in-Package (SiP) database (fully or partially routed), modifying various components and pin buffers, as well as interconnect, and then setting a range of sweepable parameters to simulate them in TLsim, SigXplorer's native simulator. Depending on the simulated results, you may decide to modify parameters, measurements, or simulation settings, or add a termination scheme. You can also use the imported cross-section, and modify it to see the effects on your topology. You capture constraints in a topology template file and import it to Constraint Manager as an Electrical Constraint Set (ECSet) to refresh the design.

Note: The SigXplorer environment also supports the HSPICE and Spectre (Unix only) simulation engines.

Tools you use in the pre-route analysis flow include *SigXplorer*, *SigWave*, *Model Editor*, *Constraint Manager*, and a PCB, Package, or SiP layout tool.

- **Post-Route Verification**

  In the post-route verification flow, your focus is on extracting a signal from a PCB, Package, or SiP database (fully routed), setting sweepable parameters, and simulating the topology. You use the *Results* window of SigXplorer, built-in reports, and SigWave to verify that the integrity of the signal meets your requirements.

  Tools you use in the post-route verification flow include *SigXplorer* and *SigWave*.

Changing Your View of the Canvas and Windows

The easiest way to zoom in and out, and move (roam or pan) across the canvas, is using the middle mouse button.

You can pan a topology (move across a topology in the canvas) to view different parts in it. To pan a topology, you need to hold the cursor inside the canvas, and then click and hold the middle mouse button as you drag the cursor across the topology. As long as the mouse button remains pressed, you can move all areas of the topology into full view.

Tip

Use the arrow keys on your keyboard to pan in the desired direction.

To zoom in or out, rotate the scroll wheel of the mouse. Table 1-1 on page 19 displays the SigXplorer icons and keyboard shortcuts you can use to perform various zoom functions.

Table 1-1 Zoom Functions

Choose *View* –	Shortcut	Function
Zoom By Window		Magnifies the display so that the specified, bounded, region fills the canvas.
Zoom Fit	 `F2`	Changes the display so that the topology fills the canvas
Zoom Center		Changes the display so that the topology is left justified and centered vertically within the canvas
Zoom In	 `F11`	Magnifies the topology to make it larger and display less of it in the canvas.
Zoom Out	 `F12`	Shrinks the topology to display more of it in the canvas

Choose *View* –	Shortcut	Function
Zoom Previous	`SHIFT+F11`	Displays the previous zoomed *view* of the topology on the canvas

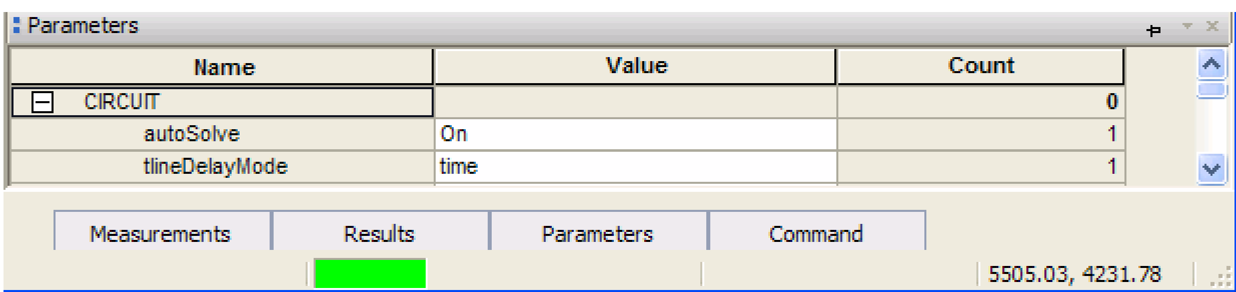
Re-sizing the Canvas

You can move the border between the spreadsheet window and the canvas to increase or decrease the view. You need to drag the edge of the border of the spreadsheet with your mouse, and then move the divider left or right, as required.

Working with Foldable Windows

The foldable windows are particularly useful on a single monitor setup because they provide more work space, while giving you the option of seeing the window information by hovering your mouse over their respective tabs: *Command*, *Measurements*, *Results*.

Passing the cursor over any of the tabs unfolds the window for viewing or editing. As you move the cursor off of the tab, the corresponding window retracts. By default, the *Parameters* window appears on the right of the application window. To create more workspace, you can drag it to the bottom of the screen. When you pin the window, it retracts to the bottom as a tab. As you hover the mouse pointer over each tab, the corresponding window appears.



Persistent Windows

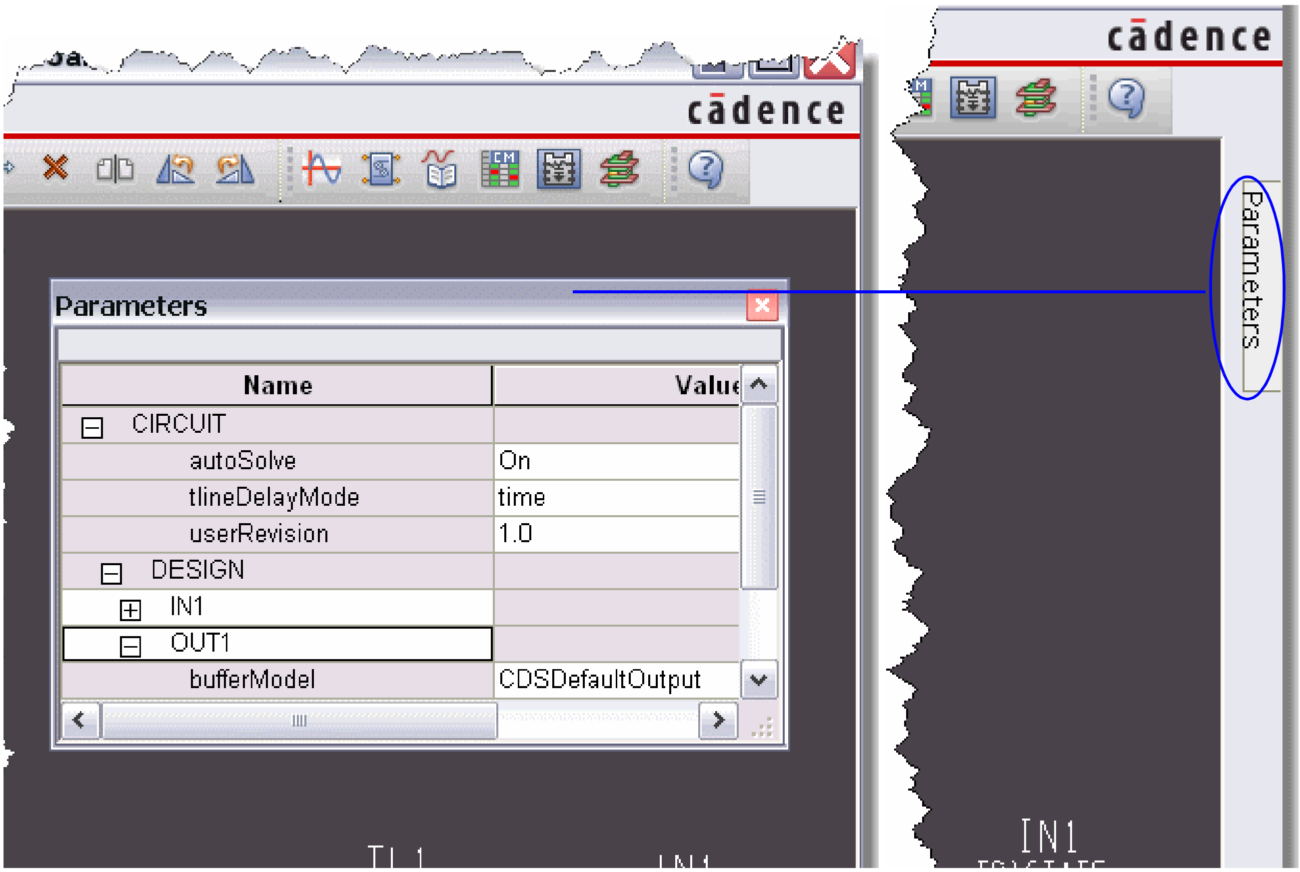
Rather than having a window retract when you move your cursor away from it, you can make the visibility of the window persistent by passing your cursor over a tab, and then clicking the pin.

You can click the pin again (unpin) to make it retractable, or click `X` to close a pinned window.

Note: If you close a window once, you must choose *View – Window – [Window Name]* to restore its visibility.

Undocking Windows

If you have pinned a window (see Persistent Windows on page 21), you can relocate it by dragging it by its title border anywhere on your desktop. You can unpin the window to make it retractable.



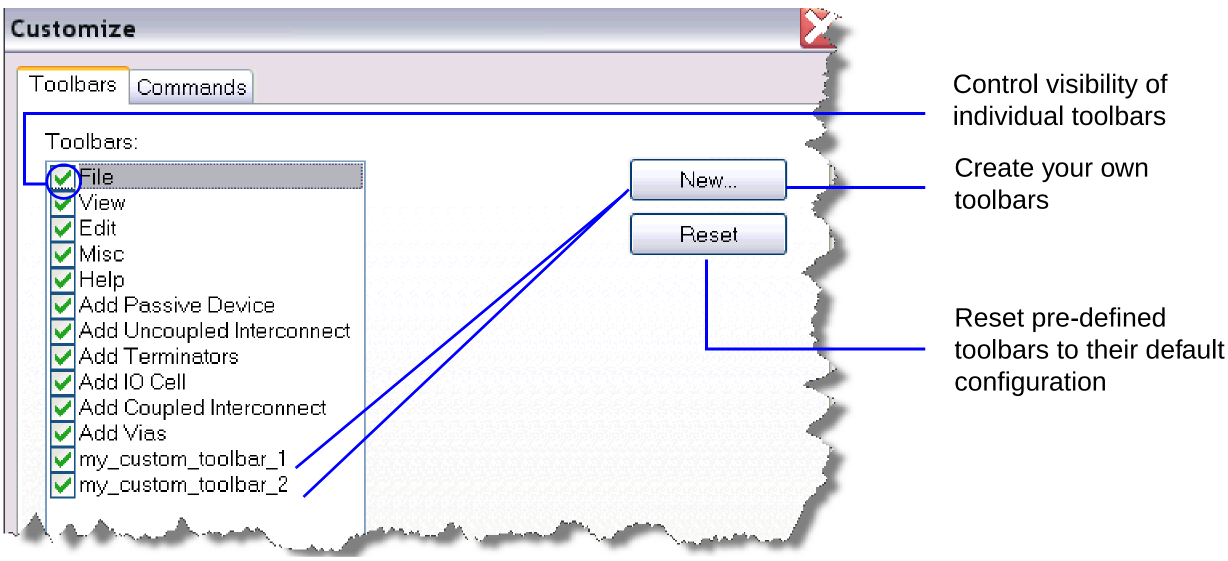
Tip

To restore windows to their default locations (*Command, Measurements, Results* at the bottom of the canvas and *Parameters* to the right), choose *View – Reset UI to Cadence Default*.

Working with Toolbars

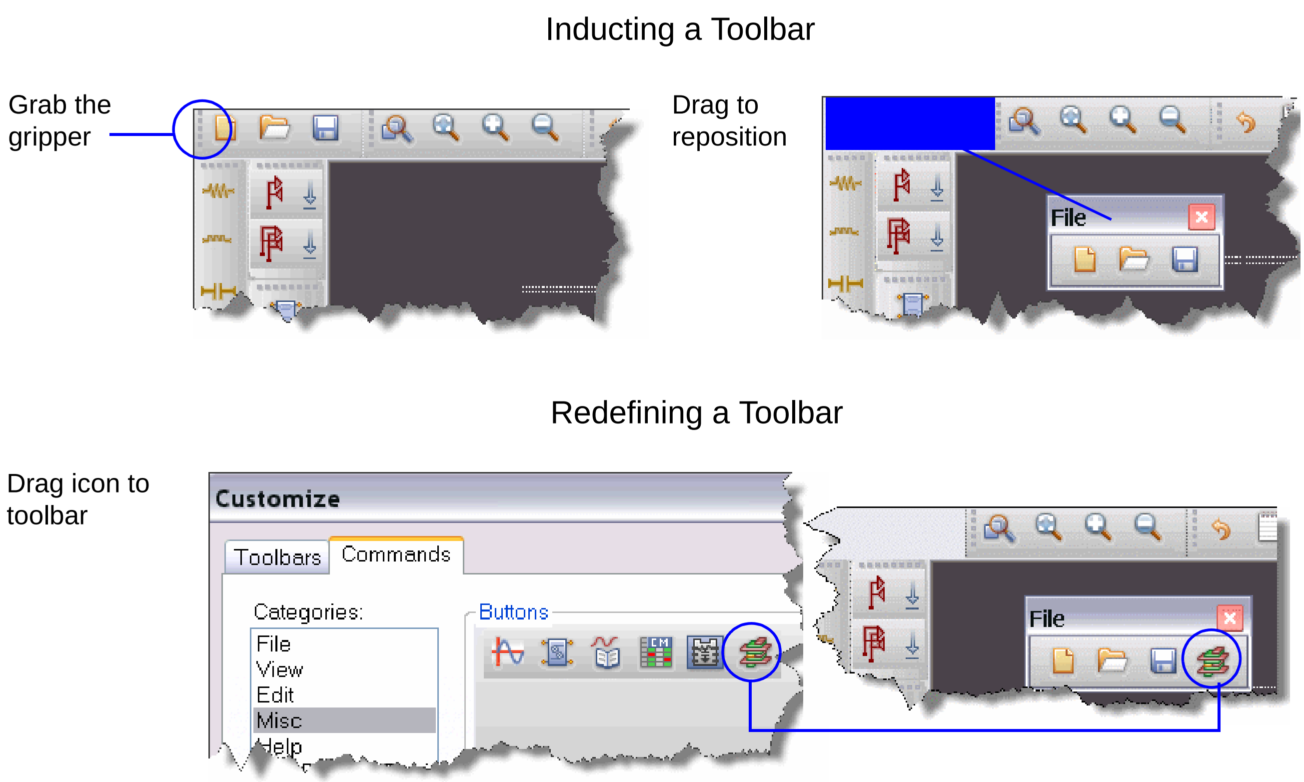
SigXplorer contains many icons for quick access to commands. Icons are logically organized by function into toolbar groups. You can selectively show or hide toolbar groups (see Figure 1-3).

Figure 1-3 Customize Dialog Box (Toolbars tab)



Toolbars can also be repositioned (undocked) within or outside the canvas, anywhere in your workspace, independent of other toolbar groups. You can also change the member icons of a pre-defined toolbar group by adding or removing icons. For maximum flexibility, you can create your own custom toolbar groups (see Figure 1-4 on page 24).

Figure 1-4 Customize Dialog Box (Commands tab)



Tip

To learn the name of a toolbar group, undock it.

With the *Commands* tab active, as shown in Figure 1-4, you can drag any icon from a pre-defined toolbar (that surrounds the canvas) to remove the icon from that toolbar group. To restore a deleted icon to its pre-defined toolbar, choose *View – Reset UI to Cadence Default*.

2

Working with Topologies

Topics in this chapter include:

Creating a Topology

There are three use models for working with topologies in SigXplorer:

- Create a new circuit topology from scratch
- Open an existing topology saved from an earlier session
- Extract a topology from a PCB, IC Package, or SiP database (see Extracting a Topology on page 51)

Creating a Topology from Scratch

You create a topology (in a Constraint-driven flow) by:

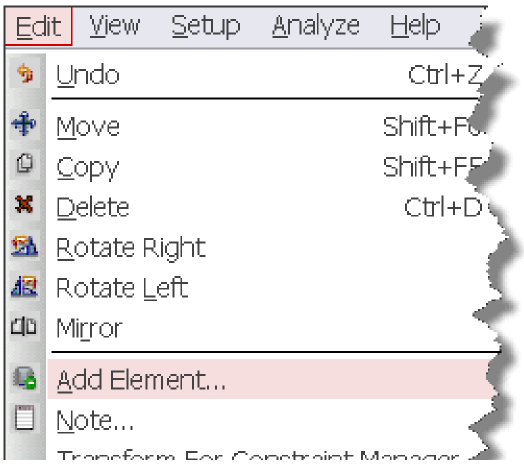
1. Adding Elements or models from the Cadence default libraries or from user-defined libraries.
2. Placing each element in the SigXplorer canvas.
3. Wiring the elements.

Using SigXplorer, you capture the net schedule (see Scheduling a Topology on page 43), impedance, delay, and termination of a net, and save it to a topology template. A single topology template can control an entire bus. You package the constraints as an electrical CSet (ECSet), which applies to every net. This eliminates the need to create a topology for each net of the bus.

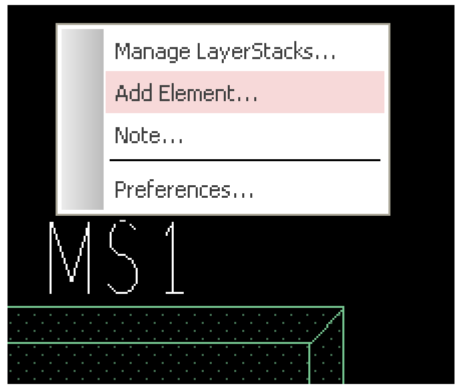
Adding Elements

You add elements (models) to the topology using the Add Element Browser window. To display this window, use one of the following three methods:

Menu Access

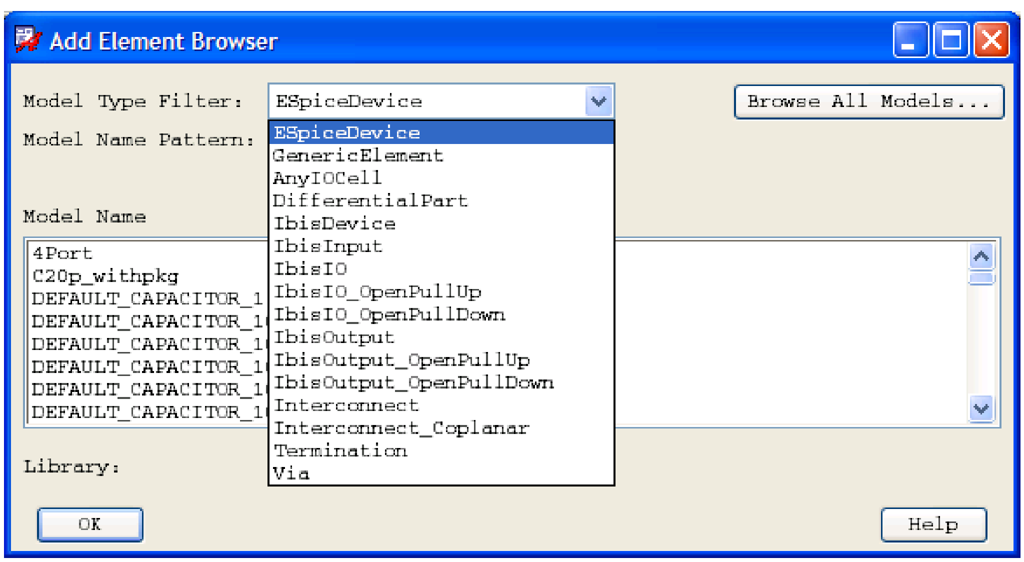


RMB Menu Access



Icon Access

Figure 2-1 Add Element Browser



Placing Elements

When you select a model in the Model Name list, the corresponding symbol for the model is attached to the end of your cursor as you drag it to the canvas. The element is instantiated with each click that you make on the canvas. You must right-click and choose *End Add* from the pop-up menu to stop instantiating the selected element on the canvas.

Wiring Elements

You wire or connect elements on a canvas by selecting yellow dots on each part. To continue a connection, you need to double-click on the wire stub.

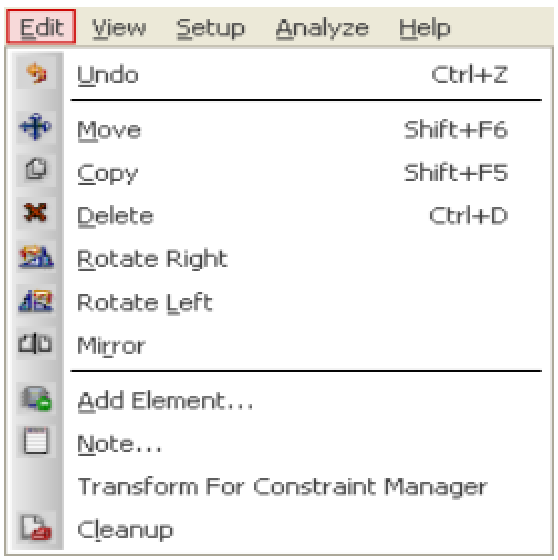
You can edit the topology using any of the editing features available in the SigXplorer interface. For more information, see the *SigXplorer Command Reference*.

Common Editing Operations

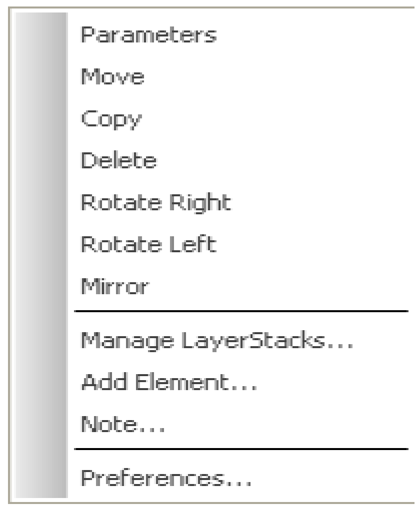
As you develop your topology, you need to be aware of SigXplorer's useful editing techniques, such as moving, copying, deleting, rotating, and mirroring elements on the canvas. Rotating and mirroring parts can improve the readability of topologies or fix issues where connections are crossed.

Along with keyboard shortcuts, and icons, there are two menus, which help you perform editing operations on a selected element on the canvas.

Edit Menu



Context Menu (right-click)



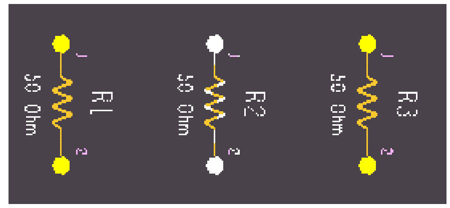
Selecting Elements

To perform an edit operation on elements (move, copy, delete, rotate, or mirror), you must first select them on the canvas.

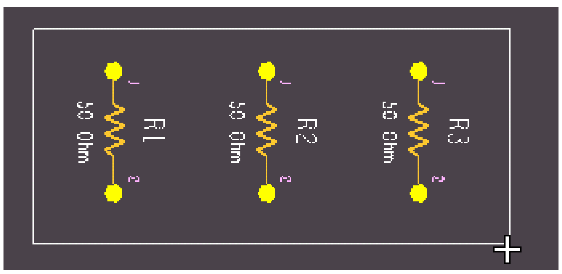
Note: You can do `Ctrl+click` to add to, or remove from, a group of selected elements.

Single-element Selection

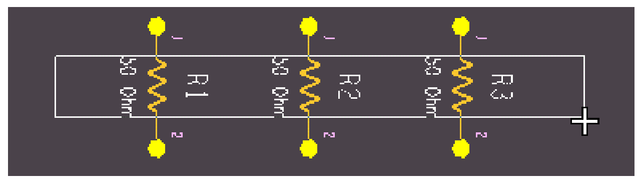
Click an element to select it. The middle resistor is selected, as indicated by the change in color.



Multiple-element Selection (enveloping)



Multiple-element Selection (swipe)



Moving, Copying, and Deleting Elements

To move an element:

1. Click the element.
2. Drag and drop the element to anchor it elsewhere on the canvas in a single operation.

To move multiple elements:

1. Click the first element.
2. Keeping the Ctrl key pressed click the other elements.
3. Once you have selected the elements you want to move, drag them to the new location.

To copy an element:

1. Select the element.
2. Keeping the *Ctrl* key pressed move the element.
3. Click to anchor the element elsewhere on the canvas.

To copy multiple elements, after you have selected the elements, click the location where you want to copy the elements. Ensure that the *Ctrl* key is pressed all the while.

To delete an element:

1. Select the element.
2. Choose *Delete* from the *Edit* menu OR right-click the element and choose *Delete* from the pop-up menu

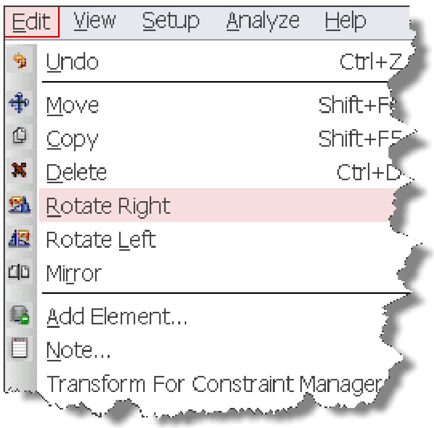
Tip

You can also press the *Ctrl—D* key combination to delete a selected element on the canvas. To delete multiple elements, select the elements keeping the Ctrl key pressed and then press the *Ctrl—D* key combination.

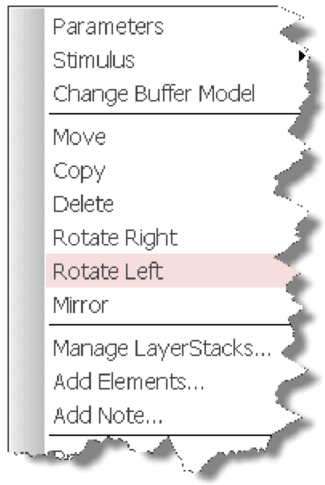
Rotating Elements

SigXplorer can rotate an element on the canvas 90^0 clockwise, or counter-clockwise. If an element has been rotated in one direction, it can only be rotated in the opposite direction. Therefore, an element cannot be rotated upside down.

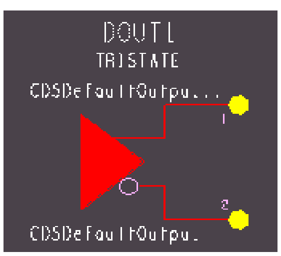
Menu Access

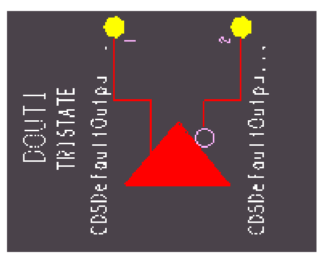


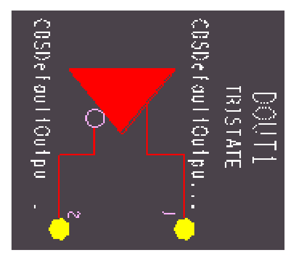
RMB Access



Icon Access

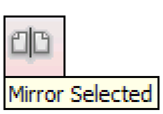





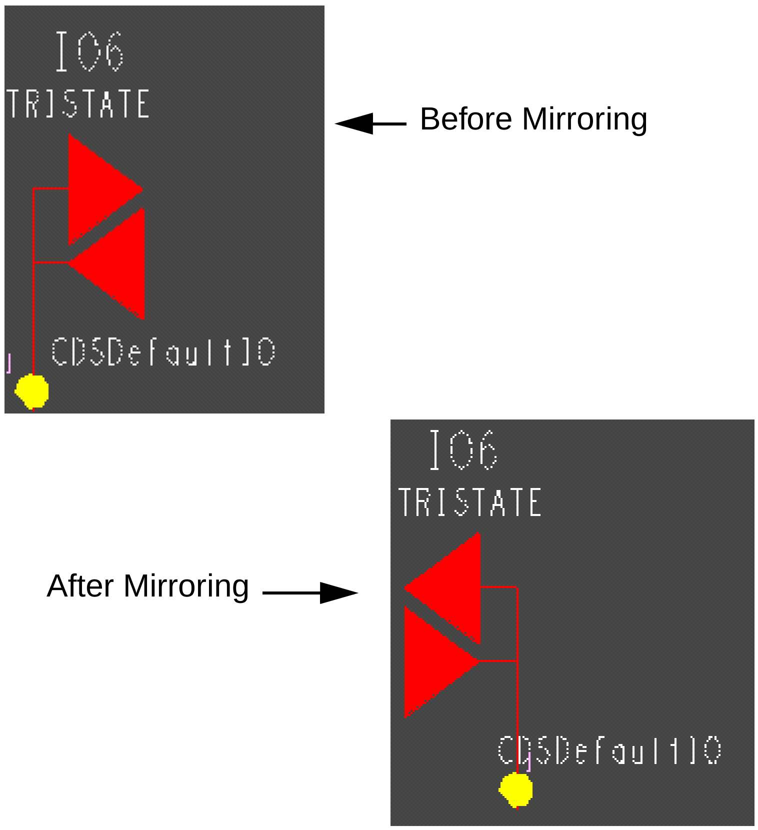


Mirroring Elements

To mirror an element, select it and choose *Mirror* from the *Edit* menu or the context-menu. You can also click the Mirror Selected icon on the toolbar after selecting the element on the canvas.



For mirroring multiple elements, you must first use one of the multiple-element selection techniques.



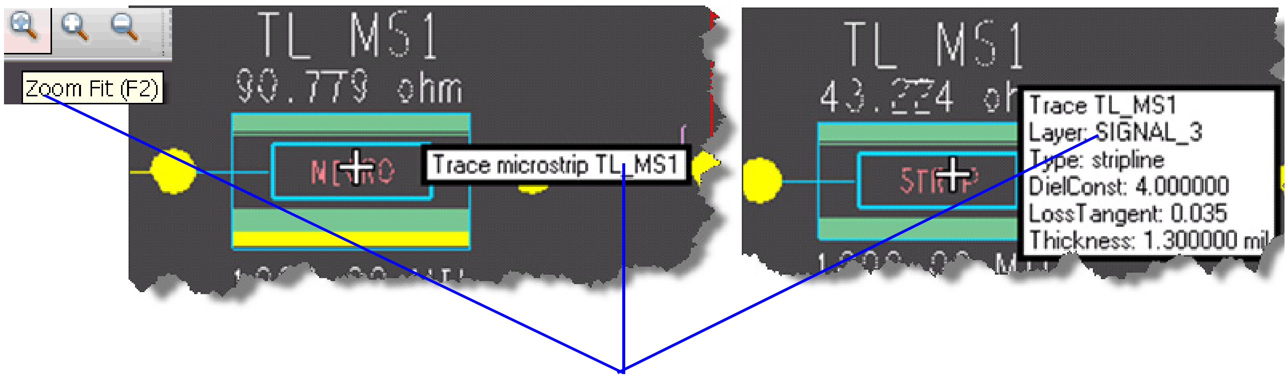
Capturing Canvas Images

You can create images of the canvas for pasting in documents or graphic editors. To capture an image of the canvas, do the following:

1. Choose *File – Capture Canvas Image*.
2. Open the target application, such as Microsoft Word or MS Paint and paste the captured image by choosing *Edit – Paste* or *Paste* from the RMB menu.

What Are DataTips?

SigXplorer uses datatips or tooltips to reveal information about icons and elements on the canvas. A datatip can be as concise as the brief usage tip that appears when you hover your cursor over an icon, or as detailed as parameters set on a trace model.



DataTips

Tip

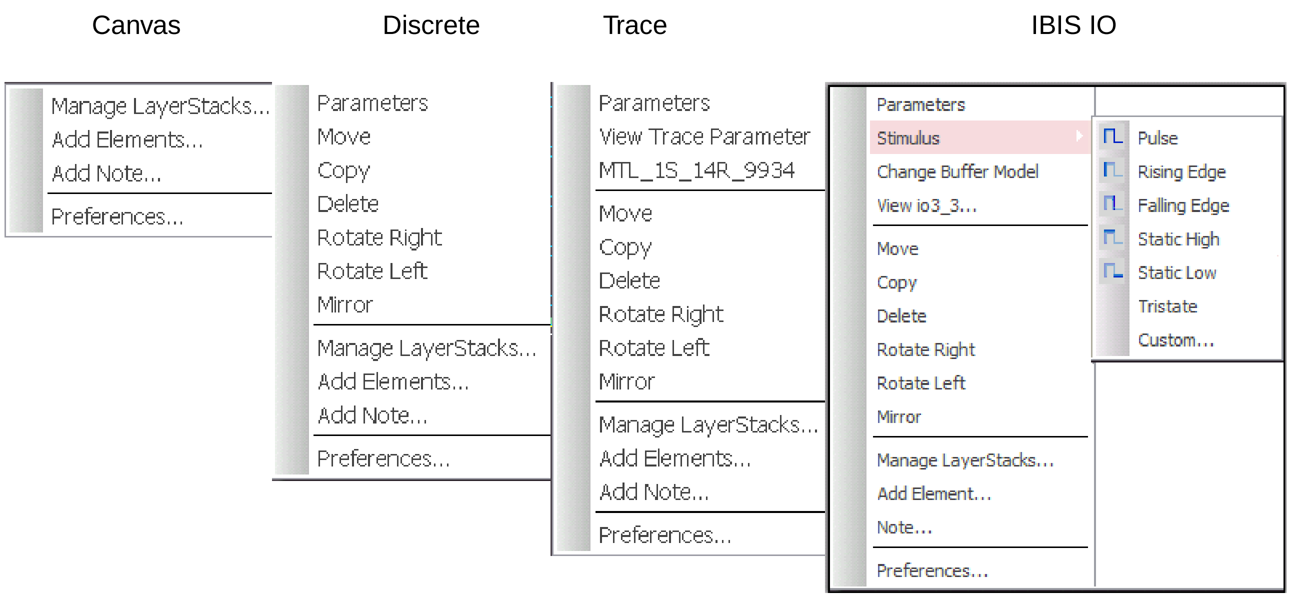
Briefly hover your mouse cursor over an icon to display a datatip describing the purpose of the icon.

Editing in Context

SigXplorer supports right-click access to commands in the context or pop-up menu. Commands in this pop-up menu vary in context, depending on what is selected—an element on the canvas, or the canvas itself.)

Notice that the *Preferences* command is listed last and the *Parameters* command is listed first (except when you right-click on the canvas). General editing commands are clustered together and element-specific commands are listed at the top (see Context-Menu Commands (based on selection) on page 35).

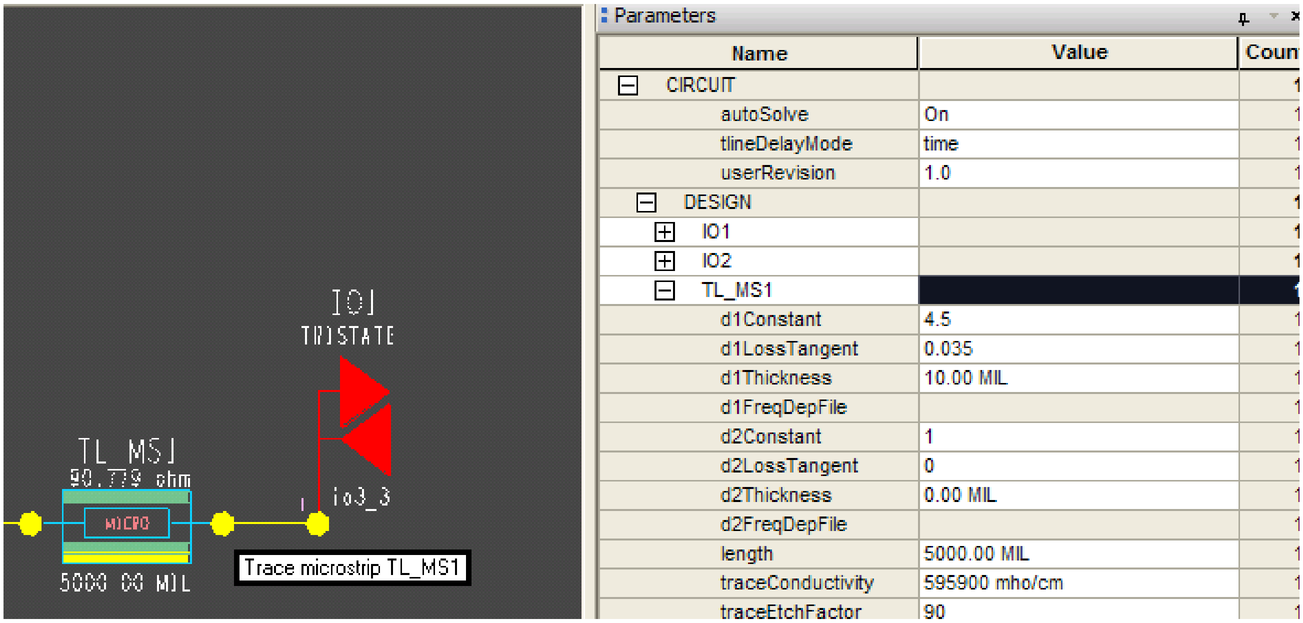
Figure 2-2 Context-Menu Commands (based on selection)



Modifying Parameters for Topology Elements

In SigXplorer, circuit data and element parameters (and their values) are maintained in the *Parameters* window. When you place an element on the topology, you see both the symbol and associated text fields that describe default parameters and other information associated with the element (see Figure 2-3). After simulating, you edit this data to further refine the circuit design.

Figure 2-3 A Placed Element on the Canvas (Parameters Window Undocked)



You can modify the following parameter information:

- Name (Reference Designator or RefDes)

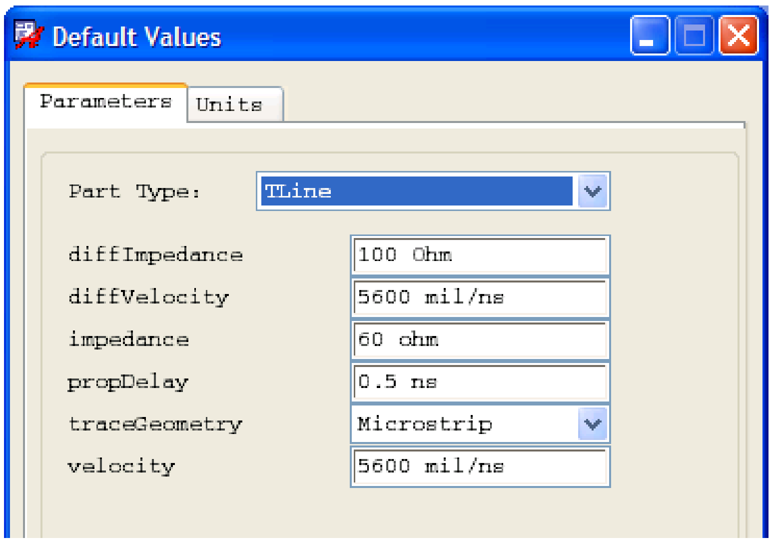
  If you click the label `TL_MS1`, for example to select the RefDes, the Parameters window opens to display all the parameters associated with the element and the value for each. The selected RefDes parameter is highlighted and ready for editing. After you complete the editing, the updated data is reflected in both the *Parameters* window and the canvas.

- Value

  To highlight the associated row for an element in the *Parameters* window, click the value of the element in the canvas and specify a new parameter value. After you complete the editing, the updated data is reflected in both the *Parameters* window and the canvas.

Default Values for Parameters

When you place an element on the canvas, a default set of parameter values are assigned to the element. You can modify these default values for component models, interconnect models, and terminator models, using the Default Values dialog box (*Setup – Defaults*). For more information, see *SigXplorer Reference*.



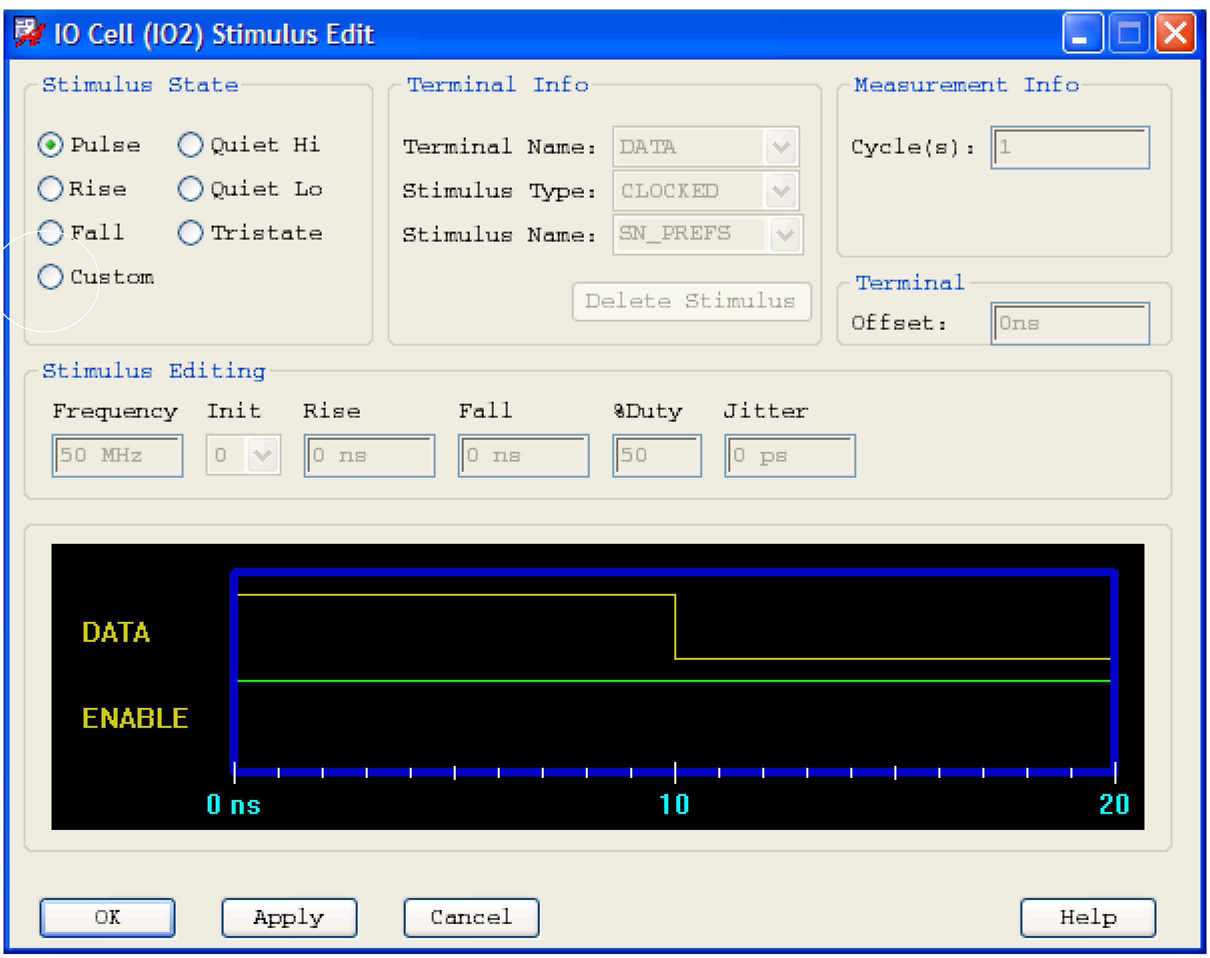
Note: When you modify a default value, it becomes the new value of the element whenever you place the symbol on the canvas.

Using the *Default Values* dialog box, you can:

- Select any of the generic component models, interconnect models, and termination models available in the *Model Browser*.
- Display the parameters for the element and the default value associated with each parameter.
- Modify the default parameter values.

IOCell Stimulus Parameters

The *Parameters* window does not contain the stimulus data for IOCell parts. You use the *IOCell Stimulus Editor* to modify this information. See <u>*SigXplorer Reference*</u> for more information.

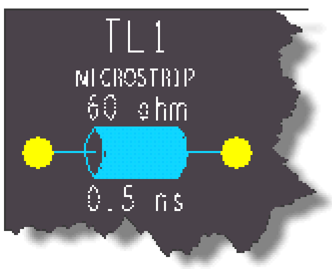


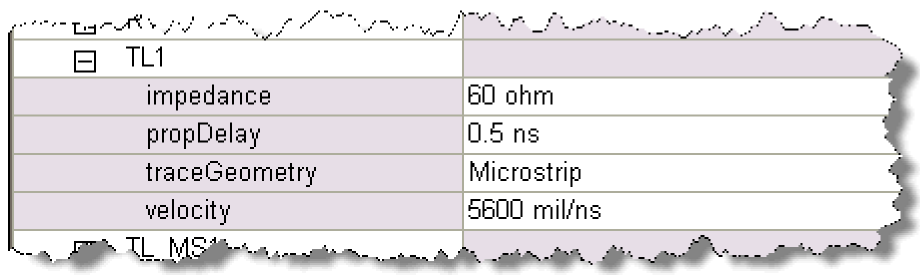
Tip

You can access the IO Cell Stimulus Editor by choosing *Stimulus* from the pop-up menu when you right-click on the IO component.

Wiring the Topology

An ideal transmission line (TL or Tline) element in SigXplorer is characterized by *impedance*, *propDelay*, *traceGeometry*, and *velocity* values, which you can view in the *Parameters* window. Of these four, the *fundamental* parameters are *impedance* and *velocity*.





The *propDelay* parameter, as an indication of trace length, is defined in nanoseconds, while *velocity* is defined in mils-per-nanoseconds.

The traceGeometry parameter does not affect simulations, but allows you to set default velocity values to any microstrip or stripline transmission lines you add to your topology. Knowing the parameters of *impedance* and *velocity*/*propDelay*, you can infer L and C per unit length to define an ideal (loss less) transmission line.

Advantages of using a lossless transmission lines in pre-layout simulations are faster simulation times when compared to MS/SL, and slower absorbtion rate of reflection, particularly when simulating very long trace lengths for a termination scheme. Once you have established a satisfactory base, you may substitute the lossless transmission lines for microstrip or stripline models.

To wire a topology, click the pin of a topology element (see Figure 2-4 on page 41). As you move the cursor, a wire moves with the cursor away from the pin. Click the destination pin on another element. A wire is drawn between the elements. Click another pin to extend the wire.

Note: To delete a wire from the topology, click on the wire anywhere, except where it meets the pin of an element.

SigXplorer provides the following two methods for wiring a topology:

- Pin-to-Pin Method as shown in Figure 2-4 on page 41
- Pin-over-Pin Method as shown in Figure 2-5 on page 42

Figure 2-4 Wiring a Topology: Pin-To-Pin Method

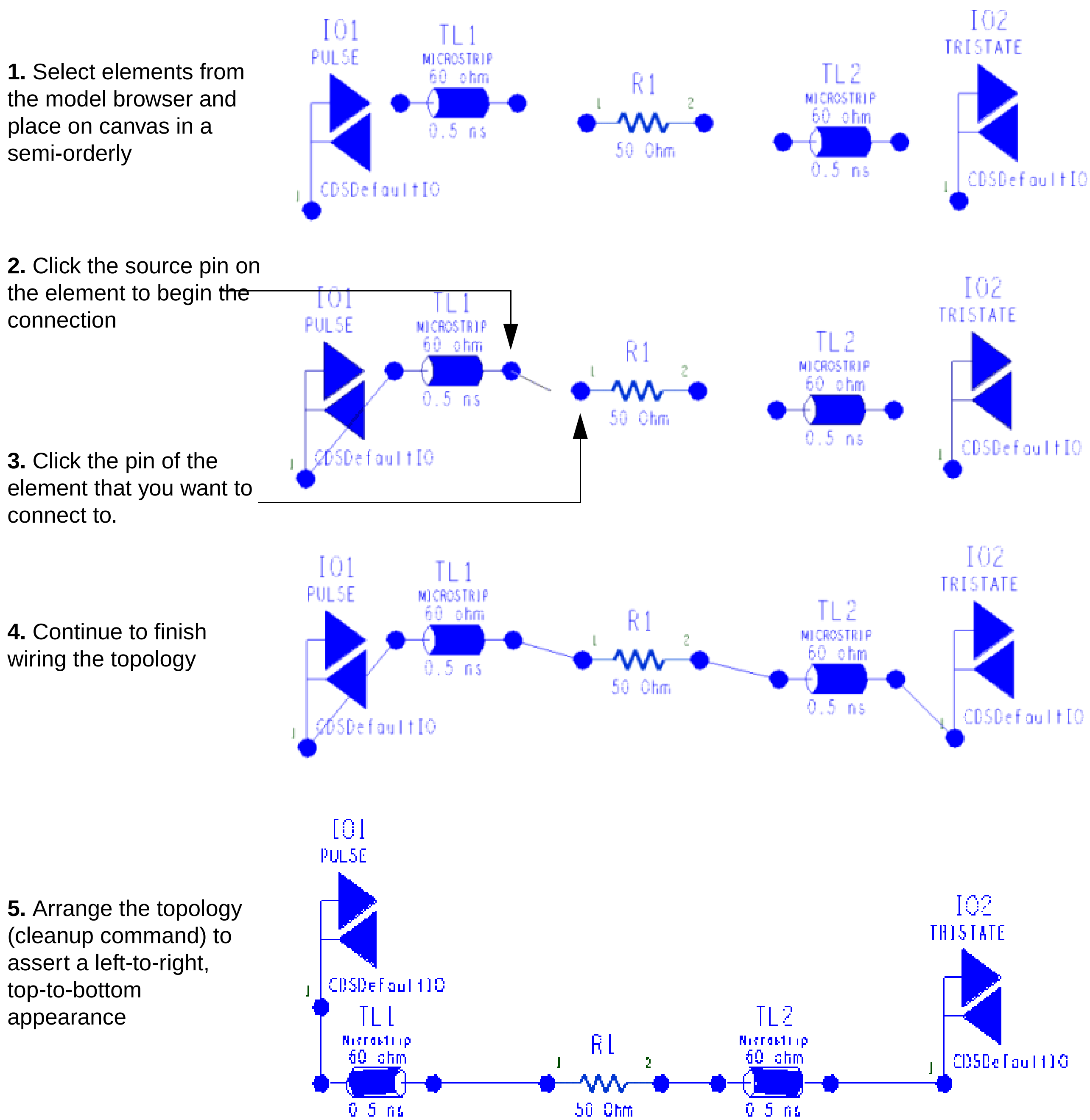
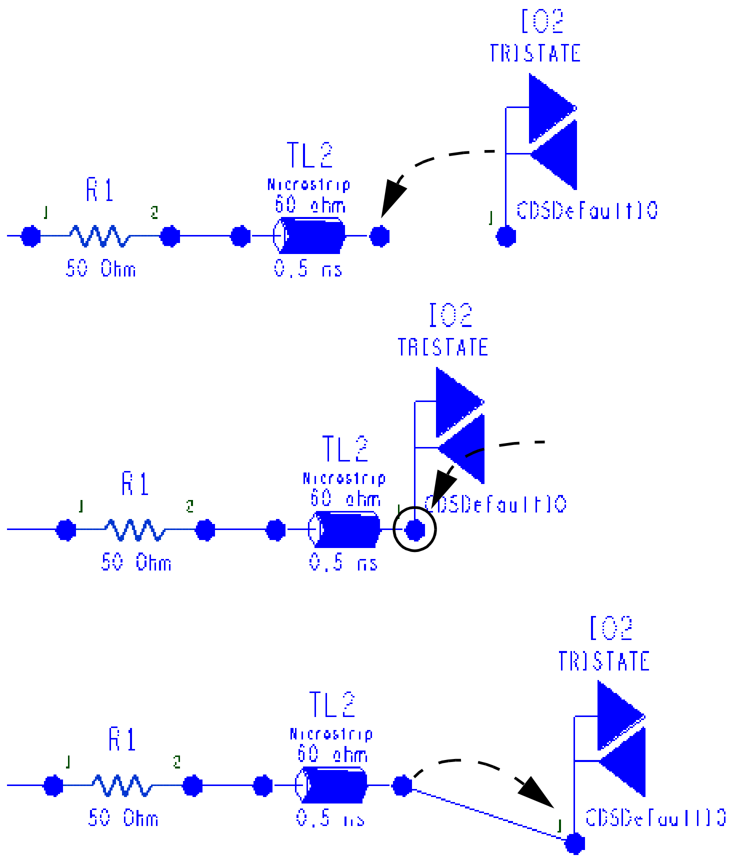


Figure 2-5 Wiring a Topology: Pin-Over-Pin Method



1. To connect the pin of the transmission line to the pin of the buffer ...

... drag the pin of the transmission line over the pin of the buffer.

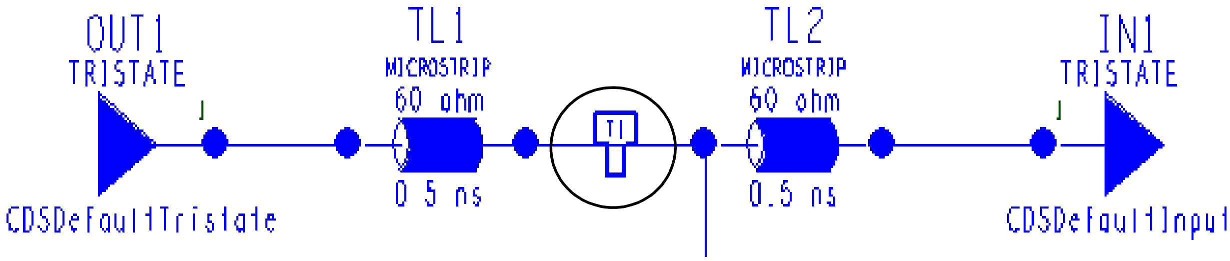
2. Then, drag the buffer away from the transmission line.

When you drop the element, observe that the connection is now made.

T-point Elements

SigXplorer automatically places a T-point element at the junction of two or more transmission lines (TLines) with no other pins on the node, as shown in Figure 2-6 on page 43. T-points represent pin-to-tee and tee-to-tee constraints. Symbols visually indicate T-point instances in the topology. You select these references in the *Set Topology Constraints* dialog box to specify TLine (prop delay, impedance, and relative prop delay) constraints.

Figure 2-6 T-Point Elements in SigXplorer



Scheduling a Topology

There are two methods of scheduling a topology. You can wire the topology interactively and create a template schedule or automatically schedule the topology by selecting from a set of generic templates.

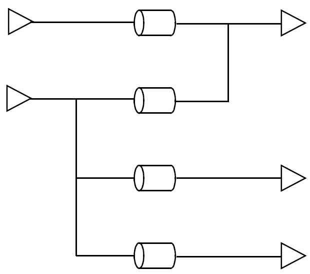
Using a generic schedule is advantageous, because it is a fast way to create a topology. In this method, you first place the IOCells on the canvas, and then select a schedule. All of the necessary TLines immediately add and connect to the IOCell pins. If you decide not to select a generic schedule, you need to add and connect each TLine to form the net schedule.

When you extract a topology template with a generic schedule into Constraint Manager, the electrical constraint set stores the specified type of schedule as a *ratsnest schedule* constraint in the constraint set. When you assign the constraint to a net, the ratsnest schedule constraint performs the specified type of ratsnesting for that net.

The ratsnest schedule of pins of the net, assigned by the constraint set, is not necessarily the same as the order of the pins as seen in SigXplorer. The ratsnest schedule depends upon the placement of the pins. The topology, simulated in SigXplorer, might not be the same as the topology assigned to the net on the board. To ensure that the net follows the schedule, apply a generic schedule, and then interactively edit any of the TLine connections, as necessary.This changes the schedule type from the selected generic schedule to a template schedule.

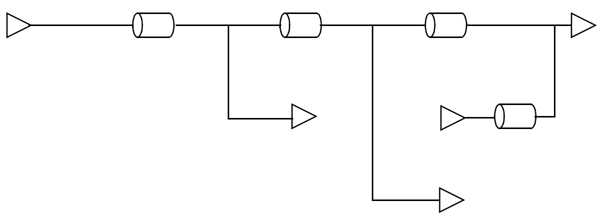
The following are the schedule types available in SigXplorer:

Min Spanning Tree



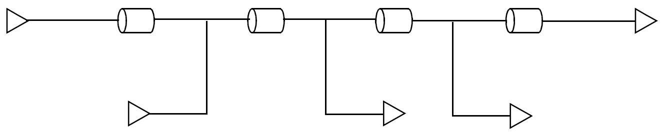
Connects all pins together with minimum connection lengths. Any pin can connect to *any number* of other pins. This schedule starts at the primary driver and selects the closest pin to this driver and connects it to the driver through a TLine. The search continues by selecting the next unscheduled pin that is closest to any of the scheduled pins and connecting it with a TLine to the pin to which it was closest. This continues until all pins are connected.

Daisy Chain



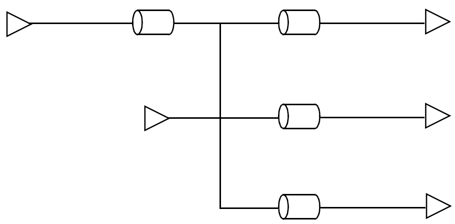
Connects pins of the topology with minimum connection lengths, only allowing each pin to connect to a maximum of *two* other pins. This schedule starts with the primary driver, and then the closest pin to this driver is connected with a TLine. The closest pin to the last pin scheduled is connected with a TLine. This continues until all pins are connected.

Source Load Daisy Chain



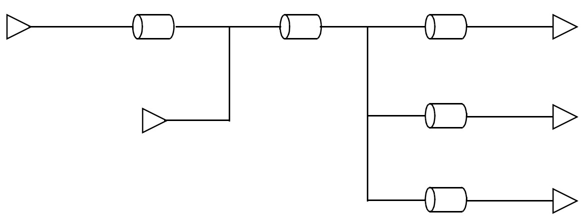
Similar to a daisy chain schedule, except that all driver pins connect first, followed by all receiver pins.

Star



Connects driver pins, which are daisy-chained together, and then all of the receiver pins connect to the last driver pin.

Far End Cluster



Similar to a star schedule, except that the last driver pin connects to a T-point to which all of the receivers connect.

Rules for Generic Topology Schedules

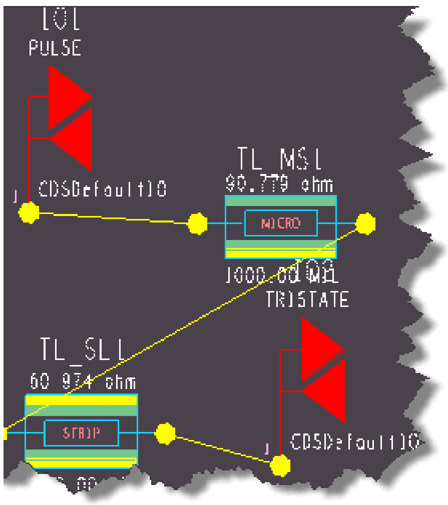
The following are general rules to follow for any type of generic schedule:

- When automatically scheduling the pins in a topology, the only restriction on allowable elements is when the topology defines a differential pair. In this case, the inverting and non-inverting signals must contain the same number and types of pins.
- A message prompts for permission to delete all unmatched pins. All the other types of elements are scheduled.
- The proximity of the pins to each other determines the scheduling.
- All types of scheduling start with sequencing the pins with the primary driver.

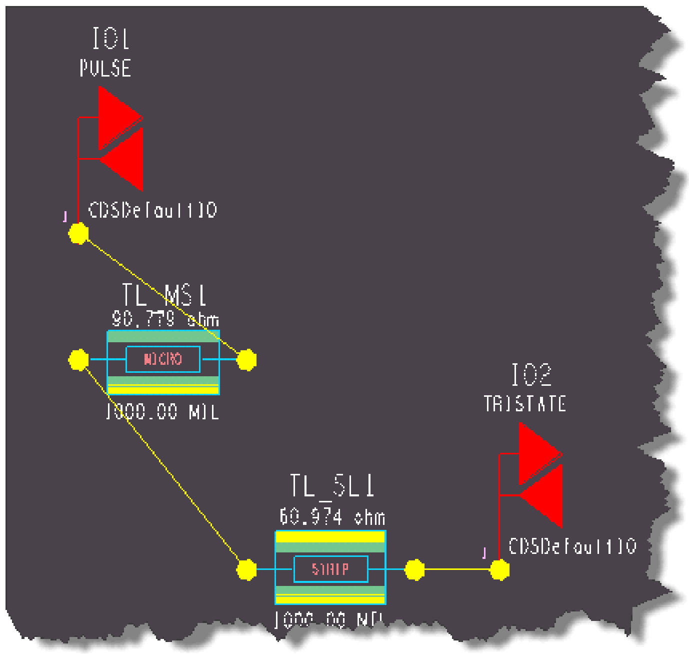
- ❑ If the topology only contains one driver IOCell, it is considered the primary driver.
- ❑ If there are multiple drivers, the active one is selected.
- ❑ If there are multiple active drivers, SigXplorer chooses one of the active drivers.
- ❑ If there are no active drivers, SigXplorer chooses one of the non-active drivers.
- ❑ If there are no drivers, but there is a non-active bidirectional pin, SigXplorer chooses the bidirectional pin.
- ❑ If a topology contains no drivers or bi-directional pins, then a warning appears, and the schedule type resets to template and no changes occur.

- Any terminator, correctly connected to a pin, remains intact during scheduling.
- After you assign a schedule to a topology, use the cleanup command to redraw the topology on the canvas (*Edit – Cleanup*).

Before Cleanup



After Cleanup



Managing LayerStacks

SigXplorer supports multiple layers, which you can create from scratch, or import from a PCB, IC Package, or System-in-Package (SiP) database. Managing LayerStacks includes:

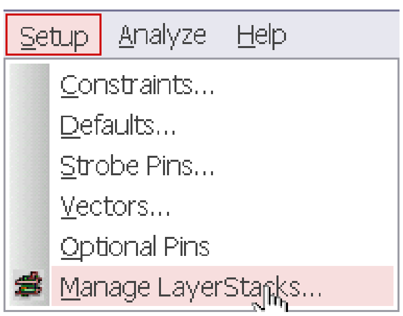
- Moving an element, such as a trace model, among different layers
- Modifying trace width or length
- Experimenting with dielectric constant, loss tangent, materials, and layer thickness values to solve for optimum impedance

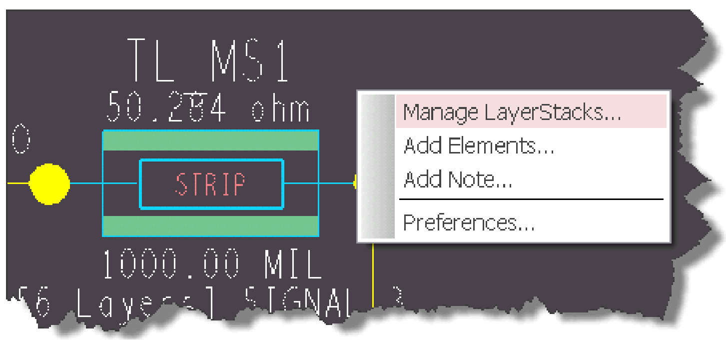
Note: You can export LayerStacks that you create to a technology file (`.tcf`).

To create a new LayerStack:

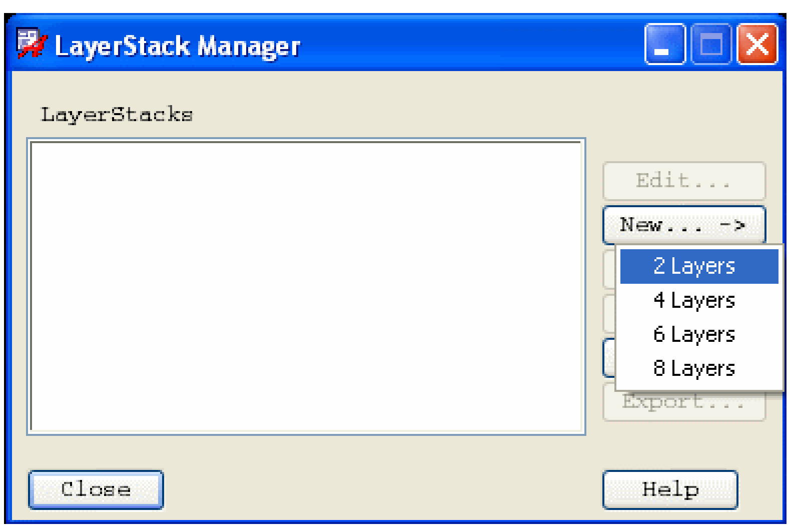
1. Launch the LayerStack Manager dialog box using any one of the following methods.

Menu Access **RMB Access** **Icon Access**



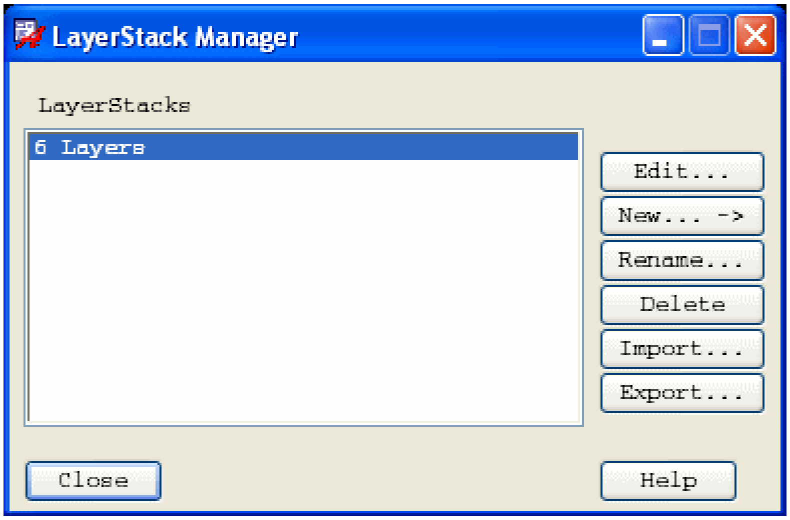


2. Choose *New.*



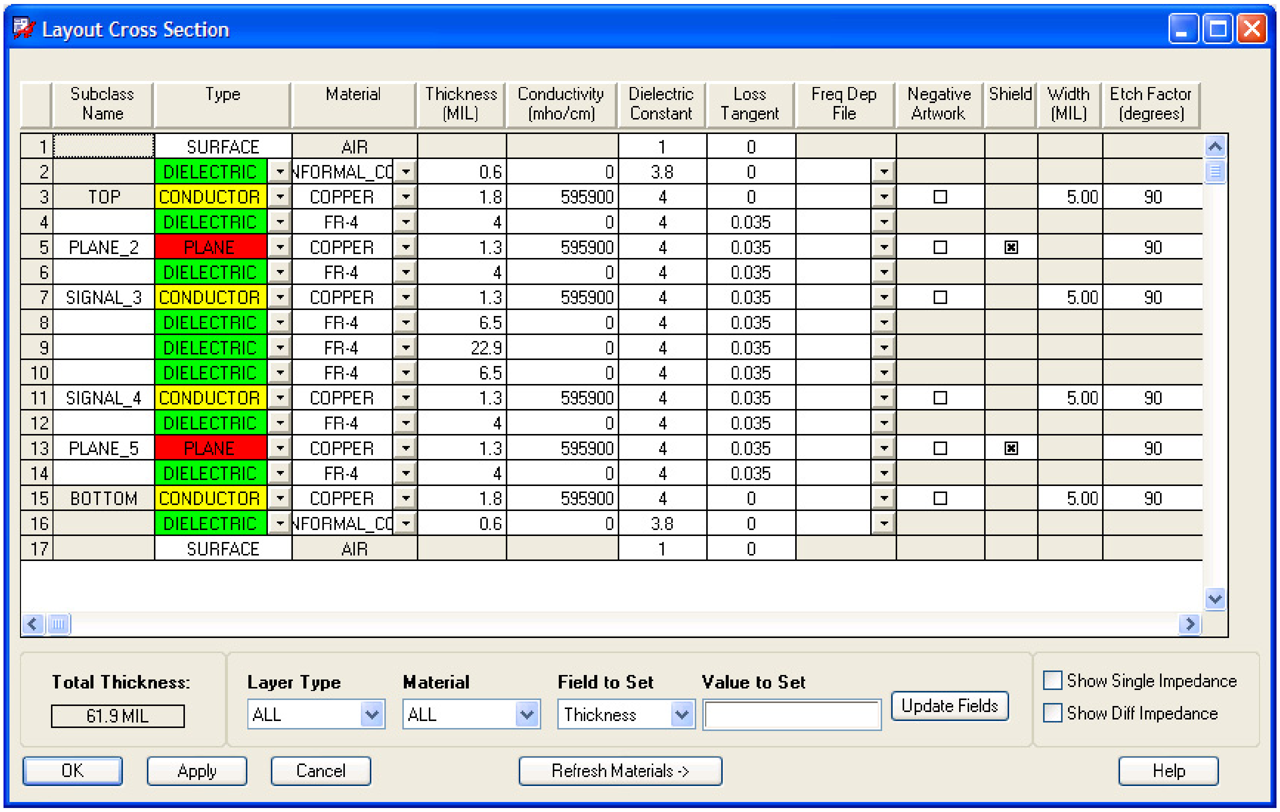
3. Select from *2-, 4-, 6-, or 8-layer* defaults to seed your stackup.
4. Specify a name for the LayerStack and click *OK*.

After a brief delay, the new LayerStack appears in the *LayerStack Manager* dialog box.



At this point, you can *rename* the default LayerStack that you selected, which is a good idea if you intend to add or remove layers.

When you click a LayerStack and choose *Edit*, the *Layout Cross Section* dialog box appears.



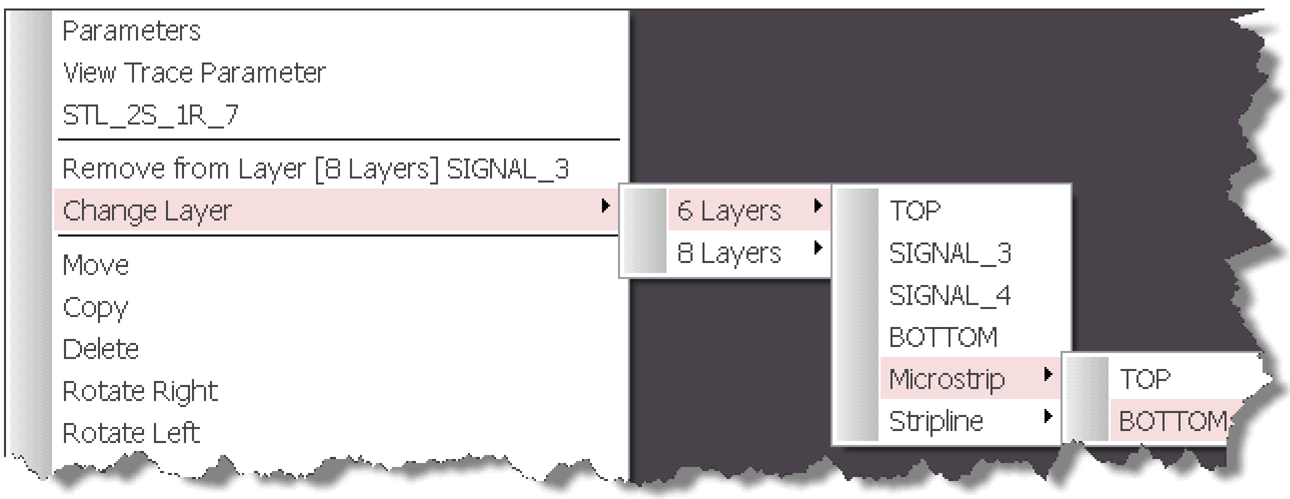
	Subclass Name	Type	Material	Thickness (MIL)	Conductivity (mho/cm)	Dielectric Constant	Loss Tangent	Freq Dep File	Negative Artwork	Shield	Width (MIL)	Etch Factor (degrees)
1		SURFACE	AIR			1	0					
2		DIELECTRIC	NFORMAL_CC	0.6	0	3.8	0					
3	TOP	CONDUCTOR	COPPER	1.8	595900	4	0		☐		5.00	90
4		DIELECTRIC	FR-4	4	0	4	0.035					
5	PLANE_2	PLANE	COPPER	1.3	595900	4	0.035		☐	☒		90
6		DIELECTRIC	FR-4	4	0	4	0.035					
7	SIGNAL_3	CONDUCTOR	COPPER	1.3	595900	4	0.035		☐		5.00	90
8		DIELECTRIC	FR-4	6.5	0	4	0.035					
9		DIELECTRIC	FR-4	22.9	0	4	0.035					
10		DIELECTRIC	FR-4	6.5	0	4	0.035					
11	SIGNAL_4	CONDUCTOR	COPPER	1.3	595900	4	0.035		☐		5.00	90
12		DIELECTRIC	FR-4	4	0	4	0.035					
13	PLANE_5	PLANE	COPPER	1.3	595900	4	0.035		☐	☒		90
14		DIELECTRIC	FR-4	4	0	4	0.035					
15	BOTTOM	CONDUCTOR	COPPER	1.8	595900	4	0		☐		5.00	90
16		DIELECTRIC	NFORMAL_CC	0.6	0	3.8	0					
17		SURFACE	AIR			1	0					

SigXplorer uses the same *Layout Cross Section* dialog box as the SI layout tools.

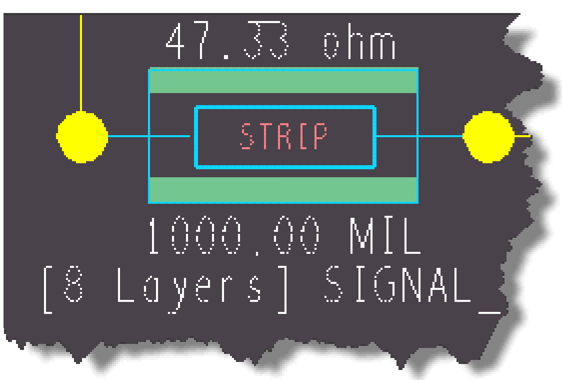
Changing to a Different LayerStack

The RMB (pop-up) menu (Figure 2-7) shows how to disassociate an element from a LayerStack, and how to associate it with a different LayerStack.

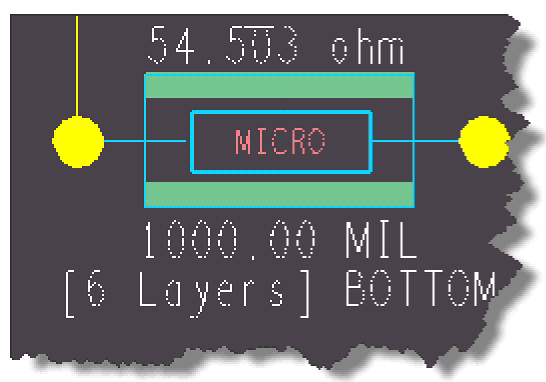
Figure 2-7 Changing Layers



Original [8 Layers]



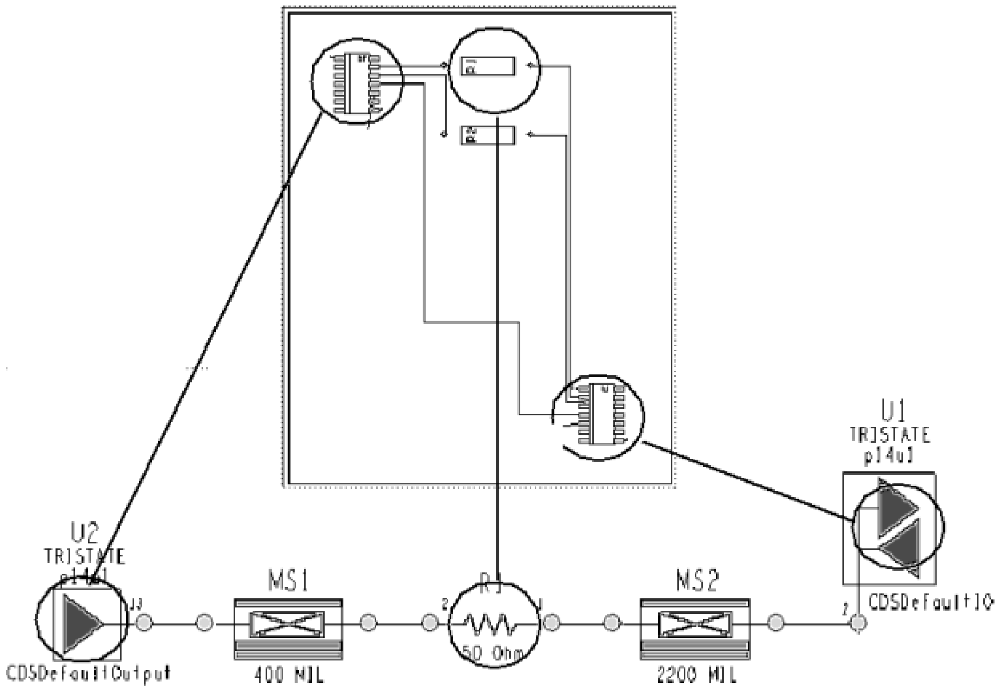
Modified [6 Layers]



In Figure 2-7, the trace model was relocated from an 8-layer, stripline LayerStack to a 6-layer, microstrip LayerStack. Note the change in the label of the trace model, and the change in impedance values.

Extracting a Topology

You extract a net topology from the SI layout tool (directly, or through Constraint Manager) into SigXplorer to see if it meets signal integrity requirements. If not, you can modify the topology until it does meet the requirements.



The *SI Design Setup* command (*Setup – SI Design Setup* in the SI layout tool) is a utility used to bridge the physical design representation in the SI layout tool with the equivalent electrical design representation in SigXplorer, by guiding you through the steps necessary to ensure a clean net extraction from the SI layout tool's database.

3

Preparing for Simulations

Topics in this chapter include:

Exploring the SigXplorer User Interface

Using SigXplorer you can create a circuit topology in the canvas and create or modify the supporting data in the integrated spreadsheet.

For more information on menu commands and procedures, see *Allegro SI SigXplorer Command Reference*.

To set up and perform simulations, you need to use commands accessed through menus and the toolbar, as shown in Table 3-1.

Table 3-1 Performing Simulations

To...	Choose...
Prepare for simulation	*Set – Defaults* from the Menu or the Parameters tab
Perform simulation	*Analyze – Simulate*
Monitor progress of the simulation	Command tab
View simulation results	Results tab or open SigWave to view the waveforms

Common Clock Simulation

Introduction

Once you have a valid topology displayed in the canvas, you are ready to simulate. You can use default simulation parameters to control how the simulation performs, or you can modify the simulation parameters before you start the simulation.

The simulation results appear as data in spreadsheet format and as waveforms. After viewing the simulation results, you can modify the circuit topology and simulation parameters and then re-simulate to examine the effects of your changes.

Repeat this process until the circuit meets your requirements. For information on simulation and analysis, see the *Allegro PCB SI User Guide*.

Setting Analysis Preferences

To set the simulation preferences or to modify the simulation results, choose *Analyze – Preferences* from the SigXplorer menu.

The Analysis Preferences dialog box consists of the following tabs:

Pulse Stimulus	Determines values for the pulse stimulus, including: ■ Measurement Cycle ■ Switching Frequency ■ Duty Cycle ■ Offset
S-Parameters	Determine: ■ Transient Simulation Method ■ DC Extrapolation Method ■ (Enforce) Impulse Response Causality

Simulation Parameters	Determine simulation parameters, including: ■ Fixed Duration ■ Waveform Resolution ■ (Default) Cutoff Frequency ■ Buffer Delays ■ (Save) Sweep Cases ■ Algorithm Model Generation ■ Simulator ■ Solver
Simulation Modes	Select simulation modes to perform a single simulation or simulation sweeping, including: ■ FTS Modes(s): Fast, Typical, Slow, Fast/Slow, Slow/Fast ■ Driver Excitation: Active_Driver, All_Drivers ■ Fast/Typical/Slow Definitions
Measurement Modes	Select Measurement Modes to specify: ■ Measure Delays At ■ Receiver Selection ■ Custom Simulation ■ Drvr Measurement Location ■ Rcvr Measurement Location ■ Report Source Sampling Data ■ Advanced Settings for: ❑ Glitch control ❑ Eye diagram measurements

EMI	Set preferences and defaults for EMI single net simulation, including: ■ EMI Regulation ■ Design Margin ■ Analysis Distance

Measurement Location

Pin and/or die measurement location for driver and receiver can be determined from the DML model defined in the setup, from the external pin node, or from the internal die node, if present. You can set these choices in the Analysis Preferences dialog box (*Analyze – Preferences*) or by using the `signoise` batch command.

Note: Editing measurement locations in the defined DML model involves manually changing the DML file by adding or deleting the appropriate keywords using the correct syntax in the appropriate section. Pin and die measurement locations are made at the external pin node and internal die node, respectively.

The following convention is used in the Results spreadsheet to distinguish whether the measurement is being made at the pin pad or the die pad:

- If taken at the pin pad, the pin pad measurement name is identical to the pin name (for example, PIN5).
- If taken at the die pad location, the pin name is displayed with an i appended to it (for example, Pin5i).

The following examples illustrate these results.

Figure 3-1 Pin Measurement Selection Report

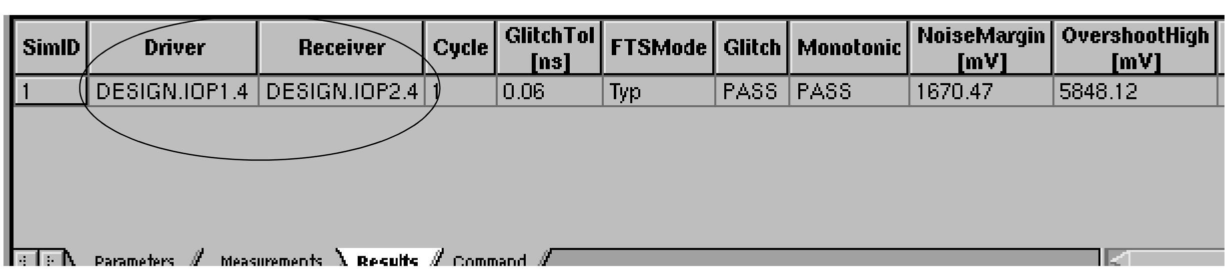
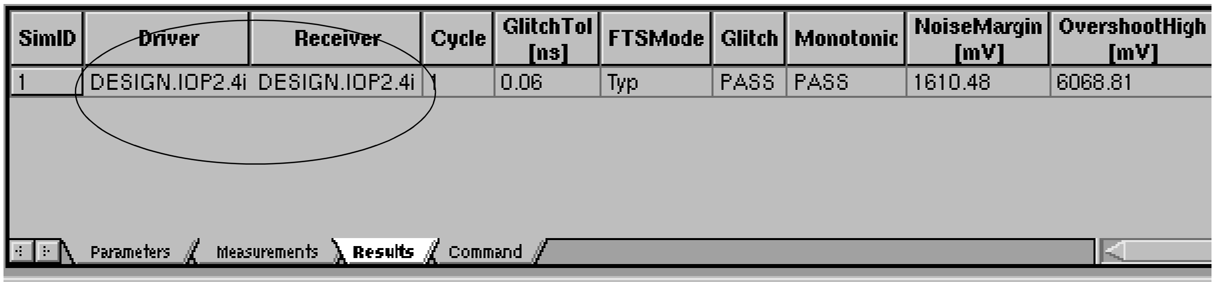


Figure 3-2 Die Measurement Selection Report



SimID	Driver	Receiver	Cycle	GlitchTol [ns]	FTSMode	Glitch	Monotonic	NoiseMargin [mV]	OvershootHigh [mV]
1	DESIGN.IOP2.4i	DESIGN.IOP2.4i	1	0.06	Typ	PASS	PASS	1610.48	6068.81

For more information, see the *Allegro SI SigXplorer Command Reference*.

Setting Stimuli and Running Simulations

When you have added all the parts to (or extracted the parts into) the topology, you need to choose a stimulus type for the driver. You can specify only one IOCell at a time to be the driver. All the other IOCells must be set to `tri-state`.

There are several types of stimulus for a driver:

- Pulse
- Rise
- Fall
- Custom
- Quiet Hi
- Quiet Lo
- Tristate

You choose a stimulus based on the AC input used for simulation. The driver output for a pulse starts at the circuit low DC point, so you use the stimulus to determine the low point.

You then extract a selected net (ratsnest) from the board layout for exploration and topology development. This unrouted interconnect models after Manhattan distance estimates based on your initial placement.

You simulate and analyze the topology, making trade-off decisions that involve:

- target impedance

- min/max length (or propagation delay)
- pin ordering
- termination strategy (and location on net)

When the simulation finishes, the Results tab pops to the top of the Spreadsheet and the SigWave window opens. The Results tab displays data for the simulation, and the SigWave window displays waveforms resulting from the simulation. You can examine the simulation results in the spreadsheet and in SigWave.

You then note what changes to make in the board design. You can modify the board by:

- adding components (terminators, etc.)
- swapping components
- moving components
- moving nets on microstrip layers to stripline layers
- adding shield or etch layers to the stackup
- varying trace geometry
- re-routing to a new pin order

When the simulation results are satisfactory, you can do one of the following:

- If you have extracted the topology into SigXplorer, you can upload the design into Allegro SI.
- If you have created a new topology or opened a previous topology in SigXplorer, you can proceed by saving the constraints with the topology (see Defining Constraints on page 85).

Performing Parametric Sweeps

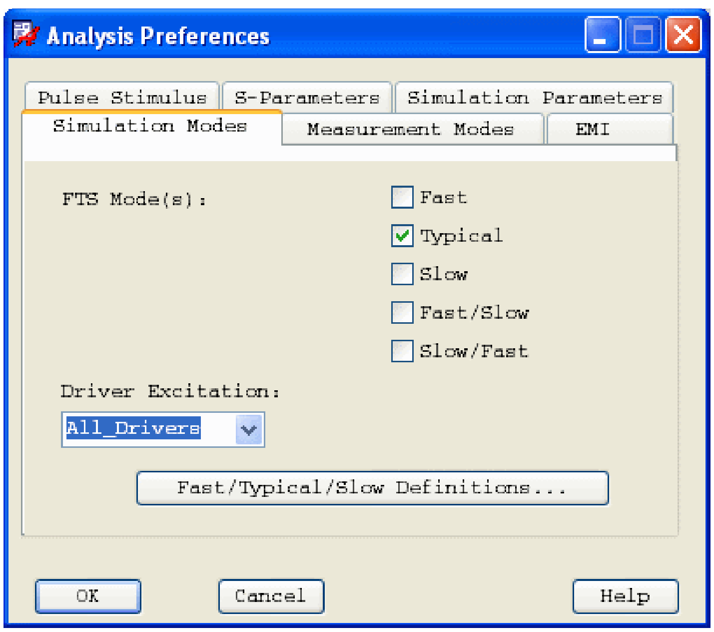
Simulation sweeping relies on combinations of the following criteria:

- Varying part parameter values
- Varying driver slew rates
- Sequencing active drivers

Sweeping by part parameter values involves covering a set or range of values (sweep count points) that you specify for eligible sweep parameters through a set of simulations. SigXplorer calculates the total number of simulations based on the number of sweep count points required for each sweep parameter.

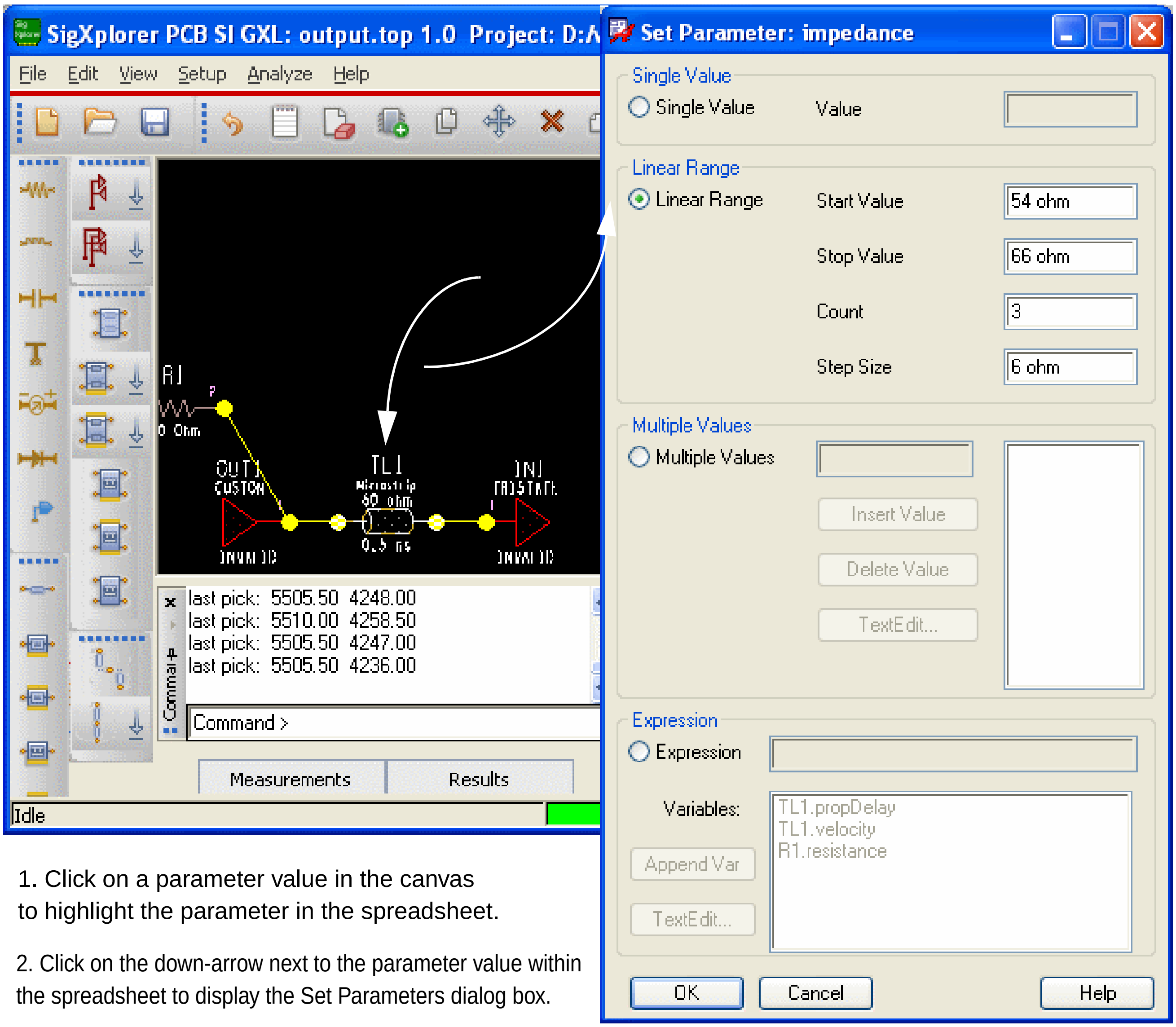
You sweep by driver slew rate by selecting a set of FTS Mode target rates from the *Simulation Modes* tab in the Analysis Preferences dialog box.

You accomplish sweeping by sequencing active drivers by selecting *All Drivers* sweep mode to sequence through eligible IOCells with each one, in turn, driving a simulation.



When you specify multiple sweep criteria, SigXplorer uses a hierarchical ordering when performing the simulations. For example, if you select multiple FTS Modes, as well as several part parameter values for sweeping, then all part parameter sweeps execute for each selected FTS Mode. Additionally, if you also select All Drivers, then part parameter sweeps for each selected FTS Mode execute as each driver activates in sequence.

Figure 3-3 Sweep Parameter Setup



Specifying Part Parameter Values for Sweeping

All parameter attributes, including parameter that you can sweep, are accessible for viewing and editing through the Parameters tab of the SigXplorer spreadsheet.

Parameters that you can sweep include:

- single number value

- linear range of number values specified as start and stop values and a step size for iterating from start (the minimum value) to stop (the maximum value).
- list of discrete number values
- expression string composed of operators, functions, and references to other parameters.

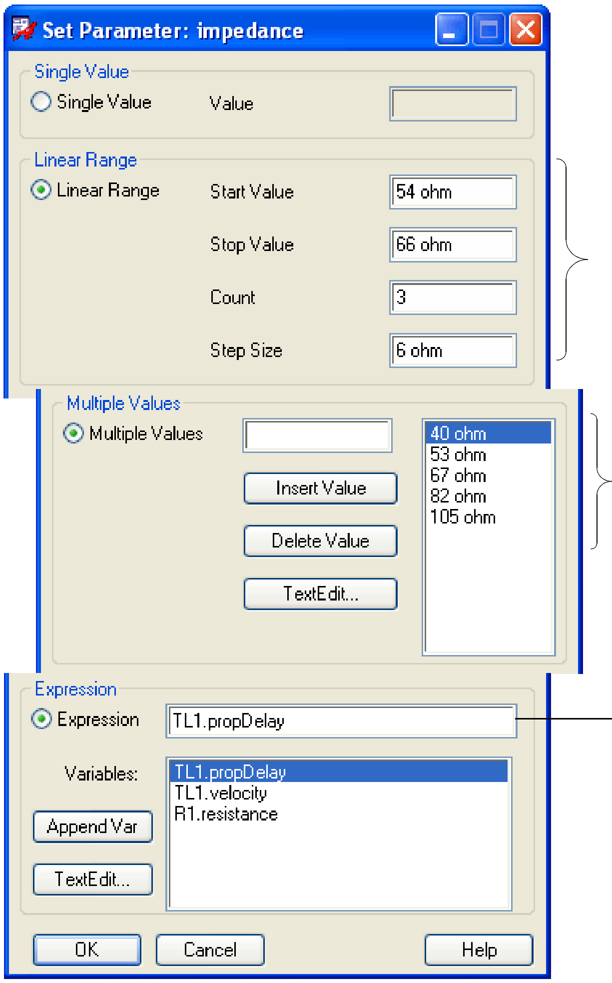
When you use an expression to define a parameter attribute that references a second parameter attribute defined as a range or list, the first parameter tracks the second parameter as it changes during simulation sweeping.

By defining an expression that references another parameter and adds a constant, you can track the first parameter with an offset.

When you delete a part, any references to the part parameter are no longer valid and appear in red within the spreadsheet.

Note: You can save a topology that contains invalid references, but you cannot simulate it.

Figure 3-4 Setting Sweep Parameters using the Set Parameters Dialog Box



Using a Linear Range

Count value determines the number of sweep count points.

Using Multiple Values

Discrete values determine the number of sweep count points.

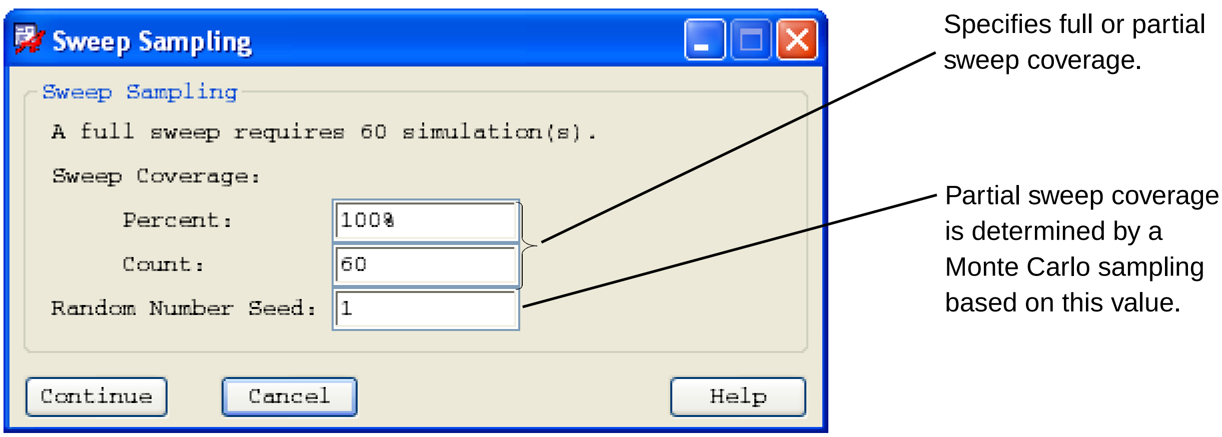
Using an Expression

Expression listed, as well as parameters referenced in the expression determine the number of sweep count points.

Controlling Sweep Sampling and Coverage

After you have set up to perform simulation sweeps, you can choose *Analyze — Simulate* to control sweep sampling in SigXplorer. The Sweep Sampling dialog box appears before an active sweep begins.

Figure 3-5 The Sweep Sampling Dialog Box



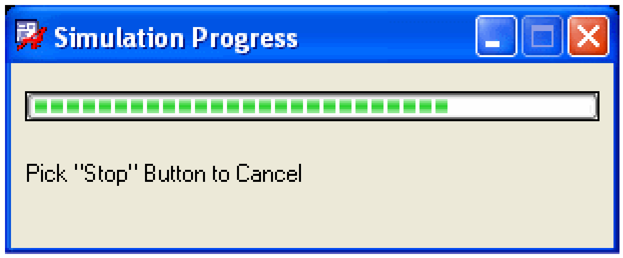
You can specify full or partial sweep coverage in this dialog box by:

- defining sweep samples as a percentage of full coverage.
- specifying an explicit number of simulations.
- specifying a seed number for random sampling.

You obtain partial sweep coverage by randomly sampling the full solution space using Monte Carlo methods. To vary sample point sets, SigXplorer selects sweep count points based on the specified random number seed.

Sweep Results

When you click *Continue* on the Sweep Sampling dialog box, the simulation begins and the Simulation progress dialog appears:
\



When you invoke parametric sweeping, SigXplorer initializes SigNoise which sweeps through the required series of simulations. Sweep results appear in the Results tab of the SigXplorer spreadsheet.

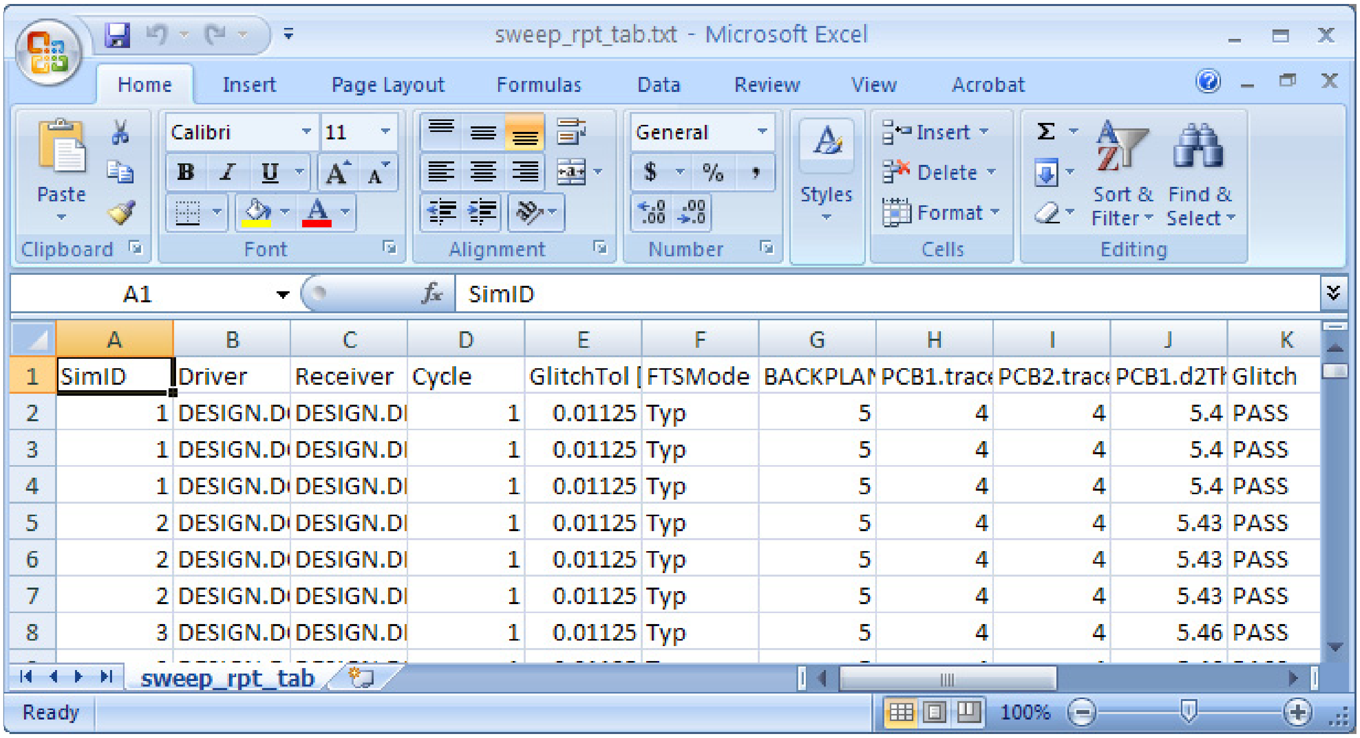
Figure 3-6 Sweep Results

SimID	Driver	Receiver	Cycle	GlitchTol [ns]	FTSMode	BACKPLANE.trace Width2	PCB1.trace Width2	PCB2.trace Width2	PCB1.d2 Thickne	Glitch
1	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.4	PASS
1	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.4	PASS
1	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.4	PASS
2	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.43	PASS
2	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.43	PASS
2	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.43	PASS
3	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.46	PASS
3	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.46	PASS
3	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.46	PASS
4	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.49	PASS
4	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.49	PASS
4	DESIGN.DOU	DESIGN.DIN	1	0.01125	Typ	5	4	4	5.49	PASS

Results

The sweep report contains information on topology, swept elements, driver and load names, impedance, and delay variables. You can save simulation sweep results in a (Spreadsheet Tabbed Text) tab-delimited text file using the *File – Export – Spreadsheet – Results* menu command. The contents of this file can be imported into an external spreadsheet program such as Microsoft Office Excel as shown in Figure 3-7 .

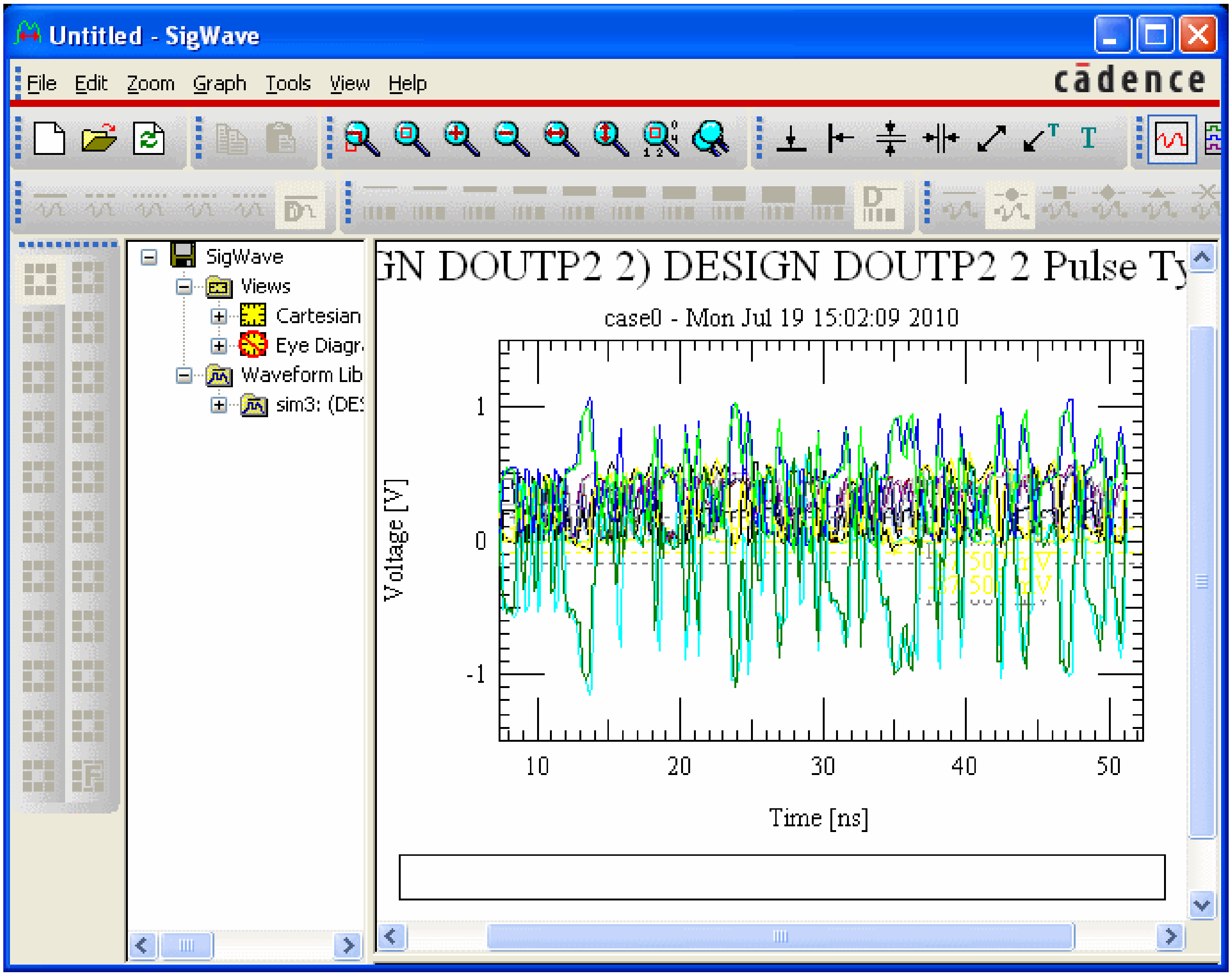
Figure 3-7 Sweep Results Exported to a Tab-Delimited Text File



Viewing Waveforms

The parametric sweep function does not produce waveforms directly. However, when viewing the sweep results in the Results tab, you can right-click a row in the spreadsheet and choose *View Waveform*. This will re-run that single simulation and open SigWave to display the resulting waveforms.

Figure 3-8 Waveform in SigWave



You can save and restore sweep simulation data. This enables you to view waveforms from any prior sweep iteration, eliminating the need to manually reset simulation parameters and perform re-simulations. You save the waveforms (*File – Save As* in SigWave) and the environment details in a case directory. Upon restoring the sweep case, you return to the same state, ensuring the data accuracy of the waveforms.

Caution

Saving waveforms from sweeps can consume large amounts of disk space.

About Sweep Case Data

Saved sweep cases comprise the following data:

- Waveforms
- Topology file
- *SigNoise* preferences
- Results spreadsheet

Saving Sweep Cases

When you start SigXplorer, it uses a default case directory to save (single or sweep) simulation results. The data in the default case is temporary as it is automatically overwritten with the data from the next simulation. Before running a sweep simulation, you can elect to save sweep case data using the Analysis Preferences dialog box shown in Figure 3-9.

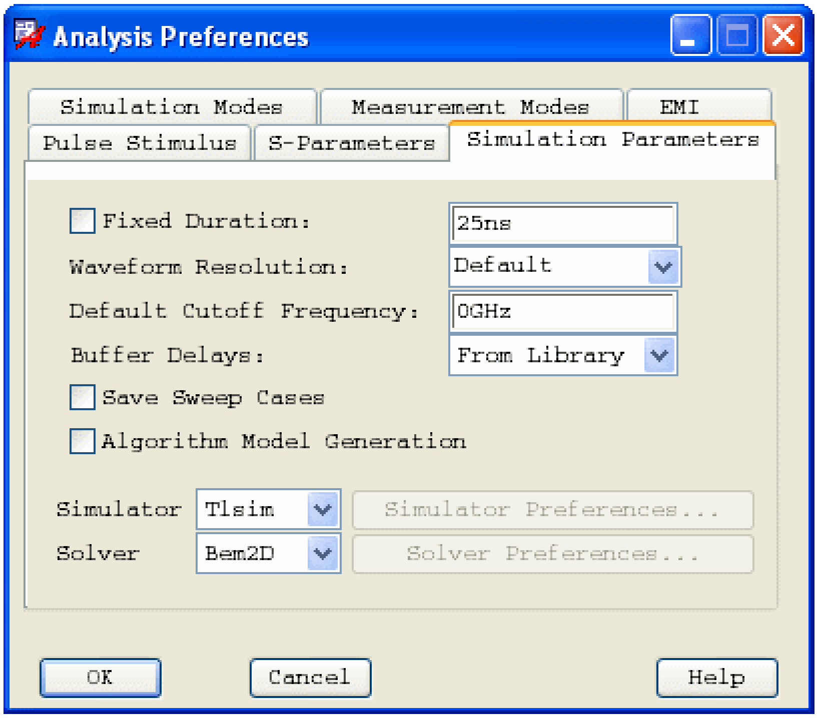
To save sweep cases

1. Choose *Analyze – Preferences* in SigXplorer.

   The Analysis Preferences dialog box appears.

2. Select the *Simulation Parameters* tab.

Figure 3-9 Analysis Preferences Dialog Box

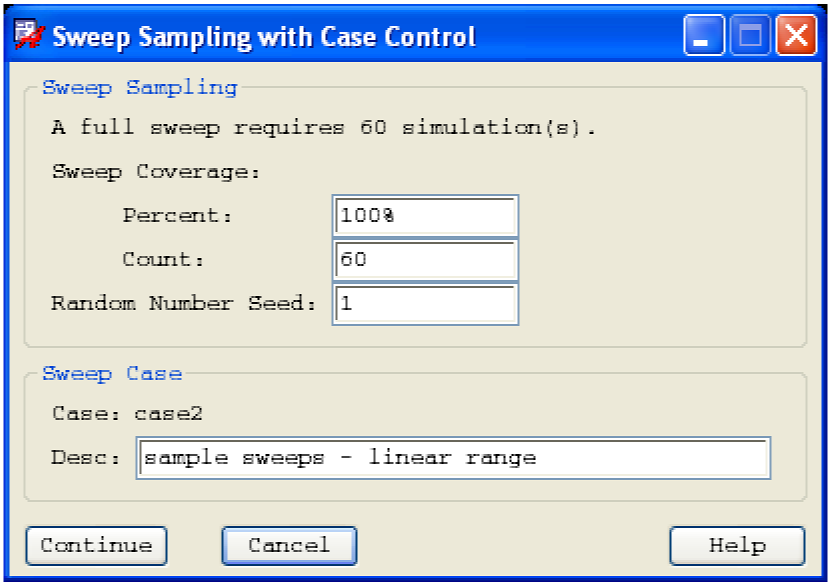


3. Click *Save Sweep Cases*.
4. Select other tabs to set additional preferences for simulation sweeps.
5. Click *OK*.

Running the Simulation

Running a sweep simulation opens the **Sweep Sampling with Case Control** dialog box as shown in Figure 3-10.

Figure 3-10 Sweep Sampling Dialog Box



Within the Sweep Case area, the assigned case number appears along with a case description field to use to enter text regarding the sweep.

When you click *Continue*, the following events are triggered to preserve the current sweep simulation data and environment details:

- Current topology file and SigNoise preferences are saved in the current case directory.
- Sweep simulation starts, saving the resultant waveforms in the case directory. After the sweep finishes, the data from the Results spreadsheet is also saved in the case directory.

Note: Initiating a sweep simulation with the *Save Sweep Cases* preference enabled, the system assigns the *next unused* case (in this example, case2) as the directory for the saved sweep data.

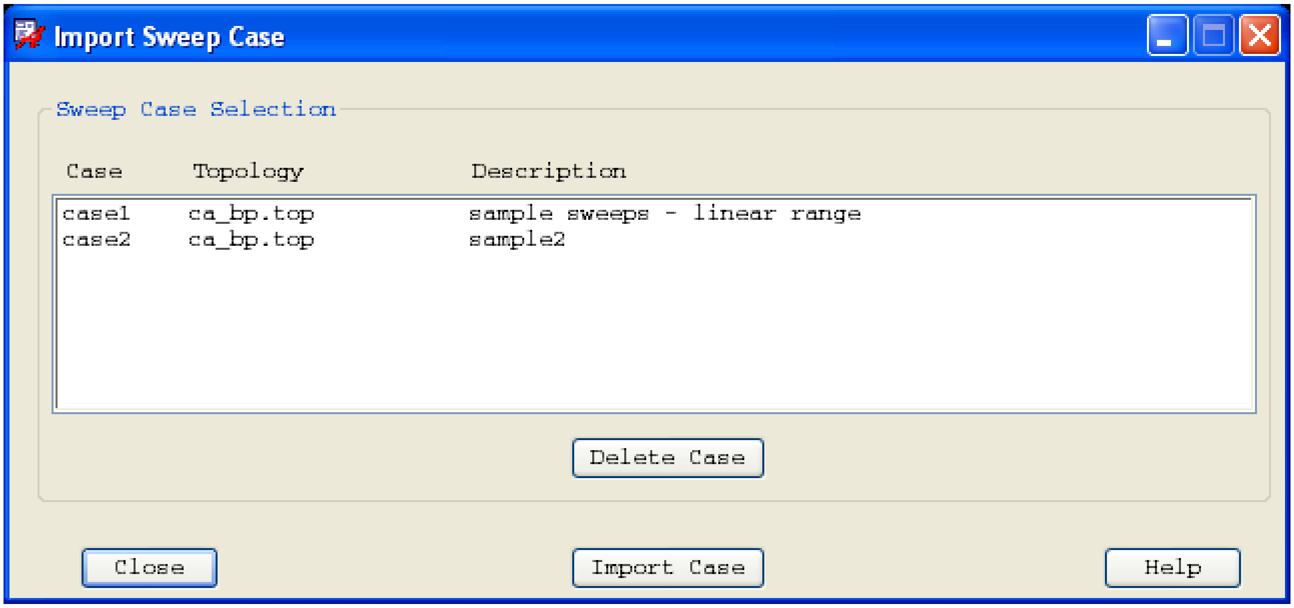
Restoring and Deleting Sweep Cases

You can restore a saved sweep using the Import Sweep Case dialog box shown in Figure 3-11

1. Choose *File – Import – Sweep Case*.

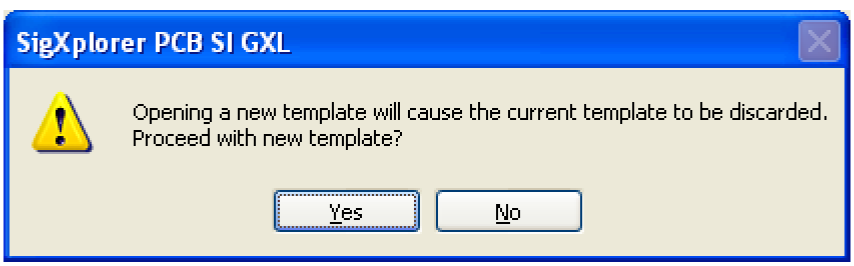
The Import Sweep Case dialog box appears.

Figure 3-11 Import Sweep Case Dialog Box



2. Select the sweep case you want to restore from the list and click *Import Case*.

   A message appears prompting you to save the current topology in SigXplorer



3. Click *Yes*.

   The topology file from the selected case is loaded into SigXplorer, the Results spreadsheet data from the selected case is imported, and the SigNoise preferences are also restored from the selected case.

To delete a sweep case:

1. Click the sweep case you wish to delete from the list.
2. Click *Delete Case*.

The case data is deleted and the case entry removed from the list.

Viewing Sweep Case Waveforms

To view a sweep case waveform:

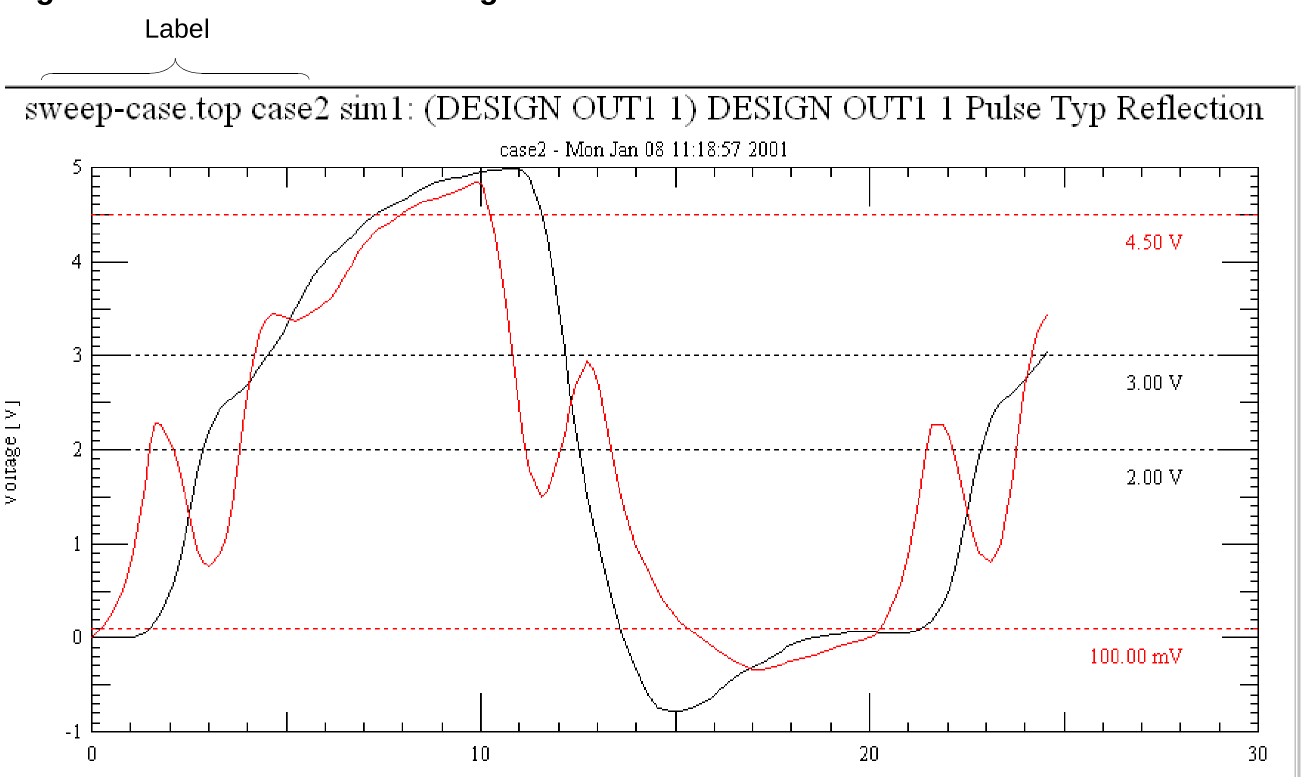
1. Right-click on the desired simulation row within the Results spreadsheet.
2. Click the *View Waveform* button.

   The SigWave window appears displaying the resultant waveform of the selected simulation.

Waveform Labels

Each waveform has its own label to assist you in mapping the data in the SigXplorer spreadsheet and the SigWave window. The label is based on the topology and case names, as well as the spreadsheet row and simulation ID number of the respective waveform. Figure 3-12 shows a waveform label.

Figure 3-12 Waveform Labeling



Crossprobing

Between two compatible applications, crossprobing enables you to highlight a design object in one application when you select the object in the other. You can crossprobe waveform objects in the SigWave window and simulation rows in the SigXplorer Results spreadsheet to enable quick and reliable identification of waveforms and the related spreadsheet data.

To crossprobe a SigXplorer Results Spreadsheet Row from SigWave

1. View a sweep case waveform from the Results spreadsheet.

   If necessary, re-position the *SigWave* window so that the Results spreadsheet in *SigXplorer* and *SigWave* are displayed simultaneously.

2. Click any one of the following waveform objects in the *SigWave* window:

   - Curve in the graphics pane.
   - Legend symbol (bottom of the graphics pane).
   - Symbol in the tree pane.

   The corresponding Results spreadsheet row for the selected waveform is highlighted.

To crossprobe a SigWave waveform from a SigXplorer Results Spreadsheet Row

1. View a sweep case waveform from the Results spreadsheet.

   If necessary, re-position the *SigWave* window so that the Results spreadsheet in *SigXplorer* and SigWave displays simultaneously.

2. Click on a simulation row within the Results spreadsheet of SigXplorer.

   The corresponding waveform object in SigWave is highlighted.

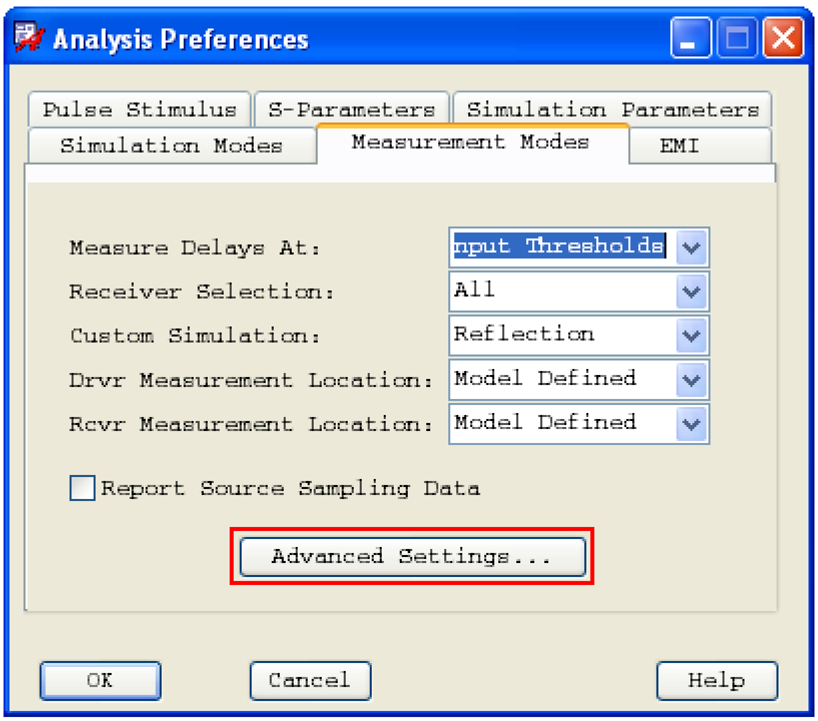
Setting Advanced Measurement Parameters

You can set measurement parameters which assist you in finding correct cycles in your waveform, such as governing glitch tolerance, and measuring eye opening and peak-to-peak jitter. You specify these settings in the Set Advanced Measurement Parameters dialog box.

To access the Set Advanced Measurement Parameters dialog box:

1. Choose *Analyze – Preferences*.
2. Select the *Measurement Modes* tab in the Analysis Preferences dialog box.

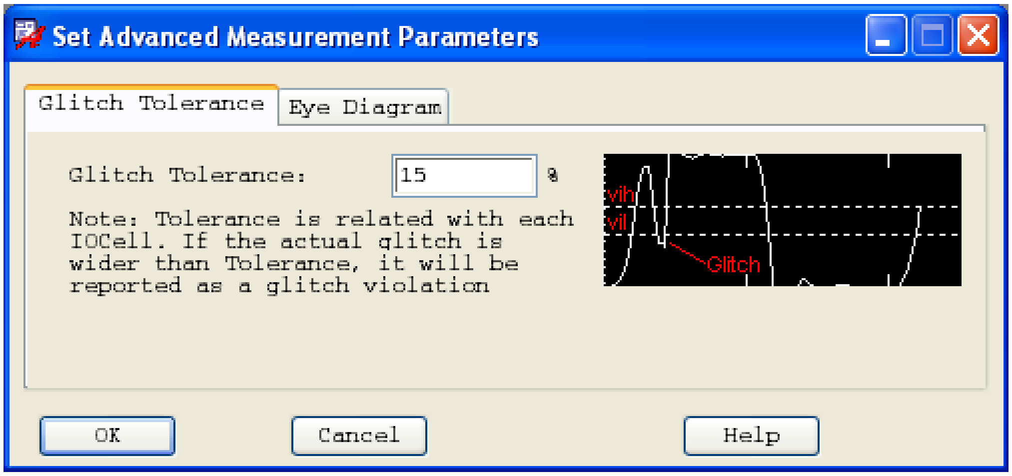
Figure 3-13 Measurement Modes – Advanced Settings Control



3. Click the *Advanced Settings* button.

   The Set Advanced Measurement Parameters dialog box is displayed.

Figure 3-14 Set Advanced Measurement Parameters Dialog Box



Measuring and Controlling Glitch

You can control glitch by setting a glitch tolerance percentage that can assist you in finding correct cycles in your waveform. The glitch tolerance setting is a relative percentage of the faster of the rising and falling edges of each IO cell buffer model you need to measure.

When a glitch occurs between the starting and ending points of a cycle, a glitch violation is reported if the value of the glitch exceeds the tolerance percentage specified in the *Glitch Tolerance* field (Figure 3-14). The glitch is *not* reported as a cycle.

When you import your board design as a topology file in SigXplorer, you can specify the glitch measurements you want to measure by selecting them in the *Reflection* category of the *Measurements* spreadsheet tab of SigXplorer:

Figure 3-15 Measurement Spreadsheet - Glitch Controls

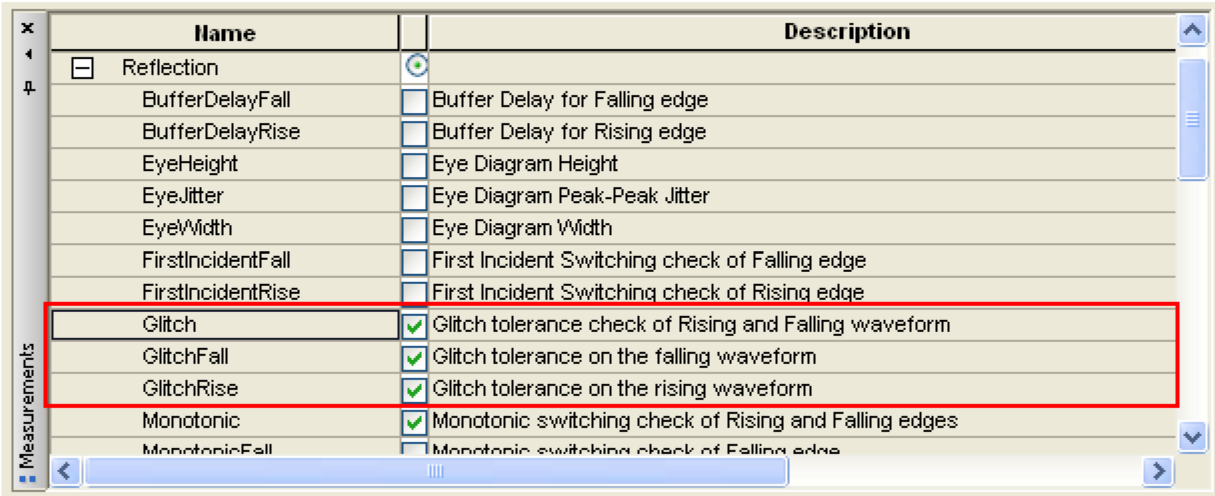


Table 3-2 Glitch Controls Description

Option	Description
Glitch	Is the tolerance check of the rising and falling waveform.
GlitchRise	Is the tolerance check on the rising waveform. If no glitch occurs in the rising waveform, the Results spreadsheet denotes a PASS in the GlitchRise column. If one does occur, it reports a FAIL.
GlitchFall	Is the tolerance check on the falling waveform. If no glitch occurs in the falling waveform, the Results spreadsheet denotes a PASS in the GlitchFall column. If one does occur, it reports a FAIL

Glitch tolerance values are saved in the topology file and in the `sigxp.run` case management directory. If the tolerance values in these locations differ, the tolerance in the topology file takes precedence.

Eye Diagram Measurements

Eye diagrams are a quick way of intuitively assessing the quality of a digital signal. Eye diagrams provide an accessible and intuitive view of parametric performance.

To measure the eye diagrams of drivers which have a custom stimulus (that is, a stimulus other than pulse, rise, fall, etc.), the horizontal and vertical eye opening, and peak-to-peak jitter within wave forms are included in the eye diagram measurements.

When you check the EyeHeight, EyeJitter, and EyeWidth items in the Reflection section of the *Measurements* spreadsheet, the measurements are displayed in the *Results* spreadsheet following the simulation.

The Eye Diagram tab in the Set Advanced Measurement Parameters dialog box displays the current eye diagram parameter settings for the combinations of the drivers and receivers of the topology.

Figure 3-16 Eye Diagram Tab

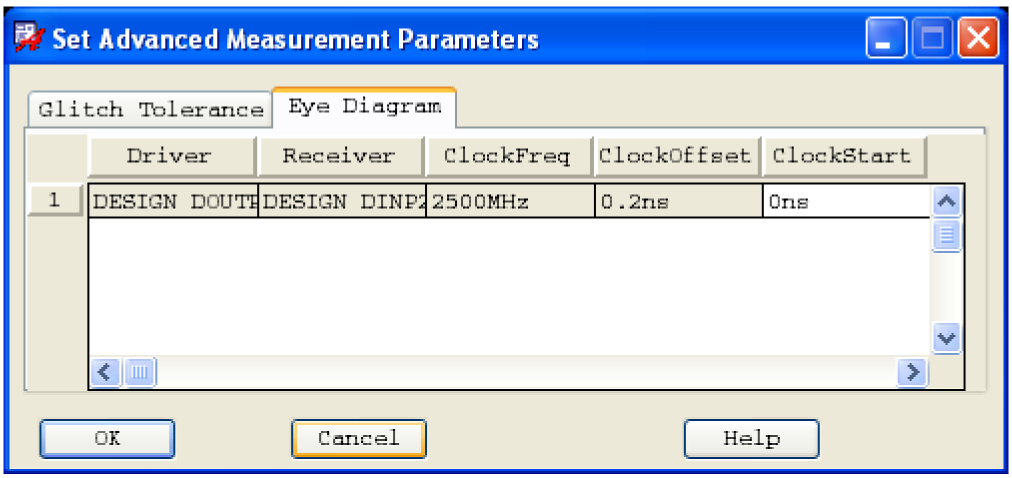
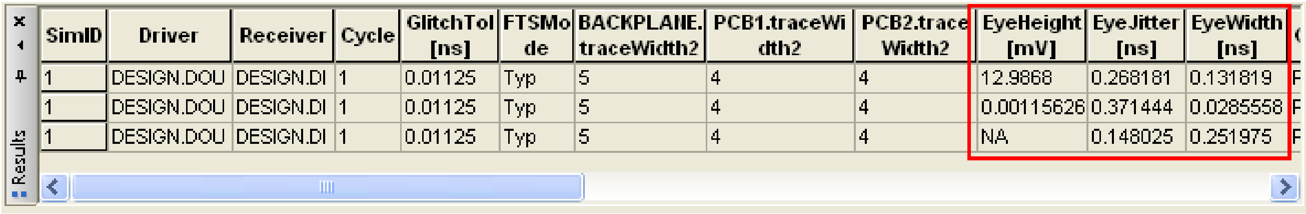


Table 3-3 Eye Diagram Controls Description

Option	Description
Driver/Receiver	Display the driver/receiver combinations in the topology
ClockFreq	Displays the value of the Custom Stimulus state set in the IOCell Stimulus Edit dialog box
ClockOffset	Displays the value in nanoseconds of 1/2 the clock frequency value.
ClockStart	Lets you define the point in time when the eye pattern data should start. The default value is *0ns*. This field is editable.

When you select the *EyeHeight*, *EyeJitter*, and *EyeWidth* options in the *Reflections* section of the *Measurements* spreadsheet, and run the simulation, the Results spreadsheet is populated with the simulation result values.

Figure 3-17 Eye Diagram Measurements in the Results Spreadsheet



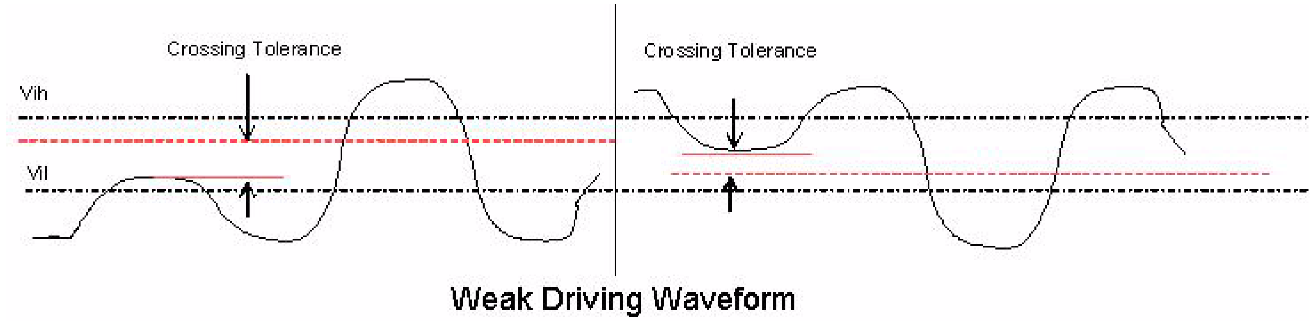
SimID	Driver	Receiver	Cycle	GlitchTol [ns]	FTSMode	BACKPLANE.traceWidth2	PCB1.traceWidth2	PCB2.traceWidth2	EyeHeight [mV]	EyeJitter [ns]	EyeWidth [ns]
1	DESIGN.DOU	DESIGN.DI	1	0.01125	Typ	5	4	4	12.9868	0.268181	0.131819
1	DESIGN.DOU	DESIGN.DI	1	0.01125	Typ	5	4	4	0.00115626	0.371444	0.0285558
1	DESIGN.DOU	DESIGN.DI	1	0.01125	Typ	5	4	4	NA	0.148025	0.251975

See the procedure for performing eye diagram measurement in the *SigXplorer Command Reference*.

Weak Driving Control

The weak driving control functionality automatically determines whether a cycle affected by a weak driver is counted or ignored. When the maximum point of the rising edge does not cross `Vih` (input logic high) but the differential of `Vih` to the maximum point is smaller than the crossing tolerance, the cycle is counted. When this differential is larger, the cycle is ignored. This is illustrated in Figure 3-18.

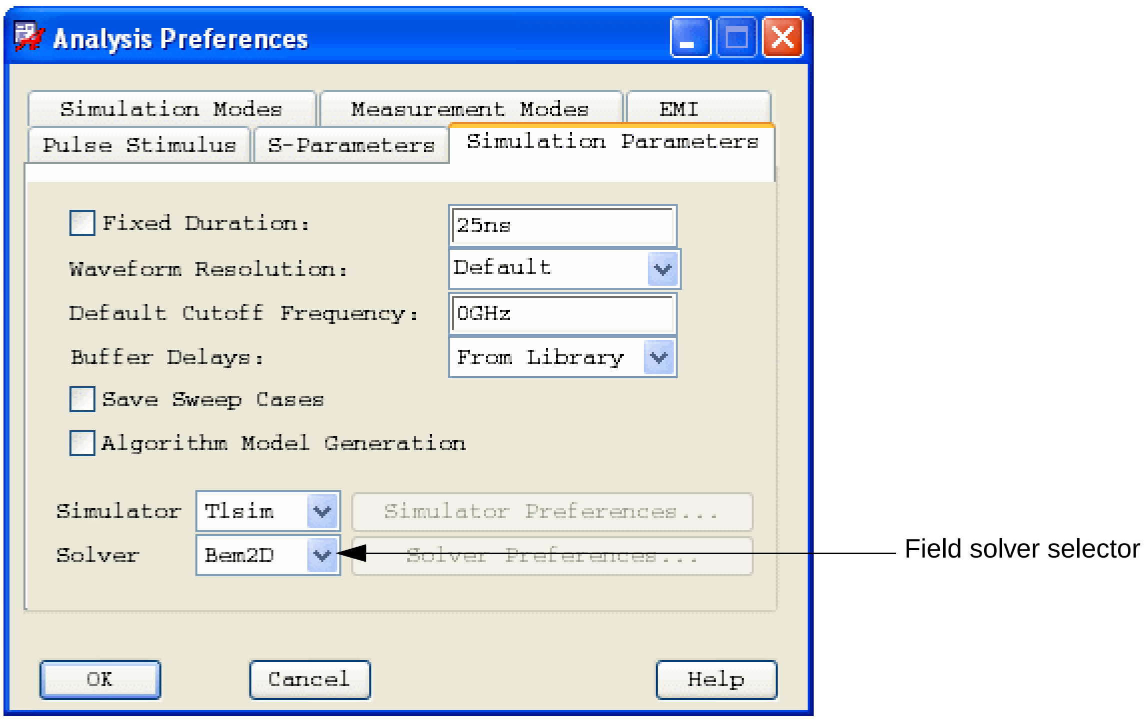
Figure 3-18 Weak Driving Control



Full Wave Field Solvers

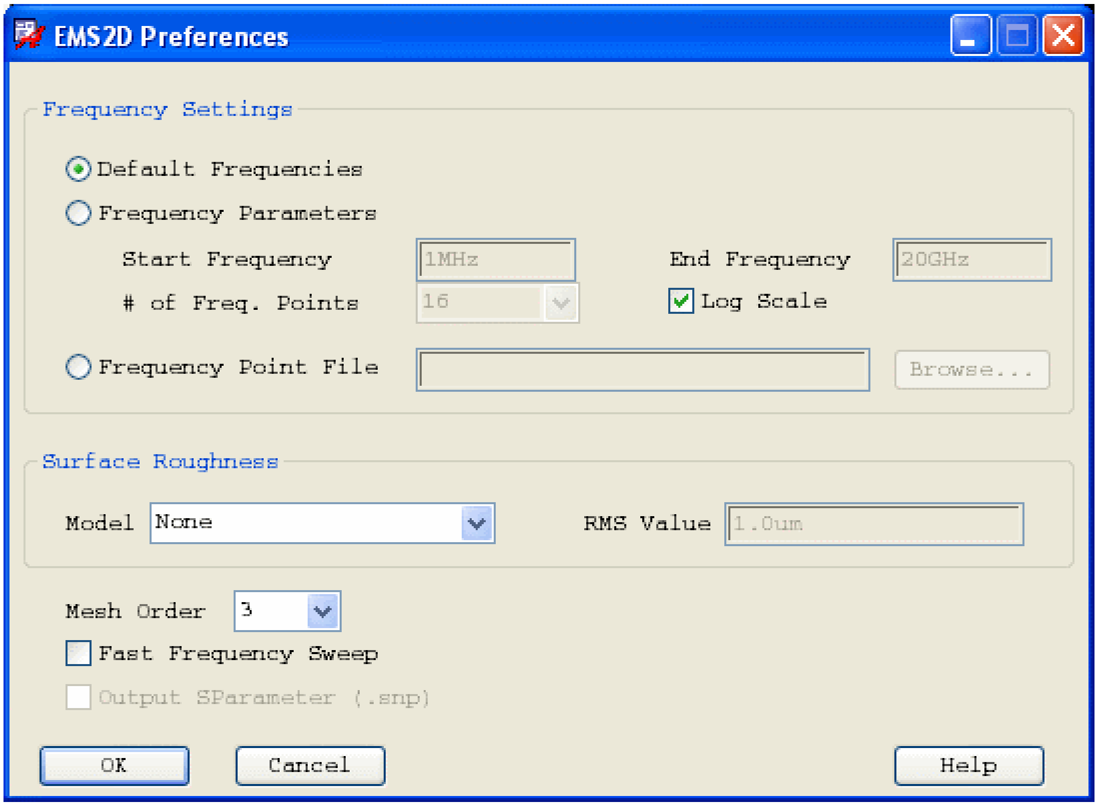
SigXplorer supports two field solvers, *Bem2d* and *Ems2d*. New models that you place in SigXplorer, either from PCB SI or by way of the *Add Element* functionality, attempt to use the solver selected in the *Simulation Parameters* tab of the Analysis Preferences dialog (shown in Figure 3-19). Pre-existing models will attempt to use the field solver type initially used to solve the model.

Figure 3-19 Field Solver Selection Control



Bem2d runs largely automatically, using default parameters. Ems2d allows you to set more preferences, by way of the EMS2D Preferences form. For details on all the controls and options in both these forms, see the online documentation available from the *Help* buttons.

Figure 3-20 EMS2D Preferences Form

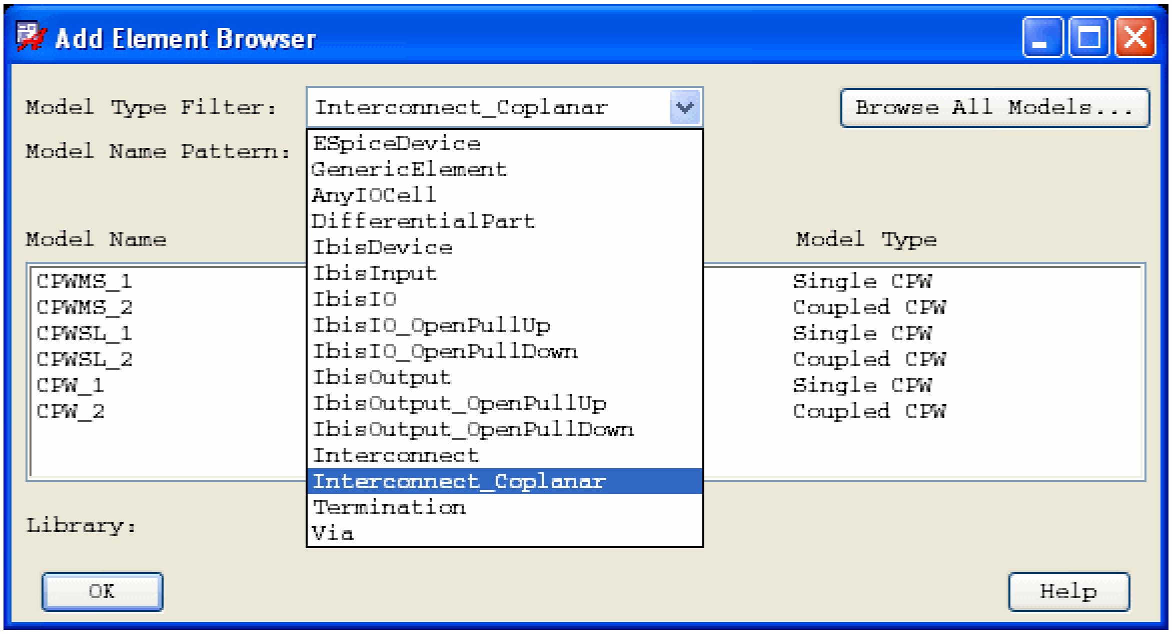


(For complete details on Edms2d and supporting components, see *“Dynamic Analysis with the EMS2D Full Wave Field Solver”* in the PCB SI User Guide.)

Coplanar Waveguide Support

SigXplorer provides full-time support for coplanar waveguide (CPW) structures in a ***Help*** topology, whether extracted from a board layout or added directly to the canvas from the Add Element Browser (*Edit – Add Element*), shown in Figure 3-21.

Figure 3-21 CPW Support in the Model Browser Form

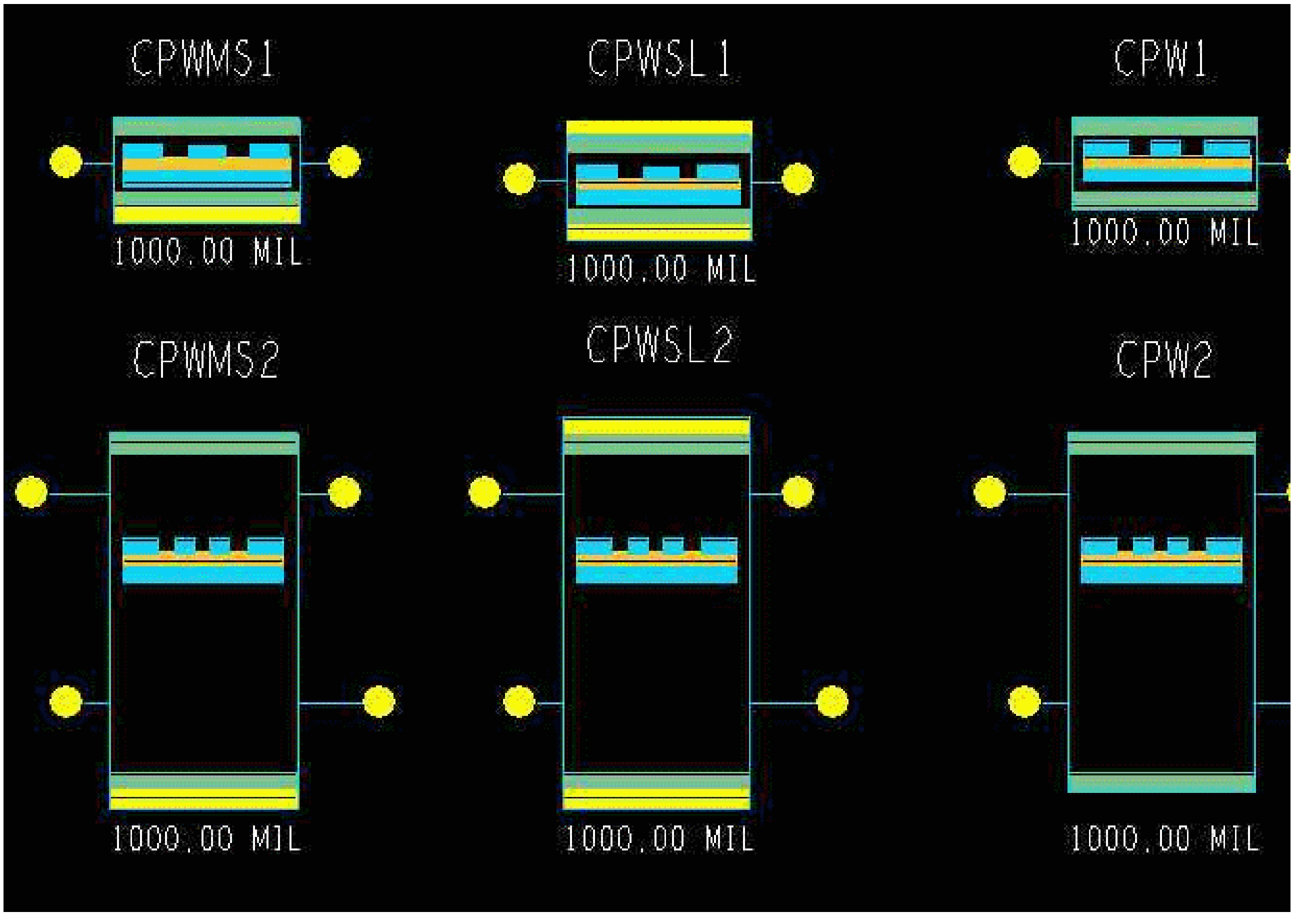


Six CPW structures are supported in SigXplorer:

CPW Structure	Description
Single microstrip	CPWMS1 two-pin symbol containing top and bottom dielectrics
Coupled microstrip	CPWMS2 four-pin symbol containing a bottom dielectric
Single stripline	CPWSL1 two-pin symbol containing top and bottom dielectrics
Coupled stripline	CPWSL2 four-pin symbol containing top and bottom dielectrics
SingleCPW	CPW1 two-pin symbol containing no dielectrics
DiffPair CPW	CPW2 four-pin symbol containing no dielectrics

The symbols for each are illustrated in Figure 3-22.

Figure 3-22 Coplanar Waveguide Symbols



4

Assigning Constraints in SigXplorer

Topics in this chapter include

Introduction

A constraint is a user-defined limit applied to an element in a design. In SigXplorer, you define topology template constraints. SigXplorer uses these constraint rules to drive both signal integrity and EMI analysis.

You can add user-defined constraints to a topology to store other supplementary constraints within a topology to later import into an electrical constraint set (ECSet) using Constraint Manager. You access these values from the design directly by the user or by other software systems.

As with all other constraints, any bus, differential pair, Xnet or net of the assigned ECSet inherits user-defined constraints. Although there are no pre-defined checks to handle these constraints, you could write a Skill routine that retrieves the constraint for a net and then performs a user-defined check. You could also have the Skill routine create a DRC marker. Alternately, these constraint values write to a file using the *extracta* program and then perform checks on the extracted data.

For more information on constraints, see the *Constraint Manager User Guide*.

Defining Constraints

You can define the following constraints in SigXplorer.

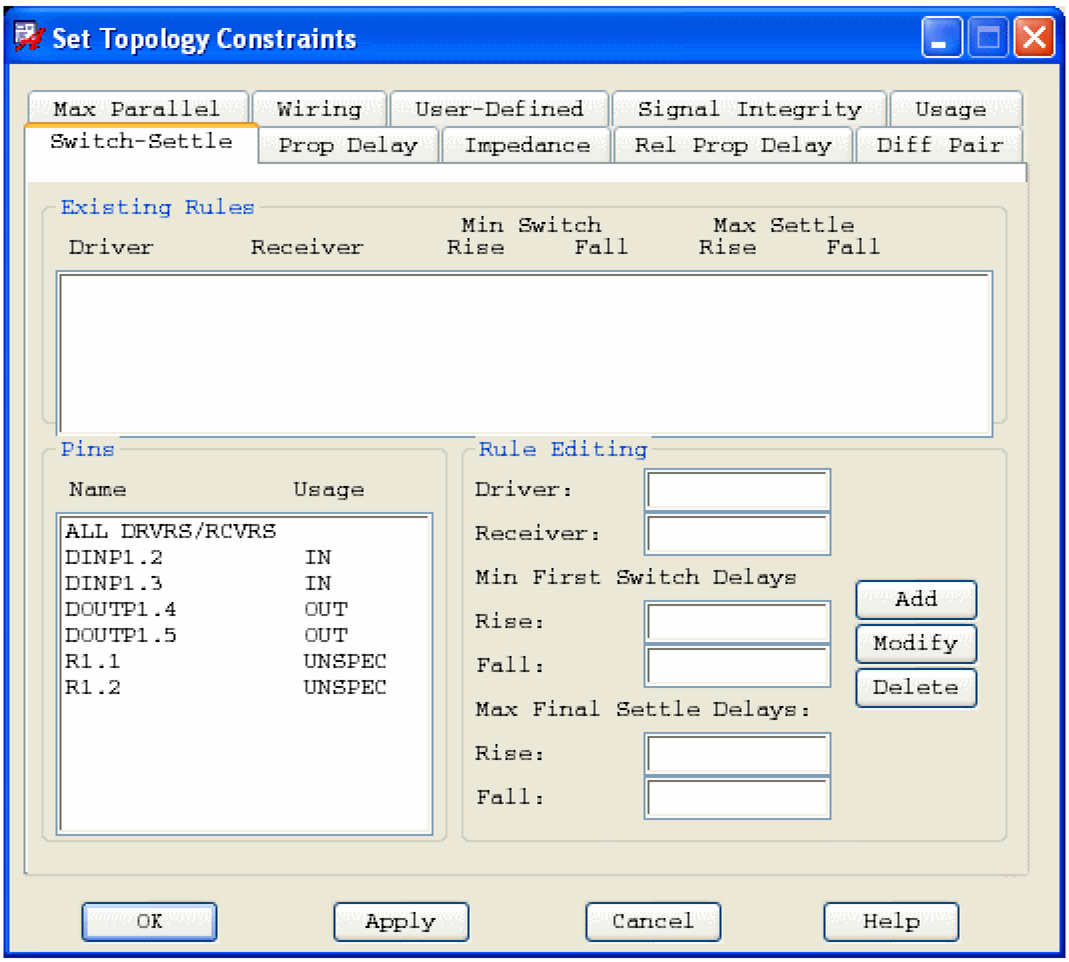
Switch-Settle	Define switch and settle delay constraints between any driver - receiver pin pair. The current rules and a list of pins appear, as currently defined.
Propagation Delay	Defines the delay in time or connection length restriction between any two pins on a net or between any pin and a T-point.
Impedance	Defines the baseline impedance value and allowable tolerance value above and below the baseline. An impedance constraint compares to the impedance of each cline segment of an extended net.
Relative Propagation Delay	Defines connections that are part of a match group. You can specify relative propagation delays between nets and within a net, as well. Assigning the `PROPAGATION_DELAY` property to one of the connections in a match group restricts all other connections in the group.
Differential Pair	Assigns differential pair rules to differential pair objects in a board design. Since a differential topology can contain two separate Xnets, SigXplorer does not allow a single Xnet constraint definition between pins on different Xnets.
Max Parallel	Defines the maximum parallelism constraint between nets. This dialog box tab shows the current coupled length and distance gap rules of the current template.
Wiring	Define topology scheduling parameters as well as physical and EMI constraint rules.
User - Defined	Define supplemental constraints for later use.
Signal Integrity	Define crosstalk, noise, and physical constraint rules.
Usage	Displays application-specific information on constraint usage for the current topology analysis.

Setting Constraints

You create and modify topology constraints in SigXplorer using the Set Topology Constraints dialog box (See Figure on page 86).

See *Allegro SigXplorer Reference* for detailed information on how to set constraints in SigXplorer.

Figure 4-1 Set Topology Constraints Dialog Box



To write the modified constraint values back to the design database (Constraint Manager), choose *File – Update Constraint Manager* in SigXplorer.

5

Common Clock Interface

Topics in this chapter include

Introduction

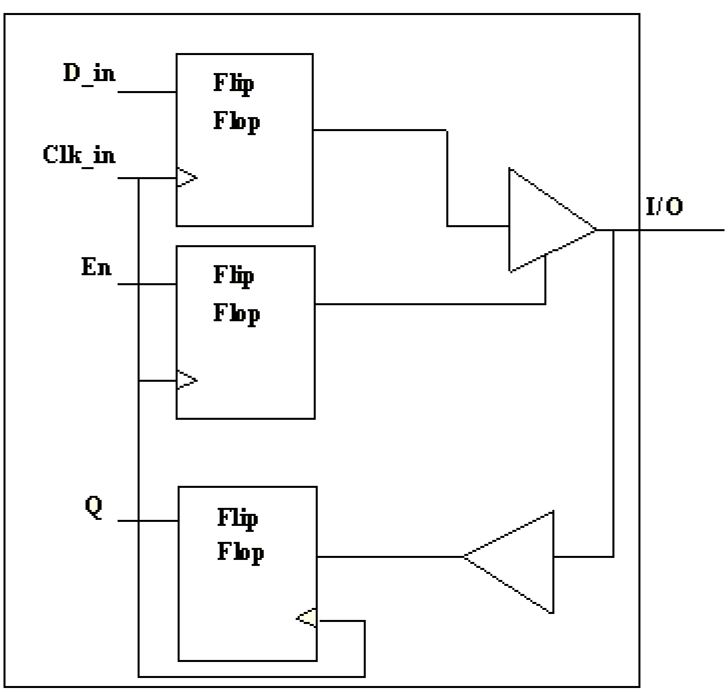
SigXplorer helps you to prototype and design high-speed bus topologies. You can create target bus topologies with multiple drops where each drop, in turn, takes and releases control of the bus based on the stimuli you define for each IOCell on the bus. This allows you to see true dynamic effects over several cycles that include bus turn-around, data-dependent noise effects, and inter-symbol interference.

The following features enable you to support high speed buses:

- Clocked IOCell MacroModels with integrated edge-triggered D-flip flops driving the IO buffers, as shown in Figure 5-1.
- Custom stimulus definition with the IOCell Stimulus Editor, so you can specify excitation of clock, data, and enable input pins of Clocked IOCell MacroModels.
- Custom measurement of setup, hold, and noise margins.
- Simulation waveform viewing in SigWave's timing-diagram mode.

In combination with coupled traces, you can explore the effects of neighbor nets through crosstalk and reflection simulations.

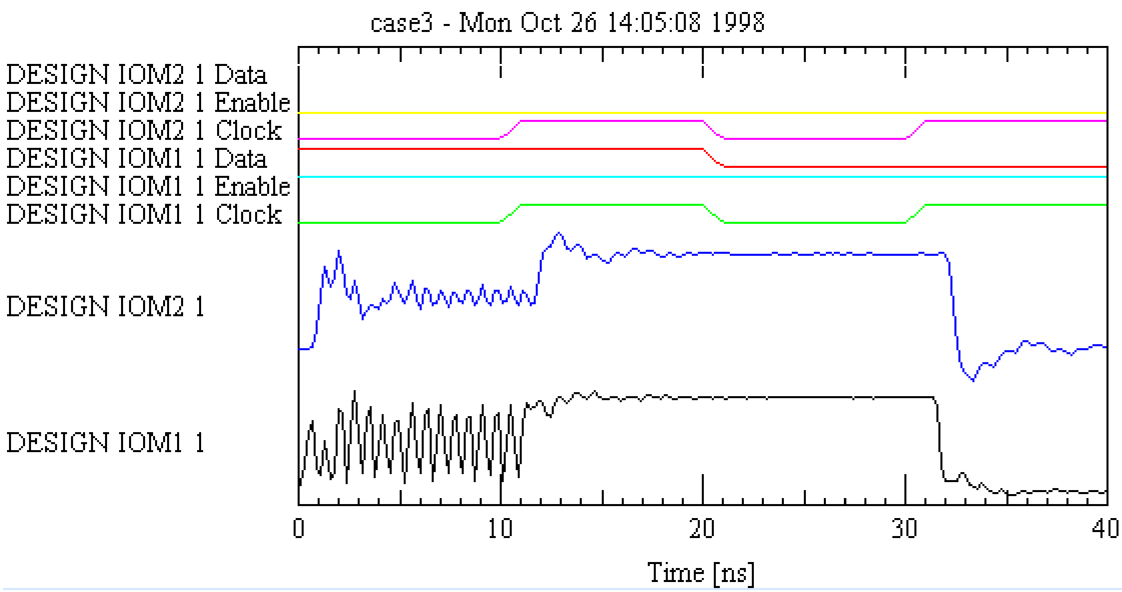
Figure 5-1 Internal View of IOCell Clocked MacroModel



Timing Diagram Display in SigWave

SigWave displays data, and enable and clock signals, in a stacked configuration, for a driver and a receiver in bus mode, as seen in Figure 5-2.

Figure 5-2 Timing Diagram Display in SigWave



Adding a Clocked IOCell MacroModel

To add a clocked IOCell macro model, do the following:

1. Choose *Edit – Add Element*.

   The Add Element Browser is displayed.

2. From the *Model Type Filter* list, choose *IbisIO*.

3. Select the desired IOCell model and drag it to the Topology Canvas for placement. For example, *CDSDefaultIO_CLK* from the Standard Cadence Library.

4. Click *OK*.

Editing a Clocked IOCell MacroModel

For the driver in a differential pair, you can edit the following attributes:

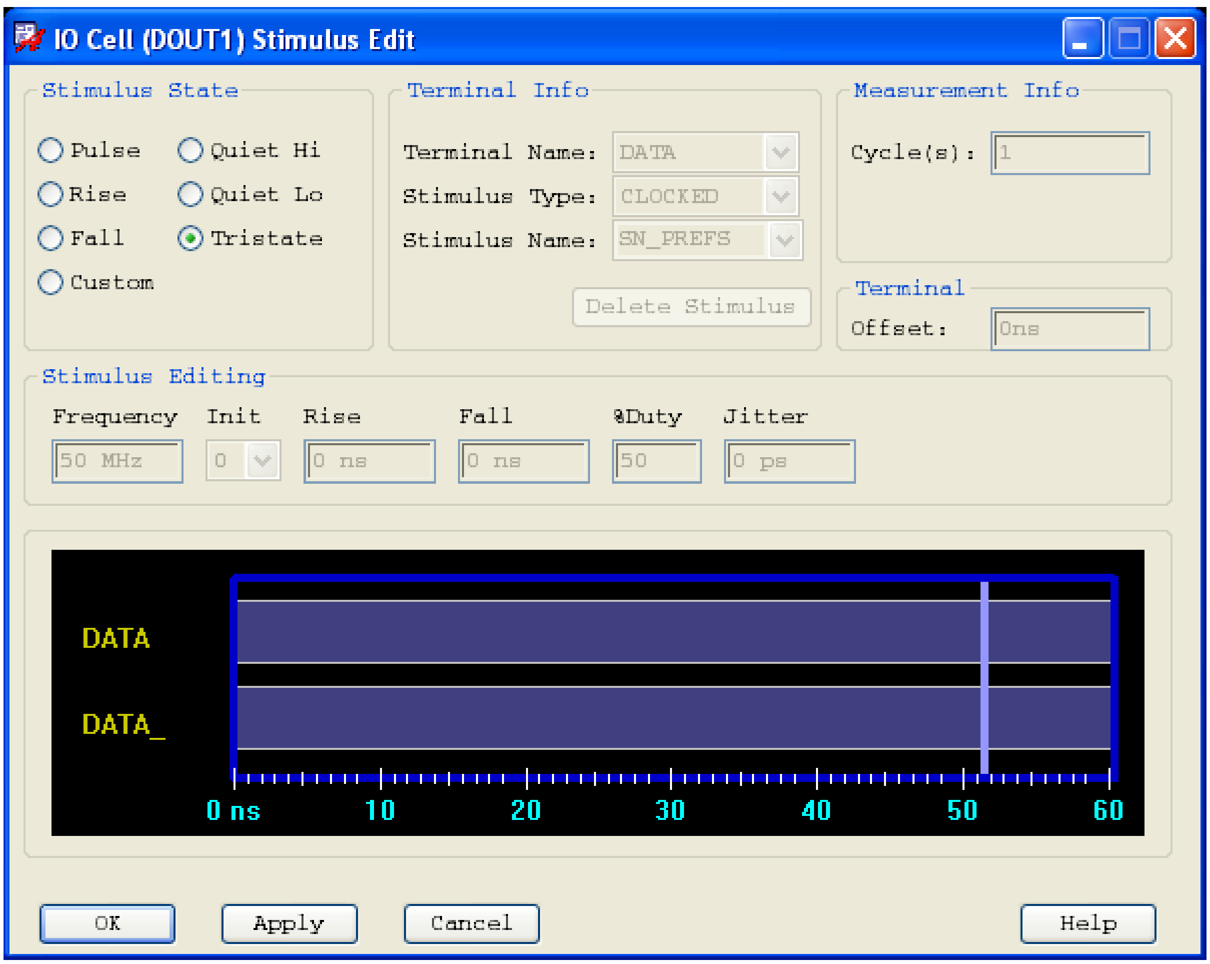
- RefDes (or part name)
- Setup and hold times and sample period
- Stimulus parameters for clock, data, and enable signals

To Modify Stimulus Parameters

1. In the canvas, click the stimulus associated with the IbisIO part symbol. For example, the stimulus might be Pulse or Tristate.

   The IOCell Stimulus Editor opens for the IbisIO with the current stimulus data displayed in the data fields.

Figure 5-3 IO Cell Stimulus Edit Dialog Box

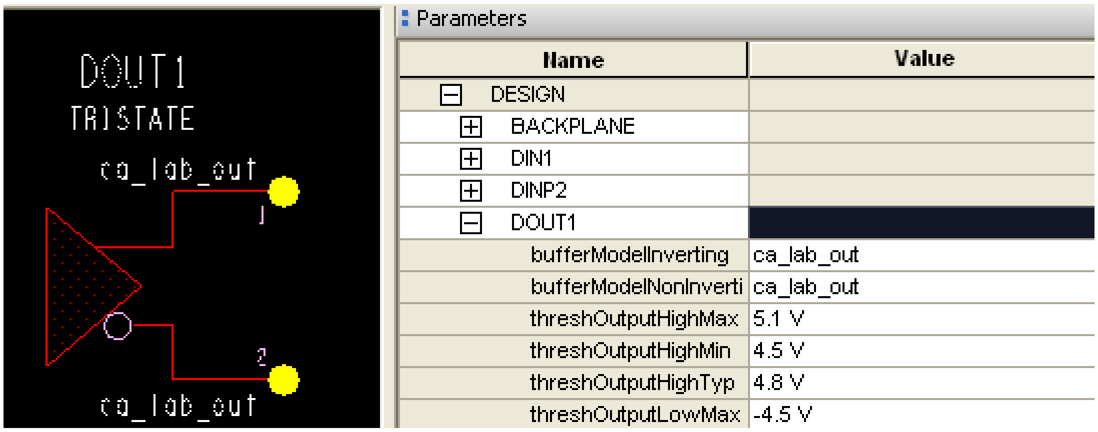


2. In the IOCell Stimulus Editor, make the appropriate edits to the clock, data, and enable signals in the Stimulus Editing section of the dialog box.

3. Click *Apply* or *OK*.

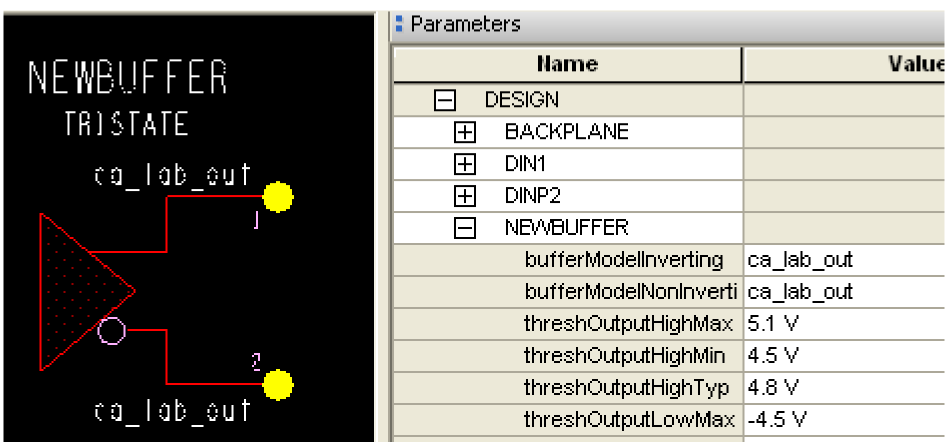
To Modify the RefDes Associated with the IbisIO

1. In the canvas, click the RefDes, or part name, associated with IbisIO symbol.



The *Parameters* tab opens with the data for the selected IbisIO expanded and the RefDes in the Name column highlighted for editing.

2. Enter the new RefDes and click *Enter*.



The new RefDes replaces the old in both the Name column of the spreadsheet and with the part symbol in the canvas.

To Modify Setup and Hold Times and the Sample Period

1. In the canvas, click the stimulus associated with the IbisIO part symbol. For example, the stimulus might be Pulse or Tristate. The IOCell Stimulus Editor opens for the IbisIO with the current stimulus data displayed in the data fields.
2. In the Measurement Info area of the IOCell Stimulus Editor, edit the Setup and Hold times and the Measurement Cycle.
3. Click *Apply* or *OK*.

   See Device Modeling on page 181 for more information.

Simulating a Clocked IOCell MacroModel

➤ Choose *Analyze – Simulate* to start the simulation.

During the simulation, messages display in the Command tab. When the simulation is complete, the Results tab displays the simulation result data. The SigWave window opens to display the differential waveforms.

6

Source Synchronous Interface

Topics in this chapter include

Introduction

When you work with source synchronous bus applications, you need to have more control over measurement data than that supplied by the IOCell Stimulus Editor (Common Clock). Using the Stimulus Editor, you set the cycle of the data signal on which to take measurements. For source synchronous timing applications, you need to be able to take custom measurements on the data signals during a specified cycle of an associated strobe signal.

In order to control measurements in this way you need to be able to:

- Identify a strobe, or reference, signal, and mark a strobe pin on the signal.
- Mark receivers as data pins and group the data pins with the strobe pin.
- Determine the correct time point on the strobe signal to use as the basis for measurement of the data pins.
- Make the indicated custom measurements on the data pins relative to the strobe pins at the appropriate time.

Source synchronous timing measurements are available only for use with custom measurements. All standard measurements still base the measurements cycle on the data signal itself.

Understanding Source Synchronous Custom Measurements

When a topology includes at least one strobe pin, it is considered a source synchronous topology and custom measurements are taken for each data pin relative to the strobe pin associated with it. You overlay the waveform for each data pin with the waveform of its associated strobe pin. Timing windows are set for the two waveforms based on the cycles of the strobe pin, and measurements are taken for the data pin.

For example, receiver `U2.1` is the data pin and `U2.3` is the strobe pin. The measurement cycle is set as cycle 3 in the IOCell Stimulus Editor. Since a strobe pin exists in the topology, the topology is a source synchronous topology and measurements for the data pins are taken relative to the strobe pin.

For strobe pin `U2.3` and data pin `U2.1`, measurements are taken at data receiver `U2.1` using the third cycle of strobe pin `U2.3` as the baseline.

With measurements taken at a voltage threshold of 2.5V for both active edges of the third cycle of the strobe signal, the markers shown are used, since they mark the rising and falling edges of the strobe pin's third cycle.

For the rising edge, the setup measurement starts with the marker at `23ns` (at the strobe pin's rising edge). Look for the data pin's previous transition through `2.5V`, which happens at about `15ns`. This produces a setup margin of 23 – 15 = 8ns

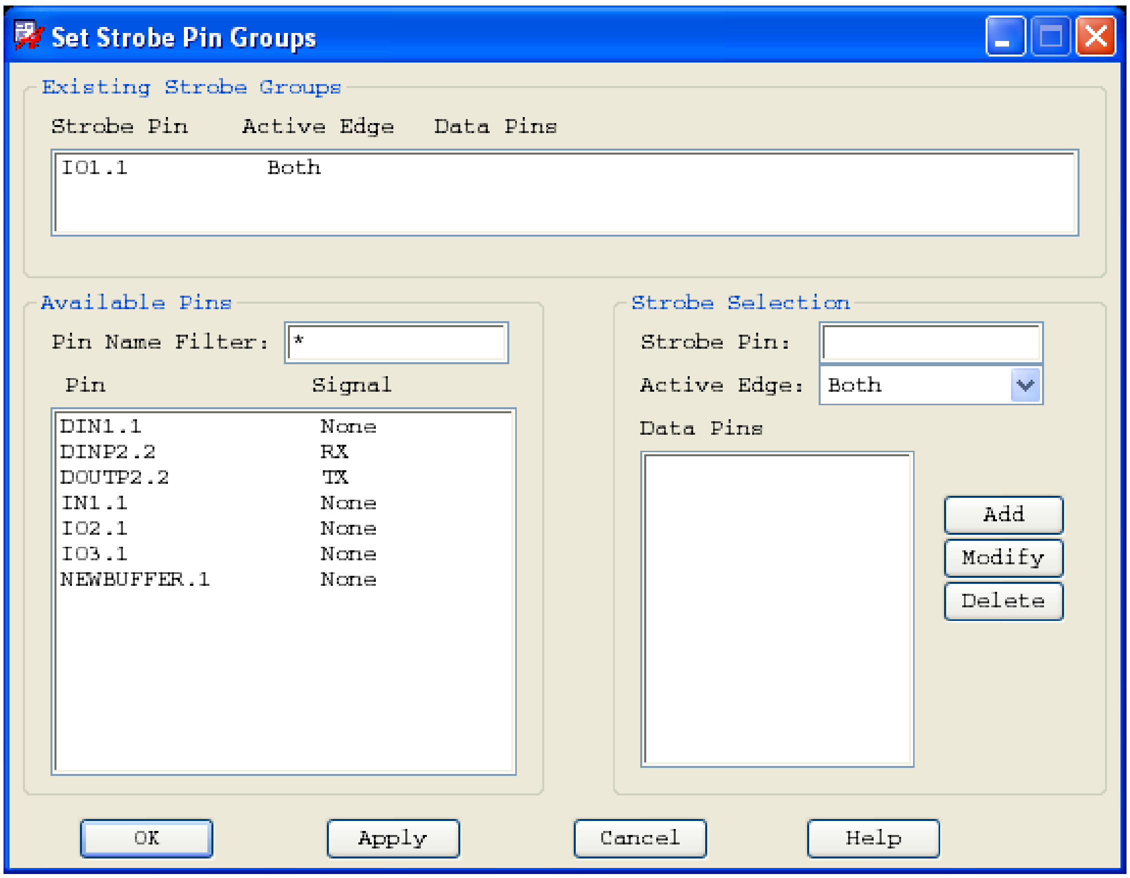
The hold margin measures the difference between the `23ns` marker (at the strobe pin's rising edge) and the data pin's next transition through `2.5V`, which happens at about 27.5ns. This produces a hold margin of 27.5 - 23 = 4.5ns.

Similarly, falling edge measurements key off the marker at `28ns` and produce setup and hold margins of `1` and `12ns`, respectively.

Marking Strobe and Data Pins

Choose *Setup – Strobe Pins* to display the Set Strobe Pin Groups dialog box where you mark and group strobe and data pins and enable the appropriate source synchronous timing measurements. When the Set Strobe Pin Groups dialog box opens, any strobe pin groups that currently exist display in the *Existing Strobe Groups* list box.

Figure 6-1 The Set Strobe Pin Groups Dialog Box



When you open the dialog box, all data editing fields in the *Strobe Selection* area are empty. The *Available Pins* list box displays by pin name all pins in the topology that are not currently marked as strobe or data pins.

See *Allegro SI SigXplorer Reference* for detailed information on the Set Strobe Pin Groups dialog box.

Creating a New Strobe Pin Group

When you create a new strobe pin group, you select pins from the *Available Pins* list box and mark them as, first, the strobe pin, followed by one or more data pins for the strobe pin group. The *Available Pins* list box displays the signal associated with each available pin, or *None*, if there is no signal associated with the pin. A pin's signal is a property of the IBIS Device simulation model associated with the pin.

1. Select a pin in the *Available Pins* list. When there are no pins displayed in the *Strobe Selection* area, the first pin selected from the *Available Pins* list box is marked as a strobe pin.

   As you select pins, they move from the *Available Pins* list box to the *Strobe Pin* field and *Data Pins* list box in the *Strobe Selection* area. The pins are also marked as the strobe pin and as data pins.

2. Specify the active edges on the strobe signal when data pin measurements trigger in the *Active Edge* field. You can choose from *Rising*, *Falling*, and *Both*.

   For most source synchronous designs, data measures on both the rising and falling edges.

3. In the *Available Pins* list box, select the data pins. When a strobe pin displays in the *Strobe Selection* area, each additional pin selected from the *Available Pins* list box is marked as a data pin.

   Each selected pin moves from the *Available Pins* list box, to the *Data Pins* list box in the *Strobe Selection* area.

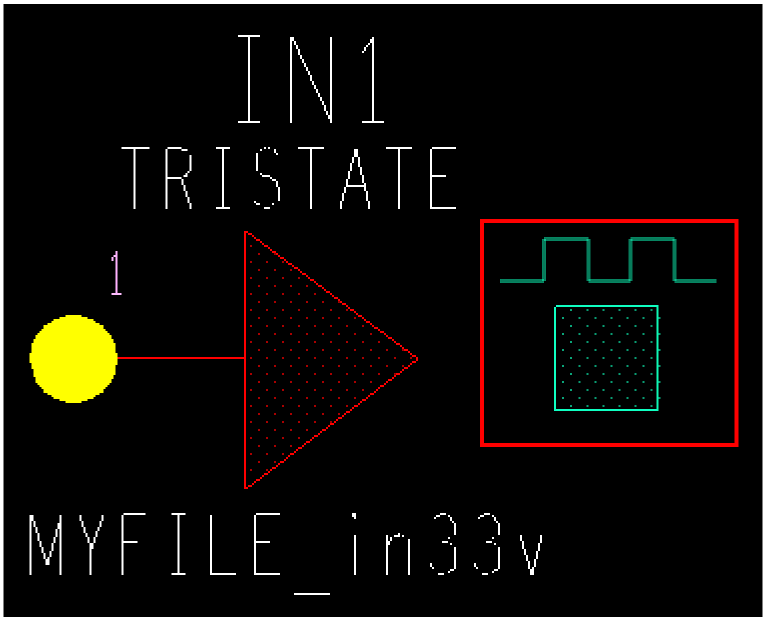
4. Click *Add* to create the strobe pin group when you finish editing.

   The strobe pin group is added to the *Existing Strobe Groups* list box.

5. Choose *OK* or *Apply* to add the strobe pin groups displayed in the *Existing Strobe Groups* list box to the topology.

   The topology is marked as a source synchronous topology and the symbols associated with the Strobe and Data pins change in the canvas to indicate that they are strobe or data pins.

Figure 6-2 IOCell Marked as a Strobe Pin



Editing Existing Strobe Pin Groups

You can modify or delete an existing strobe pin group in the Set Strobe Pin Groups dialog box. To modify an existing strobe pin group:

1. Select a strobe pin group from the *Existing Strobe Groups* list box.

   The names of the strobe pin and the data pins and the active edges data fill the editing fields in the *Strobe Selection* area.

2. Make modifications to the data fields in the *Strobe Selection* area.
3. Click *Modify* to update the strobe pin group.

   The modified pin group is displayed in the *Existing Strobe Groups* list box.

4. Choose *OK* or *Apply*.

To remove an existing strobe pin group:

1. Select a strobe pin group from the *Existing Strobe Groups* list box.

   The names of the strobe pin and the data pins and the active edges data fill the editing fields in the *Strobe Selection* area.

2. Choose *Delete* to remove the strobe pin group from the *Existing Strobe Groups* list box.

The strobe pin group is removed from the *Existing Strobe Groups* list box and appears in the *Available Pins* list box again.

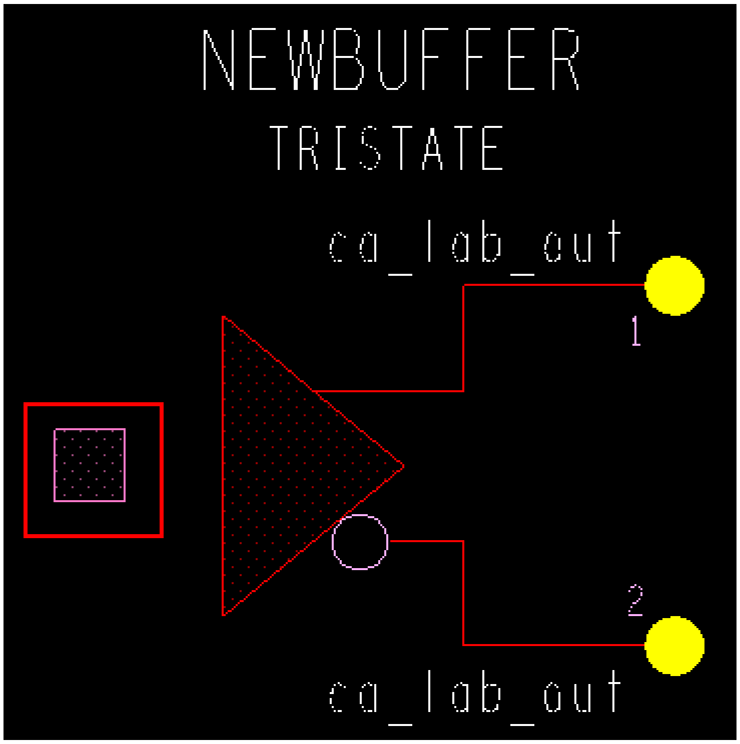
3. Choose *OK* or *Apply*.

Source Synchronous Topology Files

A topology is source synchronous when it contains strobe pins. This flags the topology so that during simulation, custom measurements for pins marked as data pins are taken with respect to the strobe pins waveform rather than its own.

Figure 6-2 on page 98 shows a canvas symbol for an IOCell marked as a strobe pin. Figure 6-3 on page 99 shows a symbol for an IOCell marked as a data pin.

Figure 6-3 IOCell Marked as a Data Pin



Setup and Hold Timing Measurements

For setup and hold timing margin measurements, use the CrossingTime waveform function. This function facilitates taking measurements with reference to a timing threshold by returning the required crossing time before or after the timing threshold.

7

Serial Link interface

Topics in this chapter include:

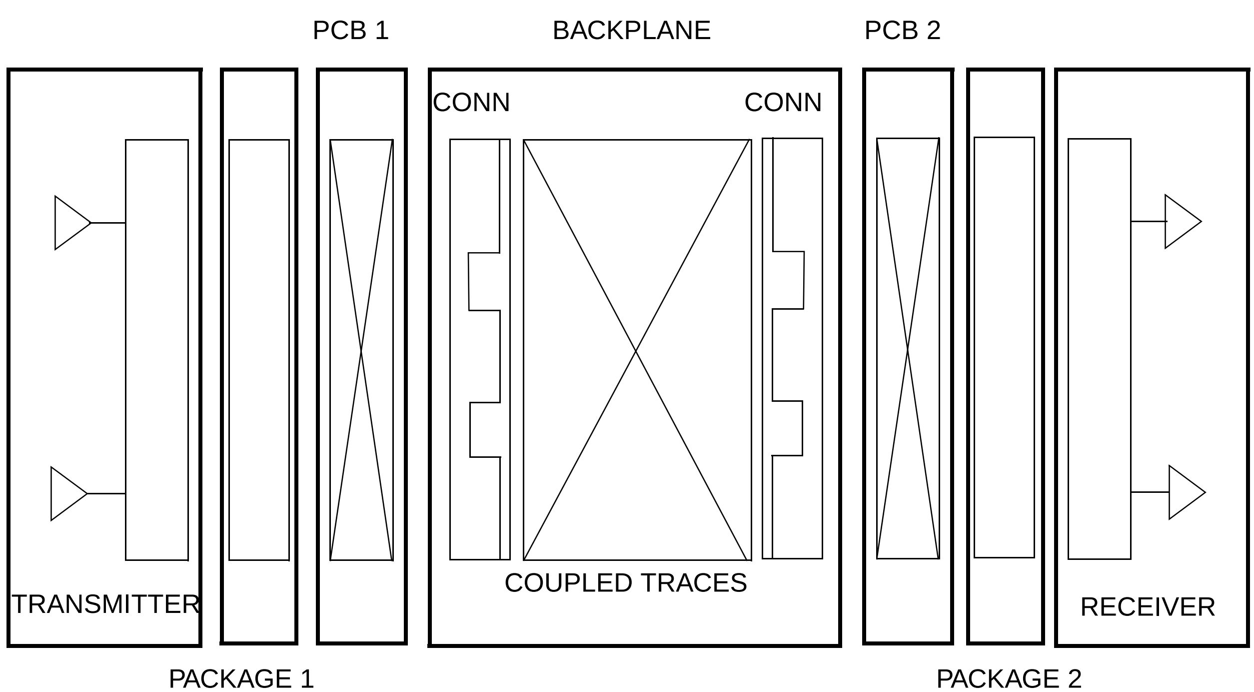
Overview of Channel Analysis -Serial Link Simulation

For serial data running at rates over 1 Gigabit per second (Gbps), the chip-to-chip signal path of the differential pair is often referred to as the *channel*. This channel may be made up of multiple printed circuit boards (PCBs), packages, connectors, and backplanes. The Channel Analysis (CA) functionality in SigXplorer provides the capabilities to aid in the design and analysis of interconnect to support gigahertz data rates.

A channel may be made up of multiple printed circuit boards (PCBs), packages, connectors, and backplanes. Channels can also include coupling from other noise sources. For example, the inclusion of neighboring differential pairs. The end goal is to design a channel whose measured eye pattern seen at the receiver meets its requirements for eye opening and jitter. If these requirements are not met, the data link's integrity may be insufficient to meet the desired bit error rate (BER) criteria for the specific application.

Note: The Channel Analysis feature is available with *Allegro PCB SI GXL* and *Allegro PCB SI Multi-Gigabit Option*.

Figure 7-1 Archetypical Serial Data Channel



Designing this type of high-performance interconnect to support multi-Gigahertz (MGH) data rates presents many challenges. You need to model devices and interconnect in extreme detail to maintain accuracy. Due to the parasitic influence of the channel on the incoming bit stream, you often need to simulate a very large number of bits in order to capture the full effects of inter-symbol interference (ISI) and to accurately simulate the resulting eye pattern. The required number of bits is often far beyond the performance capabilities of traditional circuit simulation. In addition, many MGH drivers utilize pre-emphasis, which can feature multiple programmable taps. Determining the optimum way in which to program the settings for these taps can be a daunting task, and is totally dependent on the channel itself.

With this in mind, Channel Analysis (CA) functionality in SigXplorer provides the following key capabilities to aid in the design and analysis of interconnect to support gigahertz data rates:

- Tap setting optimization

  The ability to determine the optimum settings for a given number of taps to drive a specific interconnect channel.

- Algorithmic Modeling Interface (AMI)

  Supports the introduction of complex algorithms from EDA and IC vendors to the modeling process while protecting intellectual property. AMI works in conjunction with statistical analysis.

- Statistical analysis

  A series of calculations performed to statistically analyze channel interference at various BERs typically for the purpose of estimating total jitter at low bit error rates. You can display the results in bathtub curves or eye contours. This functionality provides an alternative to the simulation option.

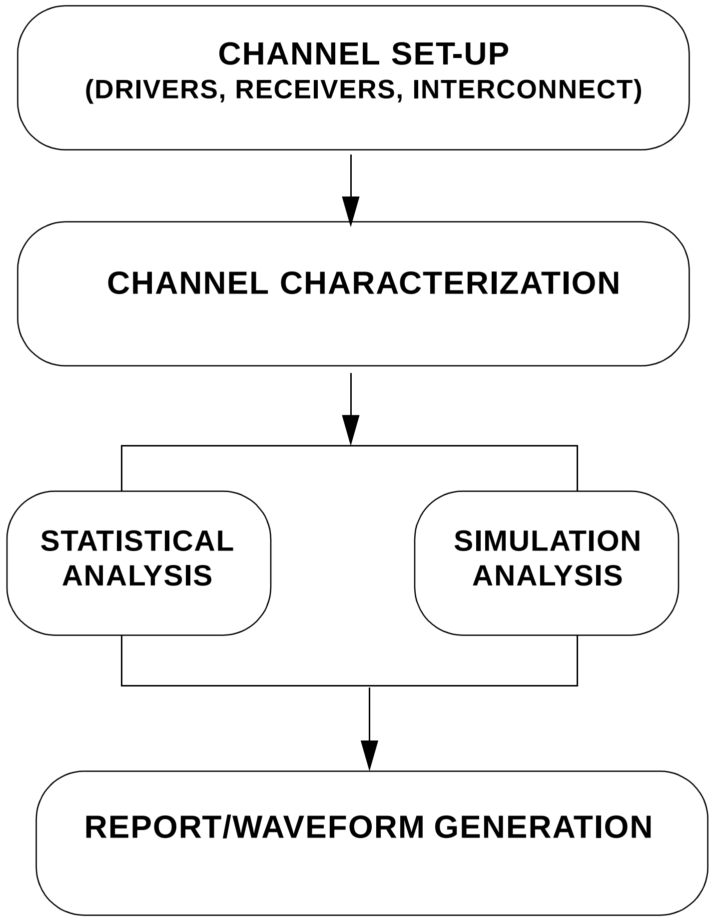
- High capacity simulation

  The ability to simulate extremely large bit streams through the channel to predict realistic eye patterns.

- Support for single-ended driver and receiver models.

Figure 7-2 illustrates the high-level components of the Channel Analysis functionality in SigXplorer.

Figure 7-2 Channel Analysis Work Flow



Prerequisites to Running Channel Analysis

Before running Channel Analysis, you must set up the appropriate topology in SigXplorer, as you would for traditional circuit simulation. Though this procedure assumes differential drivers and receivers, it is the same for single-ended driver and receiver models. If you plan to integrate IP modeling algorithms into Channel Analysis, see Using the Algorithmic Modeling Interface.

1. Identify the node of interest for the primary differential receiver for Channel Analysis.

   You determine this by using the special prefix name `RXin_diff` in a node_param statement in the MacroModel of the receiver. This can be any node in the receiver MacroModel. For example, deep inside the model after an equalization circuit. If this special prefix is *not* found in the MacroModel of the primary receiver, the node of interest for CA is automatically determined as follows:

   - ❑ If a PackageModel or PinParasitics are used, the differential receiver node at the die side of the package (“i” suffix as displayed in SigWave) is taken as the node of interest.
   - ❑ If no PackageModel or PinParasitics are used, the differential receiver node at the pin (no “i” suffix in SigWave, just the pin name) is taken as the node of interest.

Important

You *must* define differential driver and receiver models in an 8-terminal DML MacroModel, example templates of which are available at

```
<CDS_INST_DIR>\share\pcb\signal\templates\diff_8term.dml
```

2. Choose *Analyze – Preferences* in SigXplorer to access the parameters for time domain simulation, such as Waveform Resolution and Cutoff Frequency.

   a. Choose the *Simulation Parameters* tab in the Analysis Preferences dialog box.

   b. Set *Waveform Resolution* for approximately 1/10 of the driver’s rise time, which is often in the 10 picosecond (ps) range for MGH drivers.

   c. For *Cutoff Frequency*, define a non-zero value to enable lossy line simulation. It is recommended that you use 10GHz to cover up to the 50GHz range, which should be sufficient for most MGH applications.

   d. Select the circuit simulator you wish to use from the *Simulator* list.

3. Run a quick time domain circuit simulation and check the resulting eye pattern at the receiver, using a short bit stream. For example, 64 bits. Verify that your models are functioning as expected and producing a realistic voltage swing and eye opening.

Note: If your topology contains multiple receivers, you must identify the primary receiver to determine where you must locate the “node of interest” upon which CA will focus. For additional information, see Incorporating Crosstalk Effects into Channel Analysis on page 132.

4. Once traditional circuit simulation is running properly, you are ready to invoke *Channel Analysis*.

Using the Algorithmic Modeling Interface

Simulating a high speed serial link with driver and receiver from different vendors is a challenging task for designers. The challenge lies in the fact that the models supplied by different vendors are seldom compatible. Also, traditional SPICE-based analysis cannot simulate millions of bits required to accurately predict serial link results.

To address these issues, the IBIS ATM standards committee developed a standard to model high speed serial links, the Algorithmic Modeling Interface (AMI). AMI was designed to characterize pre-emphasis, equalization, and clock recovery in a transceiver. In addition to characterization, AMI aims at delivering fast time domain simulation results on a vendor independent platform just like IBIS.

AMI has been incorporated in the CA work flow and it facilitates the integration of vendor-specific modeling algorithms. AMI is also approved as part of the IBIS 5.0 modeling specification.

This section covers the following topics:

- Using IBIS 5.0
- Using MacroModel
- AMI-Related Enhancements in 16.5

Using IBIS 5.0

Cadence's Signal Integrity (SI) solution provides complete support for native IBIS 5.0 models. You can use an IBIS model file for simulation after translating it to a `dml` file format. SI provides you with the option to translate IBIS models into DML models. See Translating Models on page 150 for more information on translating models.

1. Choose *Analyze – Model Browser* to launch SI Model Browser.
2. Select the *IBIS Models* tab.
3. Click *Model Editor* to view the AMI section of the IBIS file.

Figure 7-3 AMI section of the IBIS file

```
      1.32000599E-9                  NA      1.14281301E-3                  NA
      1.80000825E-9   -307.76084168E-6   -395.37667180E-6   -245.00905420E-6
|
[Algorithmic Model]
Executable Linux_gcc3.3_32 V5_GTP_AMI_Tx.so V5_GTP_AMI_Tx.ami
[End Algorithmic Model]

|***************************************************************************
|                           MODEL DQ_HALF (Reduced-Strength IO Driver)
|***************************************************************************
```

Notice the Algorithmic Model section represented by `[Algorithmic Model]`, `[End Algorithmic Model]`. Three entries that follow the executable sub parameter are:

`Platform_Compiler_Bits File_Name Parameter_File`

Table 7-1 AMI section in the IBIS file

AMI Argument	Description
`[Algorithmic Model]`, `[End Algorithmic Model]`	Begins and ends an Algorithmic Model section, respectively
`Platform_Compiler_Bits`	Includes a string without white spaces, consisting of three fields separated by an underscore: ■ Name of the operating system followed by its version, compiler and its version, and the number of bits for which the shared object library is compiled. ■ The second field consists of the name of the compiler followed optionally by its version. ■ The third field is an integer indicating the platform architecture. **Note:** If the version of either the operating system or the compiler contains an underscore, it must be converted to a hyphen (-) to ensure that underscore is only present as a separation character in the entry.
`File_Name`	Provides the name of the shared library file (vendor executable). The shared object library should be in the same directory as the IBIS (`.ibs`) file.

AMI Argument	Description
`Parameter_File`	Provides the name of the parameter file (`.ami`). This is an external file, which resides in the same directory as the `.ibs` file and the shared object library file. The `.ami` file consists of reserved and model-specific (user-defined) parameters for use by the EDA tool and for passing parameter values to the model. (Figure 7-5 on page 110)

4. Close Model Editor.
5. In the SI Model Browser, select the IBIS model file from the *IBIS File Name list*.
6. Click *Translate*.

   The IBIS models are translated into dml models.
7. Click one model in the IBIS Models tab and then click the *DMI Models* tab.

   The same (translated) model appears selected in the DML tab.
8. Click *Model Editor* to view the translated model.
9. In Model Editor, scroll down to the *ami* section and view the details.

   Notice that in the translated file (`devices.dml`), the name of the DLL, the location, and the parameters for the model are present. These parameters are originally stored in the input file (`.ami`).

Figure 7-4 An excerpt from the ami section of the devices.dml file

```
(ami
 (V5_GTP_AMI_Tx
  (Description "Xilinx Virtex-5 GTP transmitter model provided by SiSoft" )
  (Model_Specific
   (Process 0 )
   (Tx_Equalization 0 )
   (Tx_Strength 0 ) )
  (Path "/home/soniap/junk/xilinx_v5_gtp" )
  (Reserved_Parameter
   (GetWave_Exists True )
   (Ignore_Bits 2 )
   (Init_Returns_Impulse True )
   (Max_Init_Aggressors 25 )
   (Use_Init_Output False ) )
  (amiPlatformInfo
   (Linux
    (Bits 32 )
    (Complier "gcc3.3" ) ) ) ) )
```

Important

For each parameter, only the default values are shown in the `devices.dml` file. You need to edit the `devices.dml` file to change the values of the parameters when running simulations.

Figure 7-5 An excerpt from the input ami file

```
(Model_Specific
   (Tx_Strength (Usage In)(Format List 0 1 2 3 4 5 6 7)(Type Integer)(Default 0)(Description "Differential Swing")
      (Label "000: 1100mV" "001: 1050mV" "010: 1000mV" "011: 900mV"
             "100: 800mV" "101: 600mV" "110: 400mV" "111: 0mV"))
   (Tx_Equalization (Usage In)(Format List 0 1 2 3 4 5 6 7)(Type Integer)(Default 0)(Description "Pre-Emphasis")
      (Label "000: 0%" "001: 3%" "010: 4%" "011: 10.5%"
             "100: 18.5%" "101: 28%" "110: 39%" "111: 52%"))
   (Process (Corner 0 -1 1)(Usage In)(Type Integer)(Default 0)(Description "Process Corner")
      (Label  "0: tt" "-1: ss" "1: ff"))
) | End Model_Specific
```

10. Close Model Editor.

11. Choose *Analyze – Preference*.

Important

Ensure that you choose *EMS2D* as the preferred Solver.

12. Click *Simulate*.

You can now run *Channel Analysis* and view the waveform in SigWave. You can edit values of parameters in the `devices.dml` file to try out different variations.

Using MacroModel

The modeling algorithm files provided to you by a model developer are DLL files. To implement the functionality:

1. Copy the DLLs (there can be more than one) into a directory in the CA search path.
2. Edit the DML files you are using to include an AMI section that references the DLLs, as shown and described below.

```
(IbisIOCell
.
.
.
     (pci_xp_in
     .
     .
```

```
.
(ami
(chffefilt
(params
    (forcepulse) (pulsein pin.txt) (pulseout pout.txt) ) )
(chcdr ) )
```

Note: In the DML excerpt shown above, the keyword `ami` is used to identify the modeling algorithms in the *`signal_optlib_dir`* path defined in the *Path - Signoise* section of the Allegro User Preferences Editor in Allegro PCB SI.

Two DLLs are referenced, `chffefilt` and `chcdr`. For the first DLL, `chffefilt`, the `params` keyword describes a set of specific parameters passed to the DLL as a character string in the dllcontrols argument of the ami initialization call.

3. Run *Channel Analysis*.

AMI-Related Enhancements in 16.5

With the rapid generation and consumption of Algorithmic Model Interface (AMI) models, it has become imperative to use and consume the AMI parameters associated with these models in a visually appealing manner.

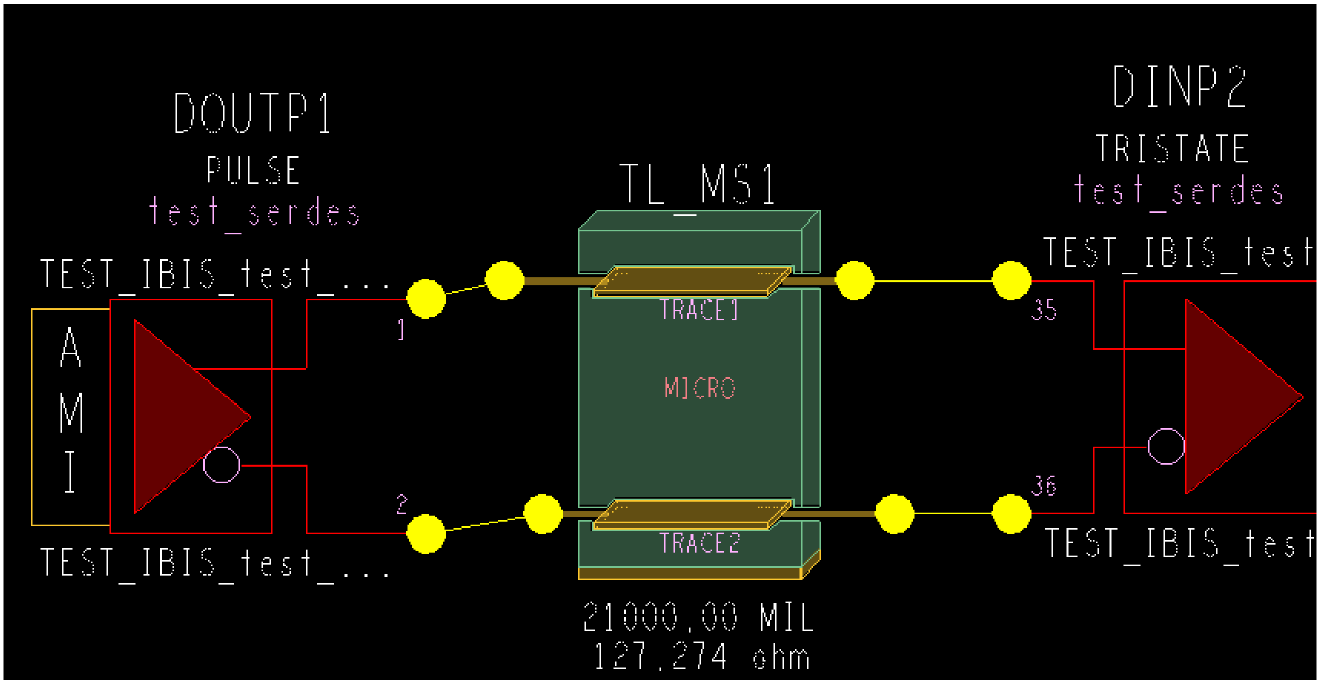
The following enhancements have been made to the Channel Analysis flow for AMI parameters consumption:

- Visually Displayed AMI Model
- Visually Displayed AMI Parameters
- AMI Parameters Information in Devices.dml

Note: AMI parameters are only supported for IBIS Diffpair Devices. If other I/O parts are present, *Channel Analysis* is performed after displaying an information message.

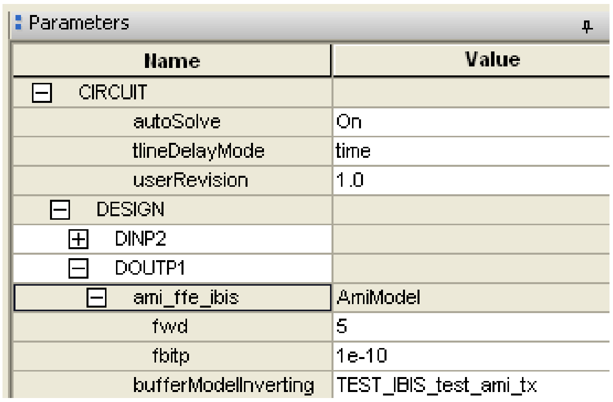
Visually Displayed AMI Model

SigXplorer visually displays a (packaged) buffer model which has an AMI model attached to it. The following figure shows how a TX buffer model appears if it has an AMI mode associated with it.

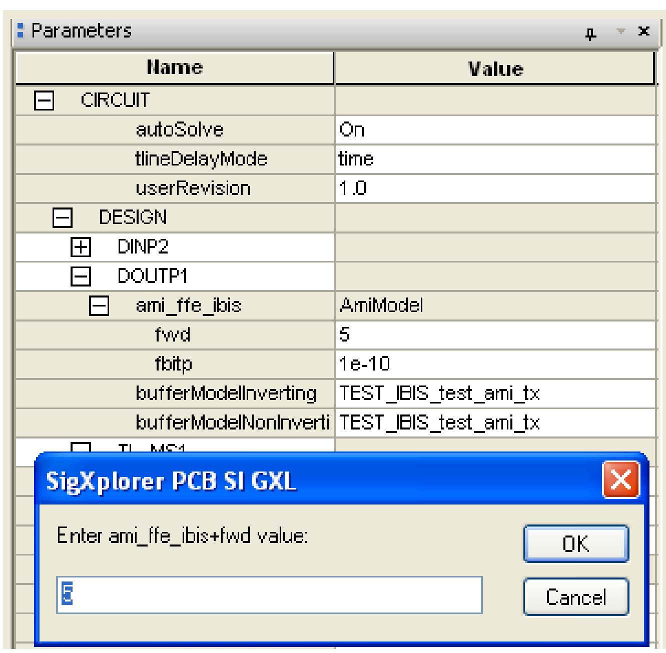


Visually Displayed AMI Parameters

When you click the graphical representation of the AMI model on the canvas, the Parameters window is populated with the model-specific parameters associated with that AMI model.



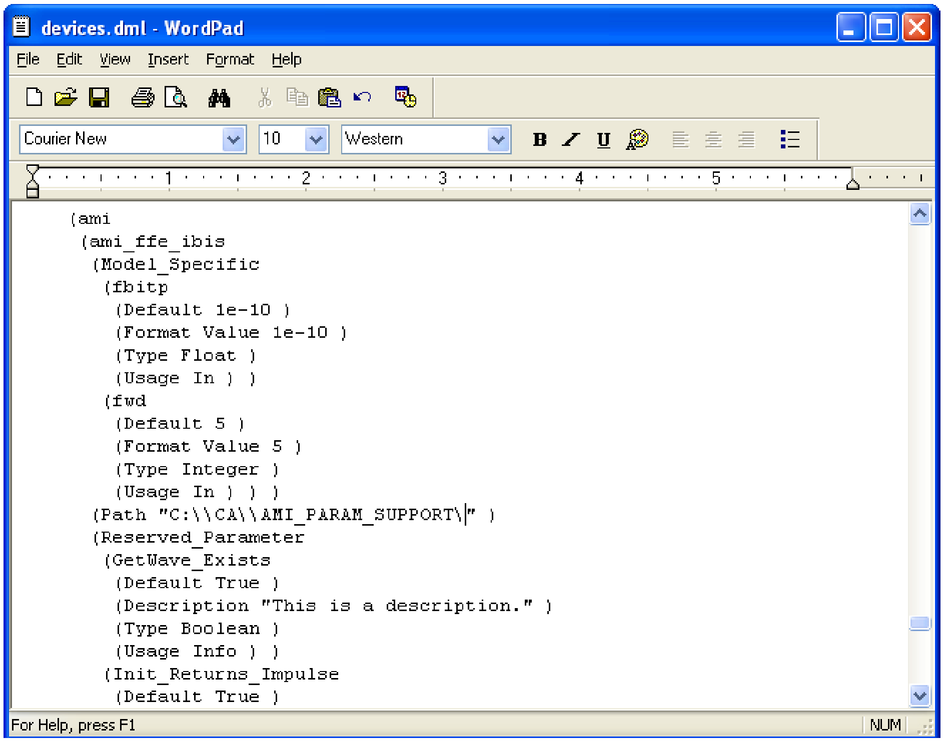
You can also edit any AMI parameter in the Parameters window depending on the permissible values.



At the time of loading the topology, the default instance of the parameter-value pairs are saved. The values are updated if you make any changes in the Parameters window. If the topology file (`.top`) file is saved with a modified parameter value, the value is accessed by Channel Analysis at run time.

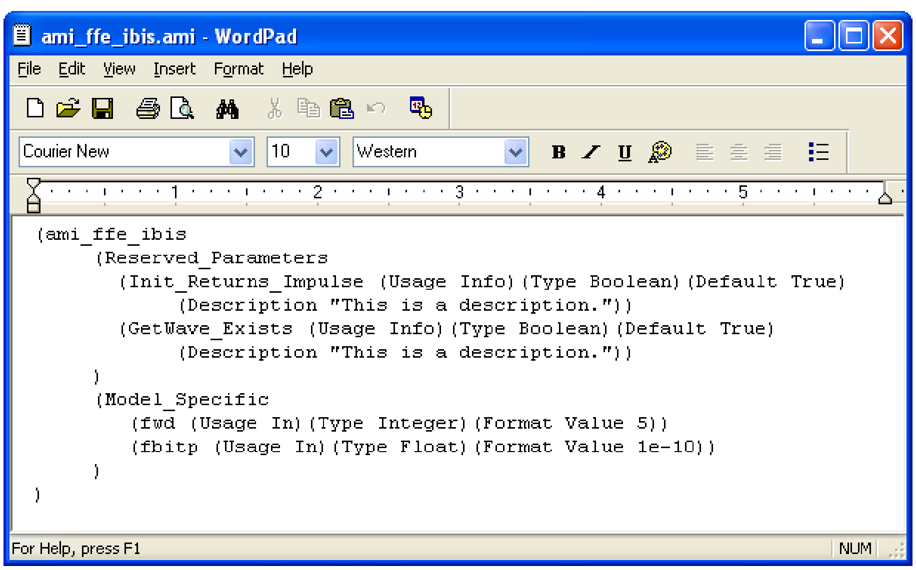
AMI Parameters Information in Devices.dml

For every part on the SigXplorer canvas, the original AMI parameter file is contained within the `device.dml` file.



```
(ami
 (ami_ffe_ibis
  (Model_Specific
   (fbitp
    (Default 1e-10 )
    (Format Value 1e-10 )
    (Type Float )
    (Usage In ) )
   (fwd
    (Default 5 )
    (Format Value 5 )
    (Type Integer )
    (Usage In ) ) )
  (Path "C:\\CA\\AMI_PARAM_SUPPORT\" )
  (Reserved_Parameter
   (GetWave_Exists
    (Default True )
    (Description "This is a description." )
    (Type Boolean )
    (Usage Info ) )
   (Init_Returns_Impulse
    (Default True )
```

The following figure shows a sample AMI file.



```
(ami_ffe_ibis
      (Reserved_Parameters
        (Init_Returns_Impulse (Usage Info)(Type Boolean)(Default True)
             (Description "This is a description."))
        (GetWave_Exists (Usage Info)(Type Boolean)(Default True)
             (Description "This is a description."))
      )
      (Model_Specific
         (fwd (Usage In)(Type Integer)(Format Value 5))
         (fbitp (Usage In)(Type Float)(Format Value 1e-10))
      )
)
```

Running Channel Analysis from SigXplorer

You access the Channel Analysis GUI by choosing *Analyze – Channel Analysis* in SigXplorer. You can also run Channel Analysis in the batch mode from the command prompt. This section describes the controls you set when you run CA in GUI mode.

Important

It is recommended that you run the *Channel Analysis Introductory Tutorial*. The tutorial consists of three lab exercises that help you get familiar with the Channel Analysis functionality.

Channel Analysis Characterization and Simulation

The Channel Analysis graphical user interface is composed of three tabs:

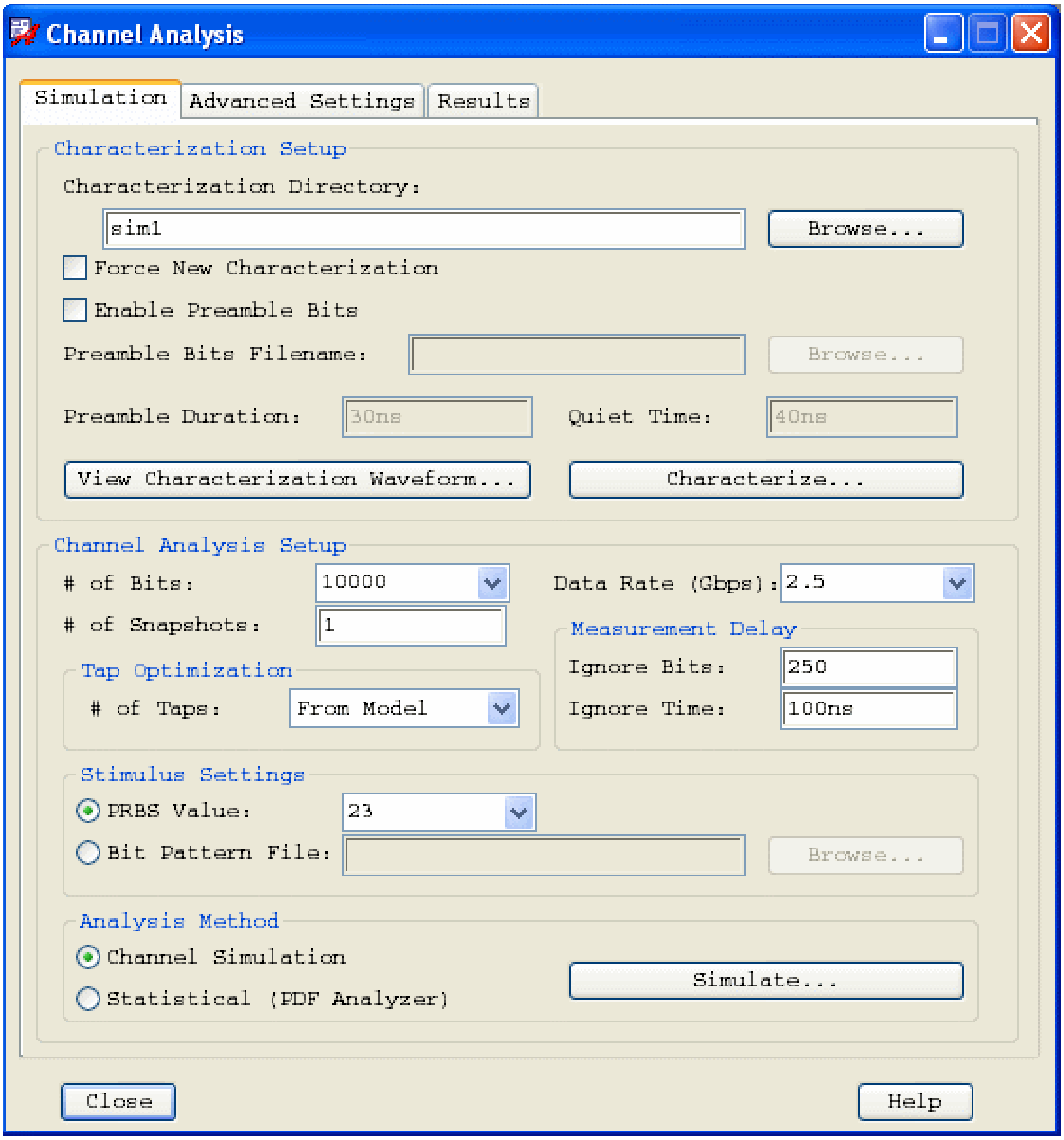
- Simulation
- Advanced Settings Tab
- Results Tab

Simulation

The *Simulation* tab contains the controls that enable you to characterize and simulate your circuit.

Note: The circuit simulator you use for characterization depends on what you select from the *Simulator* list in the *Simulation Parameters* tab of the Analysis Preferences dialog box. This does not apply if you select the statistical analysis option.

Figure 7-6 Channel Analysis GUI Simulation Tab



Characterization Setup

Before you can simulate a circuit, you must perform a characterization of the circuit to get an initial impulse response. You can:

- Define the name for a new characterization
- Reuse an existing one

- Overwrite an existing one

A new characterization is required whenever the physical topology changes. For example, when an interconnect length changes, or when a different package, connector, or driver/receiver model is used.

Important

You need to manually re-characterize when the topology of your circuit is edited because this function is *not* automatically tracked by Channel Analysis.

The controls in the *Characterization Setup* section of the CA user interface are:

Table 7-2 Simulation Tab - Characterization Setup

Option	Description
Characterization Directory	Specifies the name of the characterization you want to create, or the name of an existing characterization you want to use, which will be used as input for the channel simulation. You can type this name directly into the field or click the *Browse* button.
Browse	Opens up a browser so you can select an existing characterization. The characterizations exist as directories, typically under the `sigxp.run\case0\channel.run\<char_name>` directory.
Force New Characterization	Forces a re-characterization. Use this option if you want to keep an existing characterization name after editing the topology and want to rerun the characterization to overwrite the data.
Enable Preamble Bits	When checked, this allows you to specify an initialization bit pattern to be run before characterization occurs. You typically set this option to exercise transistor-level models (that is, to set biasing, charge up capacitances, etc.) preceding characterization response.
Preamble Bits Filename	Specifies the bit file for use before the characterization response is run. The format of this file is identical to the CA bit pattern file.
Preamble Duration	Specifies the amount of time during which the pattern of the preamble bit pattern in the preamble bit pattern file is repeated. The default value is 30ns.

Option	Description
Quiet Time	Specifies the gap time between the end of the preamble and the start of characterization. The default value is 40ns.
View Characterization Waveform	Brings up in SigWave the waveform generated by the characterization. If only one characterization file resides in the characterization directory, that file will be displayed. If there is no file in the directory, or if there are multiple files, a file browser is displayed from which you can select a simulation file for viewing.
Characterize	Generates or loads the desired characterization. **Note:** It is not absolutely necessary for you to characterize before you simulate. The software will check to determine if a characterization exists and, if it does not exist, will characterize it before simulating.

Channel Analysis Setup

You control the simulation parameters that will be used in your analysis by defining the data rate, the number of bits to simulate through your channel, the number of snapshots you want to output, and the crosstalk mode you want to assume.

The controls in the *Channel Analysis Setup* section of the CA user interface are:

Table 7-3 Simulation Tab - Channel Analysis Setup

Option	Description
Data Rate	The data rate at which the input bit stream will be generated, specified in Gbps.
# of Bits	Specifies the length of the input bit stream. This option is inactive when you choose the Statistical analysis method.
# of Snapshots	Specifies how many eye contours (graphical eye pattern plots) you want to generate. If you specify more than the default value of 1, eye contours will be spaced evenly over the duration of the simulation. This option is inactive when you choose the Statistical analysis method.

Tap Optimization

You can perform what-if analysis based on the number of taps assumed in the driver's pre-emphasis circuit.

Table 7-4 Simulation Tab - Tap Optimization

Option	Description
# of Taps	Specifies how many taps to assume for pre-emphasis what-if optimization. If you select the default mode *From Model*, no tap optimization is performed. This means that only what is already defined in the driver model itself is used and no additional what-if taps are included in the analysis. If pre-emphasis effects are already included in your driver model, leave the *# of Taps* setting as *From Model*. This means that the pre-emphasis effects were already captured during the characterization step, and channel simulation will simulate the specified stimulus through the given channel. Alternatively, if pre-emphasis effects are *not* included, or are turned off in your driver model during characterization, channel simulation can run what-if scenarios regarding the number of pre-emphasis taps in the driver model, optimizing their settings. When a number of taps is specified, channel simulation examines the characterization, synthesizes the optimum settings for these taps to overcome the channel loss, and uses these settings in the simulation. This allows you to see the what-if effects of pre-emphasis on their signals, and also to obtain the optimum tap settings. These settings are included in the report that is generated for the simulation. You can use these to guide tap settings in detailed DML MacroModels or transistor-level models for subsequent verification analyses.

Measurement Delay

Table 7-5 Simulation Tab - Measurement Delay

Option	Description
Ignore Bits	Specifies the number of bits that you want ignored before observing simulation output for eye diagram and/or statistical processing.

Ignore Time	Specifies the length of the input bit stream. This option is inactive when you choose the Statistical analysis method.Specifies the amount of time that you want ignored before observing simulation output for eye diagram and/or statistical processing.
	
	The relationship between *Ignore Bits* and *Ignore Time* values are calculated in the following manner:
	
	`Ignore Time = Ignore Bits/Data Rate`
	–or–
	`Ignore Bits = Ignore Time x Data Rate`
	
	The value of *Ignore Bits* cannot be larger than the value of *# of Bits* or less than zero (0). If it is, a warning is generated.
	
	The measurement delay options are inactive when you choose the Statistical analysis method.

Stimulus Settings

To control the specifics on how the input bit stream is derived, you set parameters for pseudo-random bit sequence (PRBS) value and whether you want to define a specific stimulus pattern through an external file. These options are inactive when you choose the Statistical analysis method.

The controls in the *Stimulus Settings* section of the CA user interface are:

Table 7-6 Simulation Tab - Stimulus Settings

Option	Description
PRBS Value	Specifies the PRBS value to use when the input bit stream is synthesized. The default value is 23, meaning the bit stream is based on a PRBS pattern of 2^23 bits.
Bit Pattern File	Enables pre-defined stimulus to be read directly from a text file, which you can access by using the associated *Browse* button.
Browse (for Filename)	Opens up a browser so you can select an existing stimulus file. The stimulus file is a simple ASCII text file (for example, `mybits.txt`), formatted as follows: (0011011010011..) *Important* The dots (..) at the end of the pattern instructs CA to keep repeating the pattern over and over until the number of bits requested in the *# of Bits* field has been reached. If the ellipses (..) are not included at the end of the sequence, the final bit in the pattern will be repeated until the specified number of bits is reached.

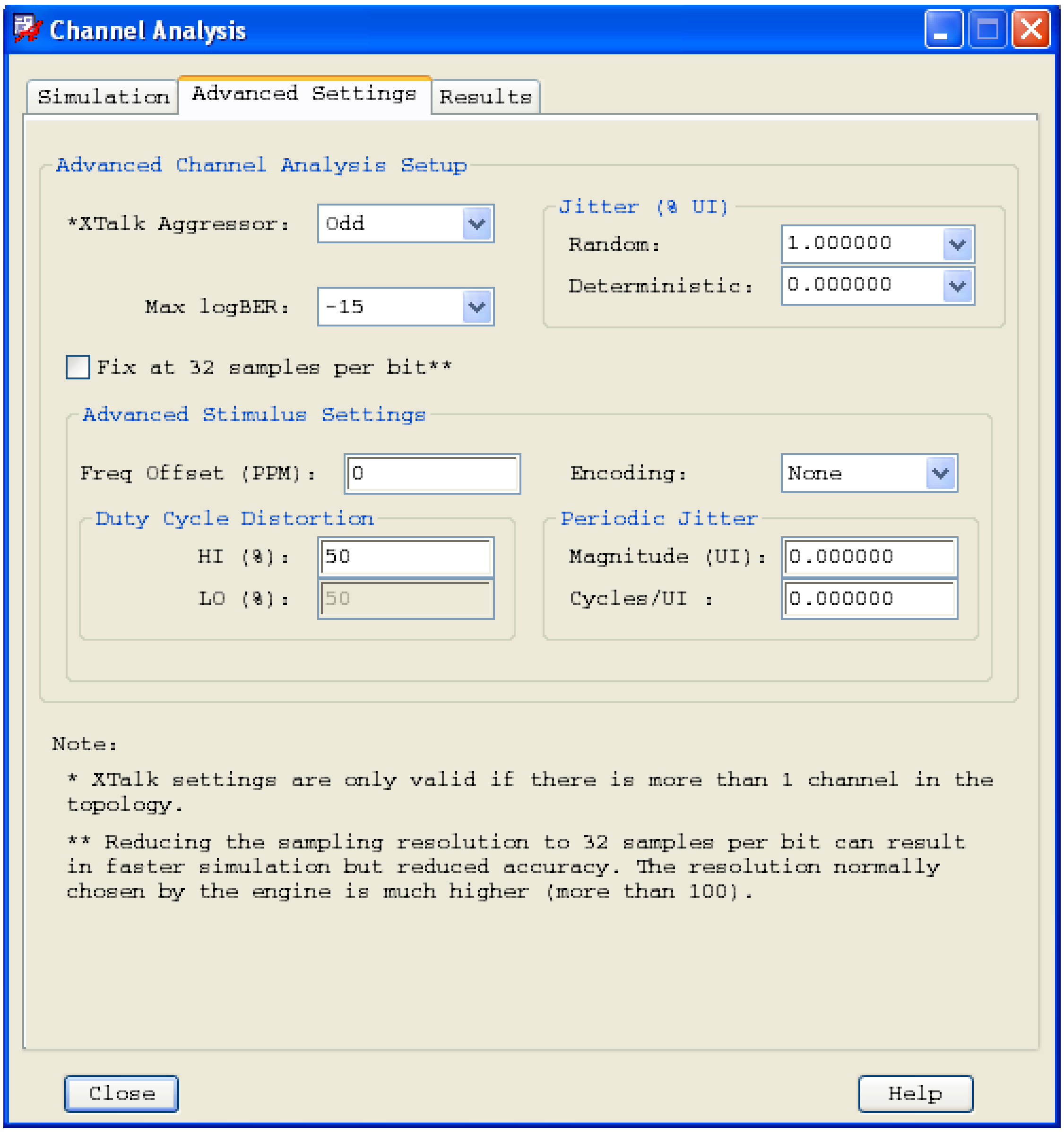
Analysis Method

Table 7-7 Simulation Tab - Analysis Method

Option	Description
Channel Simulation	Specifies that Channel Analysis performs detailed bit-by-bit simulation to generate eye diagram data.
Statistical (PDF Analyzer)	Specifies that statistical methods are used to generate eye diagram data.
Simulate	Executes the channel simulation and takes you to the Results tab.

Advanced Settings Tab

Figure 7-7 Channel Analysis GUI - Advanced Settings Tab



Advanced Channel Analysis Setup Section

Table 7-8 -Advanced Settings Tab - Advanced Channel Analysis Setup

Option	Description
XTalk Aggressor	Determines how drivers are stimulated when more than one differential driver is present. Choices are: ■ *Random* - each driver gets a unique PRBS ■ *Odd* - neighboring drivers get the opposite stimulus of that used for the primary driver ■ *Even* - neighboring drivers get the same stimulus as that used for the primary driver **Note:** If multiple drivers/channels are not present in the topology, the *XTalk Aggressor* setting is ignored. For additional information on crosstalk effects, see Incorporating Crosstalk Effects into Channel Analysis on page 132.
Max logBER	
	value range is -15 to -18.
Fix at 32 samples per bit	When enabled, allows you to boost the simulation speed by reducing the samples-per-bit to 32. If left disabled, the sample bit rate is design-dependant. Activating this option results in faster simulation but decreased accuracy. The default condition is Off. This option is inactive when you choose the Statistical analysis method.

Table 7-9 -Advanced Settings Tab - Jitter (%UI)

Option	Description
Random	Specifies the percent unit interval (UI) of jitter you want to post-process in a Gaussian distribution.
Deterministic	
	value range is 0-25.

Advanced Stimulus Settings Section

Table 7-10 -Advanced Settings Tab - *Advanced Stimulus Settings*

Option	Description
Freq Offset	Lets you specify a frequency offset between the transmitter and the receiver. The value is in parts per million. This option is inactive when you choose the Statistical analysis method.
Encoding	Enables you to specify if `8b10b` encoding should be used. The default setting is *None*. This option is inactive when you choose the Statistical analysis method.

Duty Cycle Distortion

The value you enter here calculates the difference between the bit period of 1 bit and a zero (0) bit. This option is inactive when you choose the Statistical analysis method.

Table 7-11 -Advanced Settings Tab - *Duty Cycle Distortion*

Option	Description
HI (%)	The value for the 1 bit, as a percentage. The default condition is 50%.
LO (%)	This field is read-only.

Periodic Jitter

Table 7-12 -Advanced Settings Tab - *Periodic Jitter*

Option	Description
Magnitude (UI)	Specifies the magnitude in unit intervals of periodic jitter. Periodic jitter is related to the data pattern of the bit pattern file.
Cycles/UI	Specifies the frequency of periodic jitter in unit intervals.

Note: Periodic jitter is inactive when you choose the Statistical analysis method.

Results Tab

The *Results* tab of the Channel Analysis user interface is where you examine the outputs from your channel simulation or statistical analysis run. It is composed of two sections, *View Channel Simulation Results* and *Correlation*. The outputs of channel simulation are located in

`<working_dir>\sigxp.run\case0\channel.run\<char_name>\char\results`

For additional information on output directory structures, see Channel Analysis Directory Structure on page 133.

When you choose the Results tab, the following message pops up prompting you to run CA simulation:

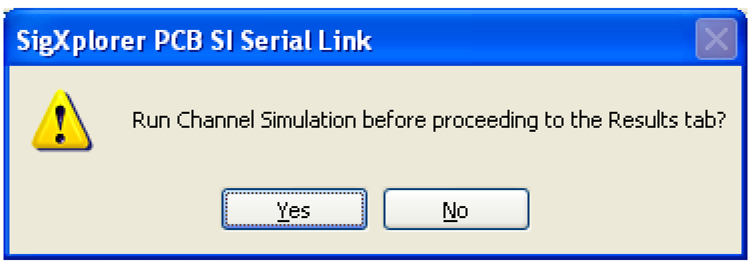
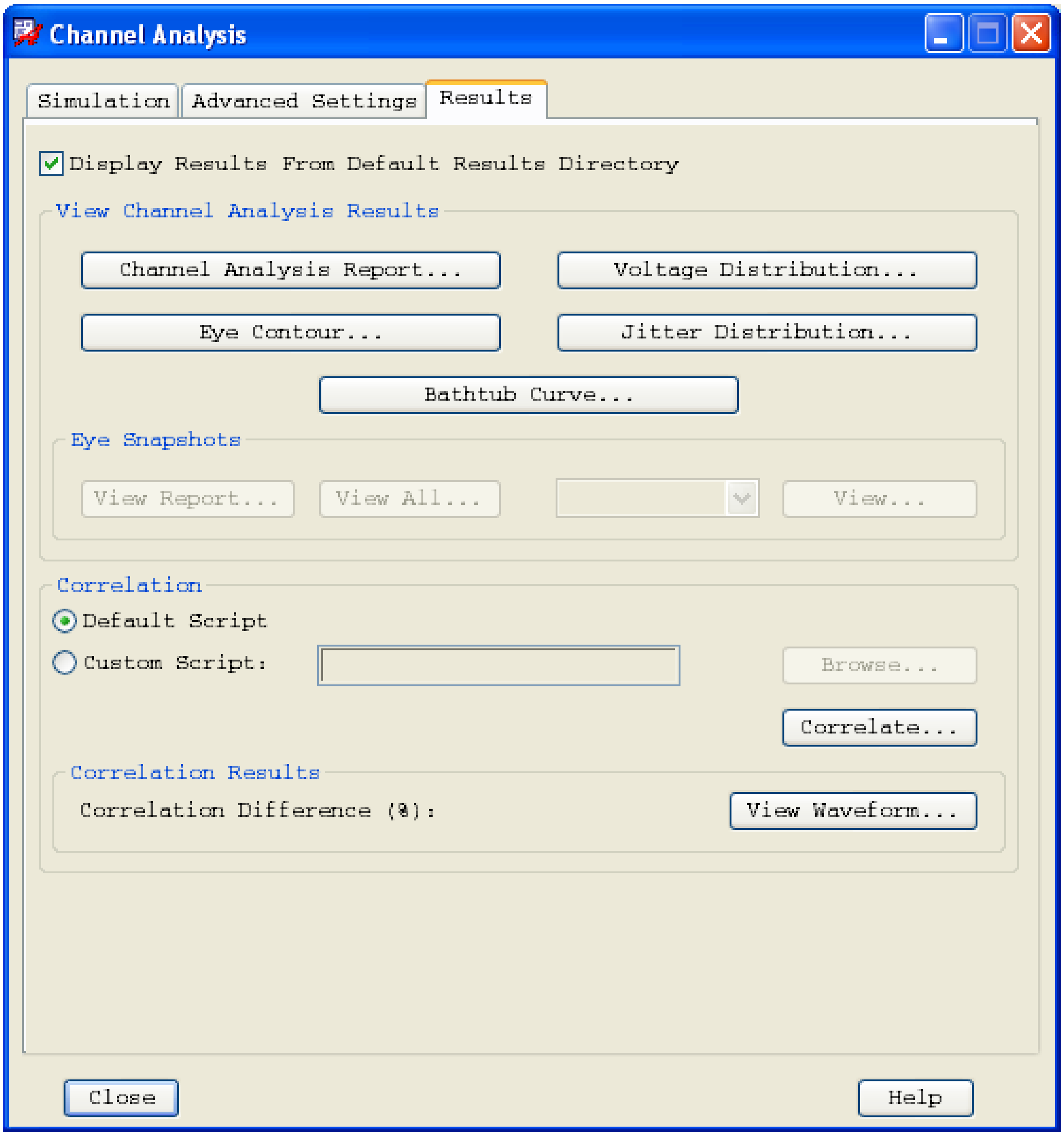


Figure 7-8 Channel Analysis GUI Results Tab



When you select the *Display Results from Default Results Directory* option, the Results Browser automatically displays the results associated with the governing characterization. When this option is left unchecked, browsers are invoked that enable you to select specific outputs to display.

View Channel Analysis Results

This section of the tab page allows you to view the outputs generated when you perform channel analysis. The controls in the *View Channel Simulation Results* section of the CA user interface are:

Table 7-13 -results Tab - View Channel Analysis Results

Option	Description
Channel Analysis Report	Displays the CA report for the current simulation.
Eye Contour	Displays the eye contour (outline of the eye pattern) for the current simulation. The x-axis of the eye contour plot is normalized to the bit period, expressed as a Unit Interval (UI). As an example, for a data rate of 5 Gbps, the bit period is 200ps. Measuring an eye jitter of 0.2 UI in the display corresponds to 0.2 multiplied by 200ps, or 40ps.
Voltage Distribution	Displays the voltage distribution for the current simulation, regarding where specific voltage levels take place. Imagine the voltage display rotated 90 degrees and overlaid on the eye contour. Two sharp, narrow peaks indicate minimal fluctuation of voltage levels in the eye pattern. The x-axis of this plot is normalized to the maximum voltage swing seen in the time delay waveforms, with the 0 point placed in the center of the full swing. The y-axis of this plot is determined in the same manner as the jitter distribution plots.

Option	Description
Jitter Distribution	Displays the jitter distribution for the current simulation, regarding where specific crossings take place at the bias level of the receiver signal. Two sharp, narrow peaks indicate minimal jitter in the eye pattern. The x-axis of the jitter distribution plot, like the eye contour plot, is expressed as a UI, except that the 0 point is placed in the middle of the bit period. The y-axis represents the number of samples that occur in the “slice” of the UI compared with the total number of samples taken overall. As an example, a data point with coordinates of (-0.4, 0.02) means that 2% of the recorded jitter samples occurred in the portion of the UI that is bounded by -0.41 UI and -0.40 UI on the x-axis.
Bathtub Curve	Typically used in conjunction with the statistical analysis option on the *Simulation* tab, plots BER as a function of sampling offset from eye center.

Eye Snapshots

From this section, you can display various *snapshots* taken during your channel simulation. A complete snapshot is a graphical representation of an eye contour that depicts the end results of the channel simulation run, including all the bits in the data stream. You can also take snapshots at various stages of the simulation run, allowing you to track changes to the eye contour as additional bits are simulated.

Note: These options require you to set *# of Snapshots* (on the *Simulation* tab) to a value greater than 1.

Table 7-14 -Results Tab - *Eye Snapshot*

Option	Description
View report	Displays an *Eye Snapshot Report*, a text file that contains the snapshot identification numbers, number of bits simulated, eye height, and eye jitter.
View All	Displays in SigWave the eye contour for all the snapshots.
View	Displays in SigWave the eye contour for the specified snapshot.

Correlation

From this section of the tab page, you can run correlations using scripts. When you perform a correlation, a known (default) stimulus is run through the topology, using both full time domain circuit simulation and channel simulation. The resulting waveforms are automatically displayed as overlays in SigWave for comparison. You can also use your own correlation scripts if you wish. The controls in the *Correlation* section of the CA user interface are:

Table 7-15 -Results Tab - *Correlation*

Option	Description
Default Script	When selected, running correlation invokes the script <CDS_INST_DIR>\share\pcb\chsim\chcorr.lsp This script: ■ Synthesizes a short stimulus bit stream. ■ Simulates this bit stream through the topology using the native circuit simulator. ■ Simulates this bit stream through the topology using channel simulation. ■ Overlays the resulting waveforms for direct comparison in the file `corr.sim`.
Custom Script	When selected, allows you to browse for a customized version of the default script, where you can define a specific bit length or make other modifications. We recommend using a copied and/or edited version of the default *chcorr.lsp* file.
Correlate	Invokes the correlation simulations and analyzes the results.

Correlation results

From this section of the tab page, you can view the percentage difference of the two simulators' correlation results and view the resulting waveforms.

Table 7-16 -Results Tab - *Correlation Results*

Option	Description
Correlation Difference (%)	Displays the percent difference between CA results and that of the native circuit simulator.
View Waveform	Invokes SigWave and displays the relevant `corr.sim` file, which overlays the results from the two simulators. **Note:** Two waveforms are displayed, one of which has the suffix *_ref*. This is the one from the native circuit simulator. For example, *tlsim*. The other waveform is generated from CA.

Procedure

Before running Channel Analysis, you must set up the appropriate topology in SigXplorer, as you would for traditional circuit simulation. See Prerequisites to Running Channel Analysis on page 105 before performing the procedure described below.

1. Using SigXplorer, open a topology. You can either create one on-the-fly or extract an existing topology from Allegro PCB Editor.
2. Choose *Analyze – Channel Analysis*.

   The *Channel Analysis* dialog box appears.
3. In the *Simulation* tab of the dialog box, enter values in the fields for Characterization Setup and Channel Analysis Setup, as described in Channel Analysis Characterization and Simulation on page 115.
4. When you have completed setting the simulation parameters, click *Simulate*.
5. When the run is complete, click *Eye Contour* in the *Results* tab page to open a contour file in SigWave. Once open, you can import additional contour files.

   Note: The default directory from which to import files is `sigxp.run/case0/channel.run/snapshots/results` in your current working directory.
6. Select and view the other outputs generated by Channel Analysis. You can also compare the results of the CA run by correlating the results with a scripted default stimulus, the controls for which are described in Results Tab on page 125.

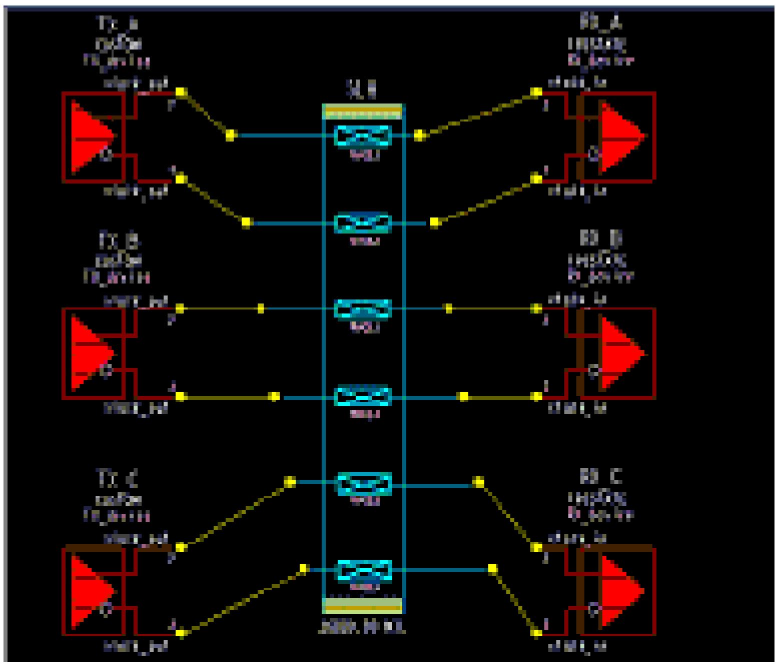
For a quick hands-on experience with the Channel Analysis functionality, see the *Channel Analysis Introductory Tutorial*. The tutorial is divided into the following three lab exercises:

- *Channel Analysis Lab #1: High Capacity Simulation*
- *Channel Analysis Lab #2: Pre-Emphasis Optimization*
- *Channel Analysis Lab #3: Data Rate What-Ifs and Outputs*

Incorporating Crosstalk Effects into Channel Analysis

When multiple drivers and receivers are included in the topology, you can include crosstalk effects into Channel Analysis. An example of this kind of topology is shown in Figure 7-9.

Figure 7-9 Topology with Crosstalk



With this type of multi-receiver topology, it is important that you identify the primary receiver. The easiest way to do this is to initially run a traditional time domain circuit simulation. When this simulation is requested, you are prompted to identify the primary receiver. Your selection determines where to take the node of interest on which Channel Analysis focuses. The determination of the specific node of interest for the primary receiver is governed by the same rules as described in Prerequisites to Running Channel Analysis on page 105.

When you move to the Characterization step, separate and distinct characterizations are automatically run for each unique driver-to-primary receiver combination. In the example shown in Figure 7-9, three distinct characterizations are run and stored in the associated *char* directory. Note that this will take approximately three times longer than when only a single characterization is needed. When you run a channel simulation, all of these characterizations are automatically used to simulate the channel.

You can control the way in which stimulus is handled for the neighboring drivers through the *Crosstalk Mode* setting in the GUI (as described in Channel Analysis Setup on page 118).

Channel Analysis Directory Structure

When you run CA, a `channel.run` directory is automatically created in your `sigxp.run\case0` hierarchy. Output files are derived from your current characterization and are stored accordingly. The outputs of channel simulation are located in:

```
<working_dir>\sigxp.run\case0\channel.run\<char_name>\char
<working_dir>\sigxp.run\case0\channel.run\<char_name>\results
```

Table 7-17 CA Directory Structure

Directory Name	Description
working_dir	Location of topology files
sigxp.run	SigXplorer run directory
case0	1st (and only) case in sigxp.run
channel.run	Directory for Channel Analysis
<char_name>	Specific characterization name
char	Location of characterization
results	Location of Channel Analysis output

Additional characterizations result in additional directories that are created parallel to the `<char_name>` directory.

Note: For a quick hands-on experience with the Channel Analysis functionality, see the *Channel Analysis Introductory Tutorial*.

8

Working with Signal Models and Libraries

About Signal Models

Procuring, developing, and verifying simulation models up front in the high-speed flows is crucial to the success of a design. Today's models come in many different styles and formats. Allegro SI DML (Device Modeling Language) enables you to accurately describe all devices and advanced behaviors.

A DML model refers to a single specific entity. That entity can be a package model, an interconnect model, an Espice model, or a translated IBIS model. It should be noted that an IBIS model can contain a package model within one translated file.

A DML file contains one or more models written in the DML language and is identified by its `.dml` extension. These model files are used in circuit simulation by analysis tools such as PCB SI and SigXplorer. Models are procured or developed in advance of simulation and used to characterize manufactured components such as ICs, discrete components, and connectors. The Allegro SI simulator requires that simulation models be in DML format for successful simulation. For further details on DML, refer to the *Allegro SI Device Modeling Language User Guide*.

Introduction to Simulation Models

There are two basic categories of models used to build circuits for simulation.

- Device Models
- Interconnect Models

Device Models

Device models are stored in files with a `.dml` extension. A device model library consists of a `.dml` file that contains one or more device models.

The available device models, their contents, and how they are used are described in Table 8-1 on page 136.

Figure 8-1 Device Models

```
DesignLink
Cable
ESpiceDevice
IbisDevice
PackageModel
AnyIOCell
IbisInput
IbisTerminator
IbisOutput
IbisOutput_OpenPullUp
IbisOutput_OpenPullDown
IbisIO
IbisIO_OpenPullUp
IbisIO_OpenPullDown
AnalogOutput
BoardModel
Connector
```

Table 8-1 Device Models Supported by Signal Integrity Tools

Model Type	Use and Relationship
Design Link	Used to specify system-level connectivity, like multi-board or advanced package-on-board scenarios.
Cable	Referenced from a system configuration. Models cables interconnecting multiple boards. Can be an RLGC model or SPICE sub-circuits.

Model Type	Use and Relationship
ESpiceDevice	Assigned to discrete parts like resistors and capacitors. Contains SPICE sub-circuits.
IbisDevice	Assigned to ICs and connectors. An IbisDevice model for a connector has package parasitics but no IOCell models.
PackageModel	Referenced from an IbisDevice model. Models the package parasitics of the entire component package. Can be an RLGC matrix model or SPICE sub-circuits.
AnyIOCell	Referenced from an IBISDevice model. An IOCell model is used to model driver and receiver buffers at the pin level.
IbisInput	A type of IOCell model. It is a receiver model.
IbisTerminator	Models termination internal to the device pin
IbisOutput	A type of IOCell model. It is a driver model.
IbisOutput_OpenPullUp	Driver model with no pullup resistor
IbisOutput_OpenPullDown	Driver model with no pulldown resistor
IbisIO	Bidirectional buffer model, which can drive or receive
IbisIO_OpenPullUp	Bidirectional buffer model with no pullup resistor
IbisIO_OpenPullDown	Bidirectional buffer model with no pulldown resistor
AnalogOutput	Models the behavior of an analog device pin
BoardModel	Referenced from a system configuration. Models entire boards for situations in which the physical Allegro database is not available. Contains SPICE sub circuits.
Connector	Used to create interconnect models (iml).

Interconnect Models

Interconnect models are extracted directly from the physical design database and synthesized on demand. Interconnect models are stored in files (`.iml`) so that they can be reused. The `.iml` files cover such items as traces and vias.

The available interconnect models, their contents, and how they are used are described in and Table 8-2 on page 138.

Figure 8-2 Interconnect Models

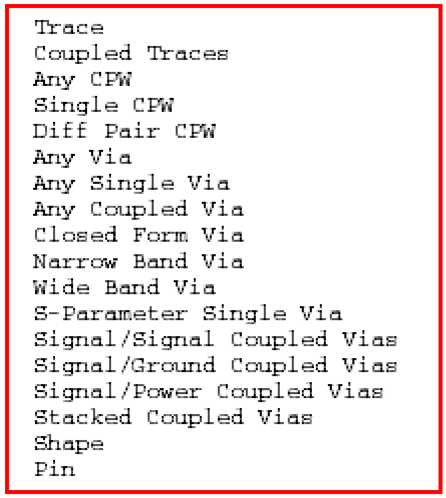


Table 8-2 Interconnect Models

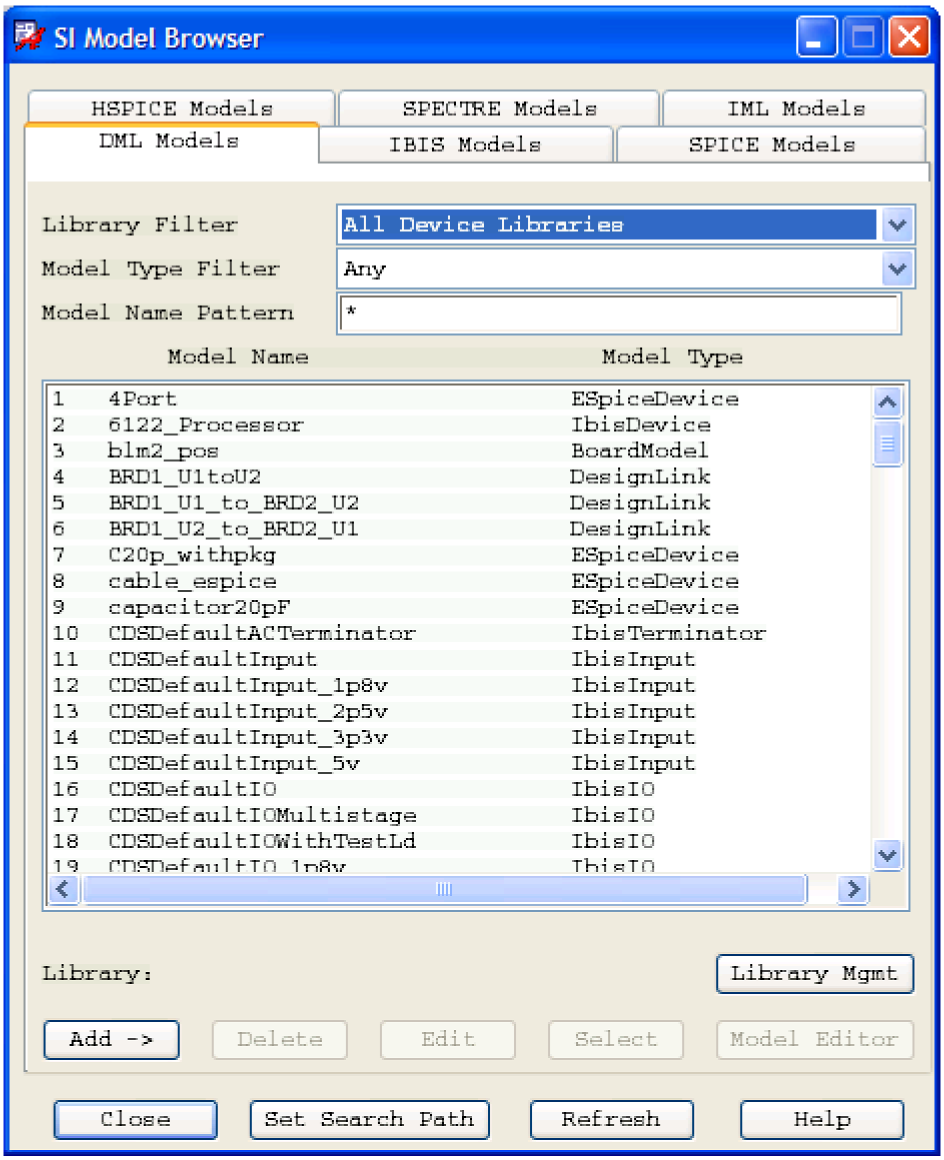
Model Type	Use and Relationship
Trace	Geometry-based model that represents a single transmission line with no coupling. A Trace can have frequency-dependent loss.
Coupled Traces	Geometry-based model representing coupled lossey transmission lines.
Any CPW	Any model representing a coplanar waveguide (CPW) structure.
Single CPW	Represents a single CPW. It is a two-pin symbol containing no dielectrics.
Diff Pair CPW	Represents a differential pair CPW. It is a four-pin symbol containing no dielectrics.
Any Via	Models the parasitics of a via providing z-axis connectivity between traces.
Any Single Via	

Model Type	Use and Relationship
Any Coupled Via	
Closed Form Via	■ Fast Closed Form This is the most basic form of a via model. It is a static method, formula-generated model. ■ Detailed Closed Form A more accurate, static method, formula-generated model created by FSvia.
Narrow/Wide Band Via	An FSvia-generated narrow band (single frequency-point) or wide band (multiple frequency-points) model containing RLGC values for a range of frequencies, as specified by the user. When generating a Wideband model, FSvia first generates S-Parameters and then creates RLGC values based on those S-Parameters.
S-Parameter Single Via	An FSvia-generated S-Parameter model containing values for a range of frequencies, as specified by the user
Signal/Signal Coupled Via	Represents a via model between two signals.
Signal/Ground Coupled Via	Represents a via model between a signal and a ground component.
Signal/Power Coupled Via	Represents a via model between a signal and a power component.
Stacked Coupled Via	
Shape	Models a copper shape encountered in a physical design.
Pin	

Managing Device and Interconnect Models

You use the *SI Model Browser* to create and manage your libraries of device and interconnect models, and launch Model Editor. You can also use SI Model Browser to specify which device and interconnect libraries you want the tool to access, as well as the order of library access.

Figure 8-3 SI Model Browser



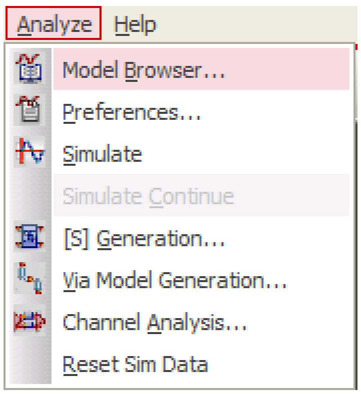
To display SI Model Browser:

- Choose *Analyze – Model Browser* in SigXplorer.

-or-

- Choose *Analyze – Model Browser* in PCB SI.

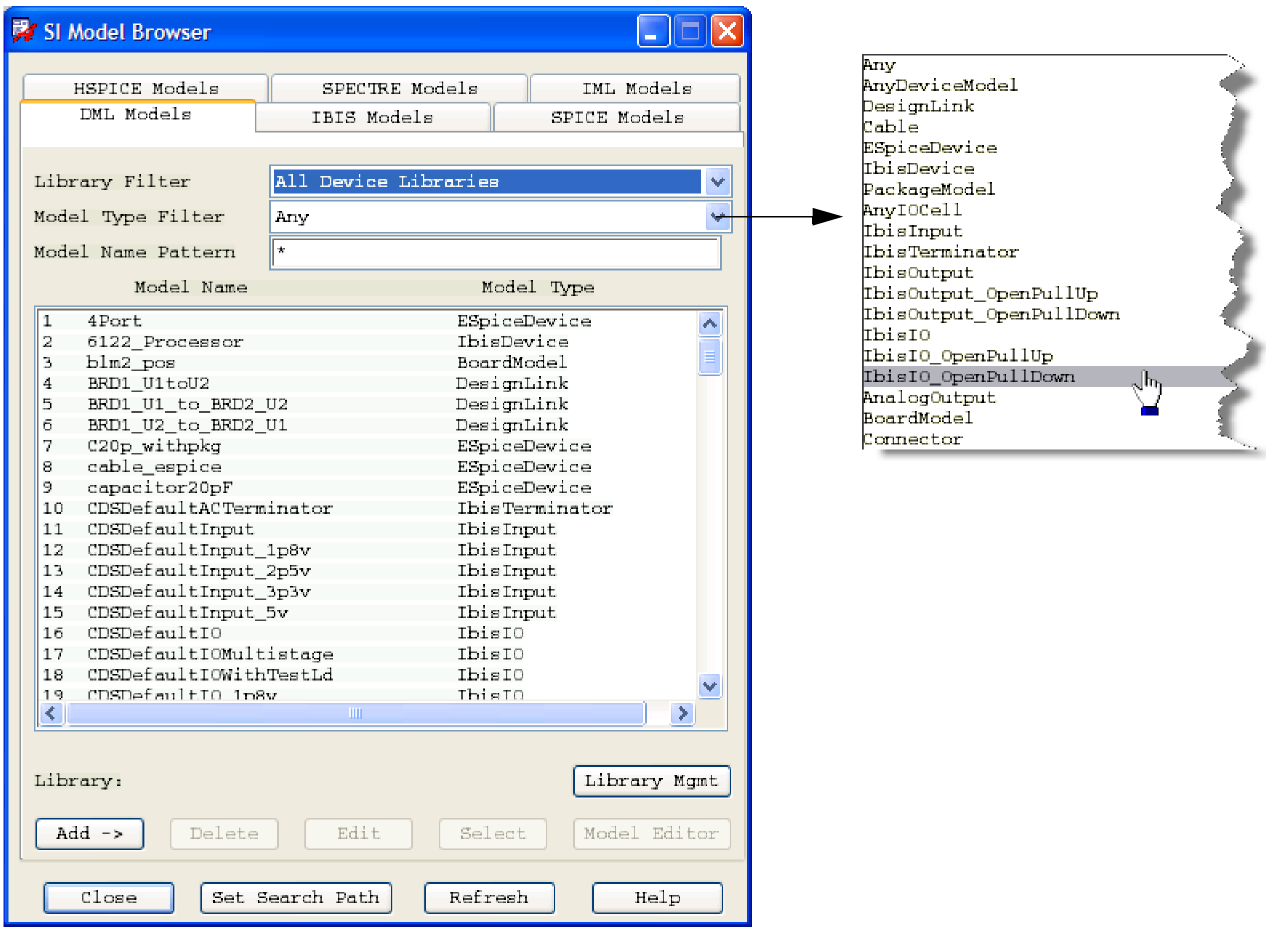
Figure 8-4 Accessing Signal Model Libraries



Consuming Signal Models

As an SI engineer, you deal with many different types of signal models. SigXplorer works with signal models in the DML model format, but can handle the translation in the background. This way you can manage and debug the original signal models in their native formats. You begin the process by accessing the appropriate device file library from the SI Model Browser.

Figure 8-5 SI Model Browser



The SI Model Browser's tabbed interface accommodates the model type that you want to translate, be it IBIS, Spectre, Spice, IML, or HSPICE. From these tabs, you can also edit a model directly in its native format or translate it. Once translated, these models also appear under the DML tab.

Each tab contains a field for filtering the listed models, as well as a button to set the model's library search path and to set its associated file extensions (see Figure 8-6 on page 144).

Specifying Library Search Order

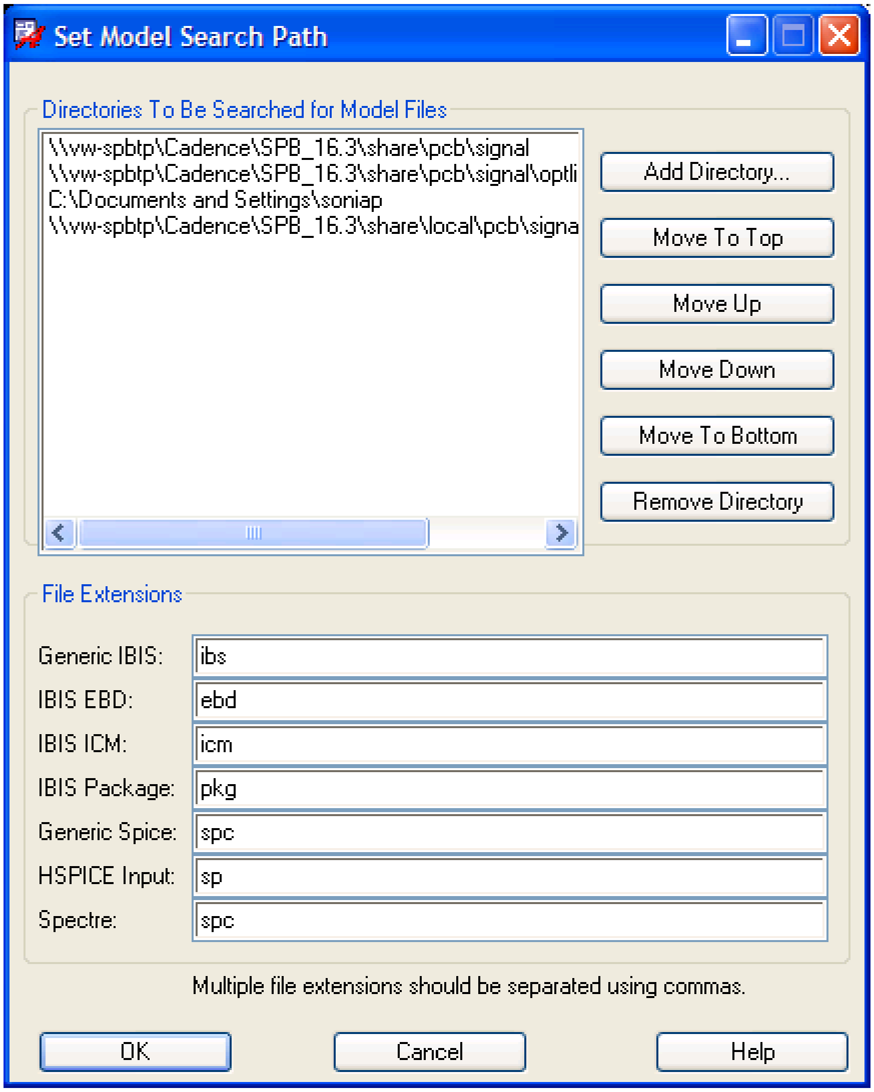
You can specify the directories in which to search for signal models, and their search order in the Set Model Search Path dialog box.

To access Set Model Search Path dialog box:

➡ Click the *Set Search Path* button in SI Model Browser.

This dialog box is used to manage the search path and file extensions for all model formats. You can extend the default file extensions provided by adding others, separating each with a comma.

Figure 8-6 Set Model Search Path dialog box



The default search locations include the current project directory and Cadence defaults. You can add a new search directory by clicking the *Add Directory* button. You can also reorder the directories to search for models and remove a directory from the search list.

Tip

SigXplorer automatically saves these edits to your local `.env` file as arguments to the following environment variables:

`SI_MODEL_FILE_EXT`	Signal model file extensions.

`SI_MODEL_PATH`	Search directories for IBIS, Spice, HSPICE, and Spectre signal models.
`SI_IGNORE_DML_LIBS`	DML libraries to be ignored while performing search.
`SI_IGNORE_IML_LIBS`	IML libraries to be ignored while performing search.

For example the corresponding entry in the `env` file for `SI_MODEL_PATH` is:

```
set SI_MODEL_PATH = . C:/Workshops/16.5/cm/site/pcb/signal D:/SPB/cds/16.5/share/
pcb/signal D:/SPB/cds/16.5/share/pcb/signal/optlib
```

These settings support the use of variables in the paths making them more portable. For example:

```
set SI_MODEL_PATH = . $MY_LIB_PATH $GLOBAL_LIB_PATH
```

The variables for `MY_LIB_PATH` and `GLOBAL_LIB_PATH` are only expanded when the files are opened and are not stored anywhere in their expanded format. When you edit the search path in the Set Search Path dialog, these variables are expanded.

These path variables are updated based on existing `case.cfg` files in run directories when you open a pre-16.3 design.

Setting Working Libraries

The `SI_DML_WORKING_LIB` and `SI_IML_WORKING_LIB` Allegro environment variables are used to store the names of the current dml and iml working libraries.

`SI_DML_WORKING_LIB`	Sets a DML library as the working library. Auto-generated models are stored in the working library.
`SI_IML_WORKING_LIB`	Sets an IML library as the working library. Auto-generated models are stored in the working library.

You can set the value of these variables to special keywords that will create specific file names. The values these two variables can take are described in the following table:

`SI_DML_WORKING_LIB`	■ **user**: The dml working library will be named <*username*>`_devices.dml` ■ **host**: The dml working library will be named <*hostname*>`_devices.dml` ■ **unique**: This option lets a single user have different file names if two different processes are being run in the same directory. Each file name is unique as it will include the process ID. With the unique option, the dml working library will be named <*host_name*>**_**<*user_name*>**_**<*process_id*>`_devices.dml`. ■ **<any other string>**: This string defines the name of the dml working library.

`SI_IML_WORKING_LIB`	■ **user**: The iml working library will be named <*username*>`_devices.iml` ■ **host**: The iml working library will be named <*hostname*>`_devices.iml` ■ **unique**: This option lets a single user have different file names if two different processes are being run in the same directory. Each file name is unique as it will include the process ID. With the unique option, the dml working library will be named <*host_name*>**_**<*user_name*>**_**<*process_id*>**_**`devices.iml`. ■ **<any other string>**: This string defines the name of the iml working library.

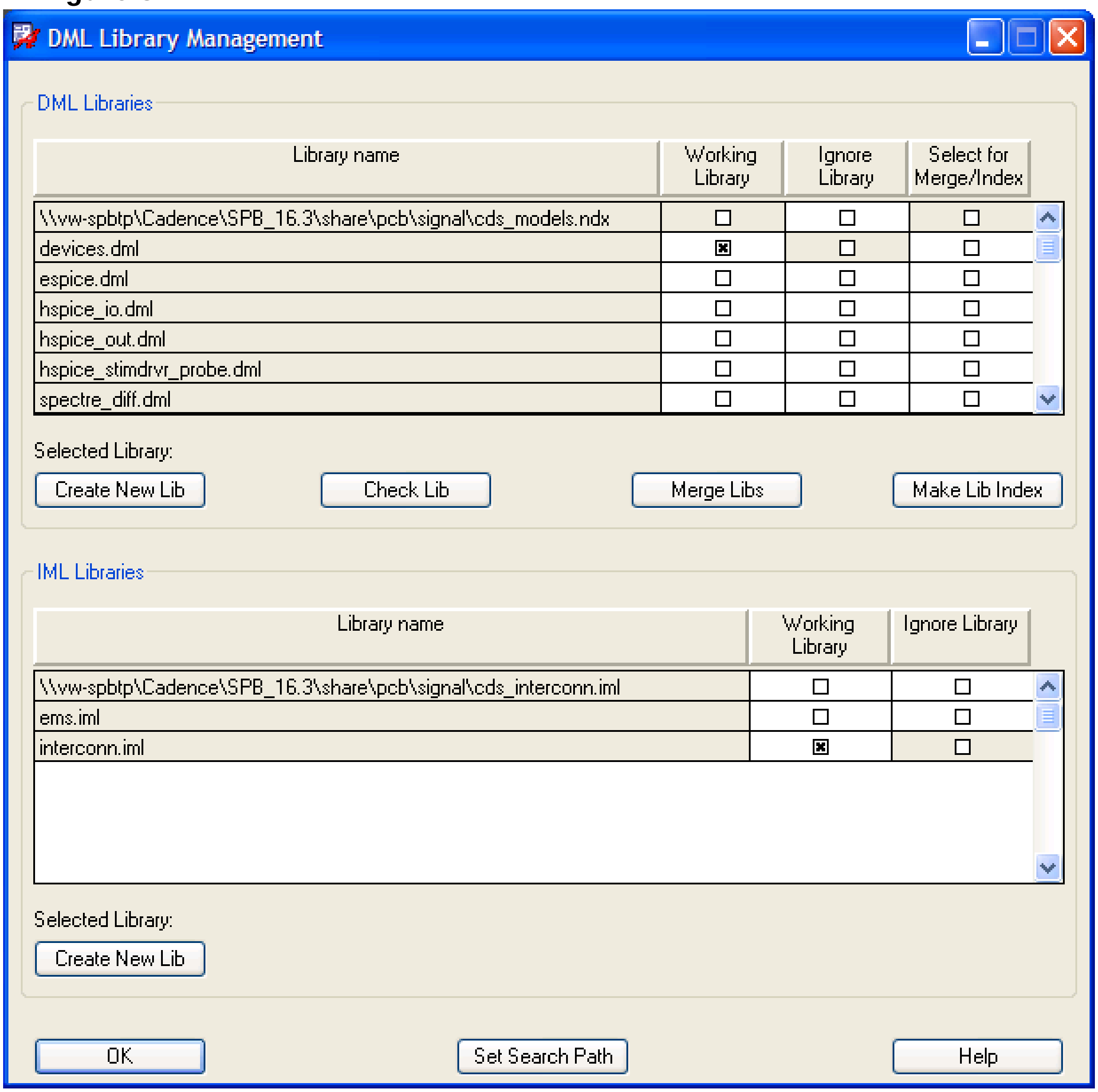
Managing DML Libraries

You use the DML Library Management dialog box to manage DML libraries.

To access the DML Library Management dialog box:

➡ Click the *Library Mgmt* button on the *DML Models* or the *IML Models* tab of SI Model Browser dialog.

Figure 8-7



The DML Library Management dialog box provides controls to set the working library, ignore libraries, and create indices.

Using the DML Library Management dialog box, you can:

- create a new library and add it to the list of libraries
- Perform a syntax check on a DML library
- specify the working libraries
- create an index for a device model library
- merge two or more device model libraries

As many functions of the *SI Model Browser* are related to *model development*, you should refer to the following if you are an SI librarian:

- *Allegro SI Device Modeling Language User Guide*
- *Allegro SI User Guide*

Translating Models

This section covers:

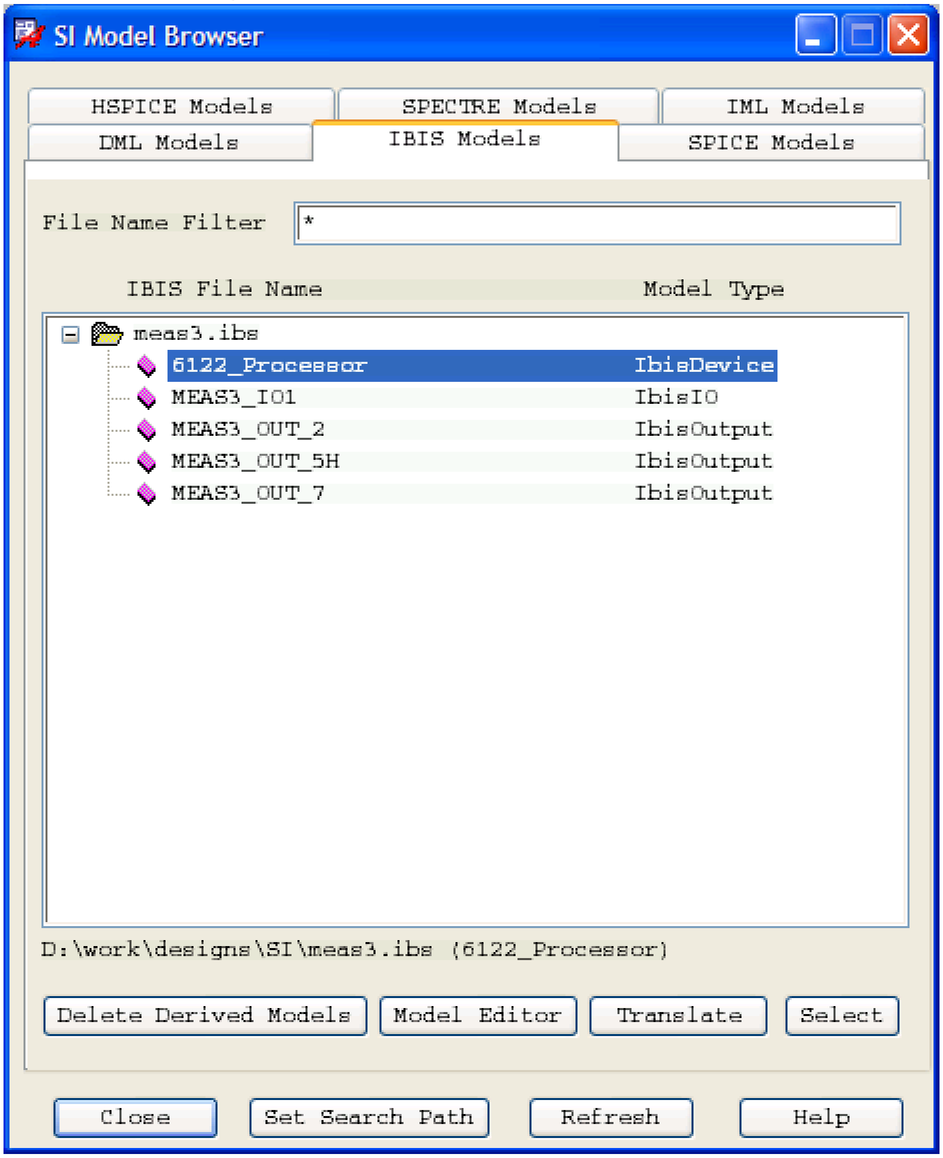
- Translating IBIS and Spice Models
- Translating HSPICE and Spectre Models

Translating IBIS and Spice Models

The process of translating IBIS and Spice models is similar, although IBIS lets you prepend the model's name to the translated DML (see Figure 8-9).

To initiate the translation, select the model under the appropriate tab (IBIS or Spice) and click the *Translate* button.

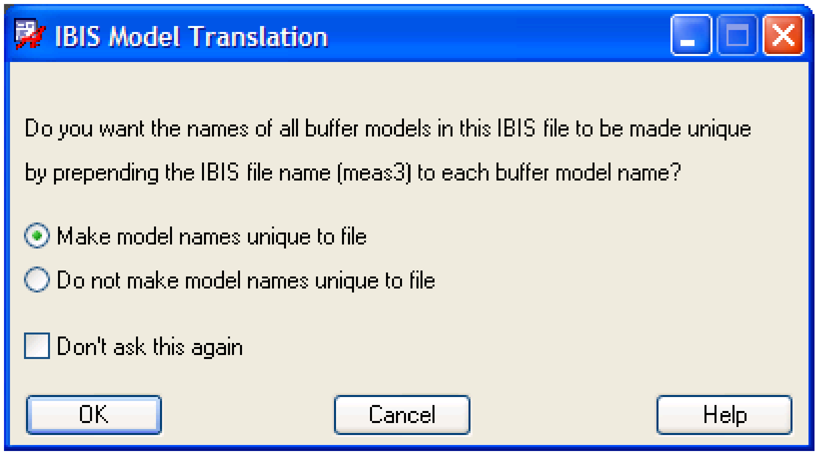
Figure 8-8 Translating Models



A dialog appears prompting you to choose how to handle the model names. IBIS files can be managed on disk in their native format and are automatically translated to DML transparent to the user. If errors are detected during the translation, an error is displayed.

This dialog is used when translating the generic IBIS file. IBIS EBD, ICM, and Package models work differently. The dialog for EBD models gives you the choice of creating either an IBISDevice or Board model. The dialog for an ICM model gives you the choice of creating a connector, blackbox, or package model. Package models are always translated to a DML package model.

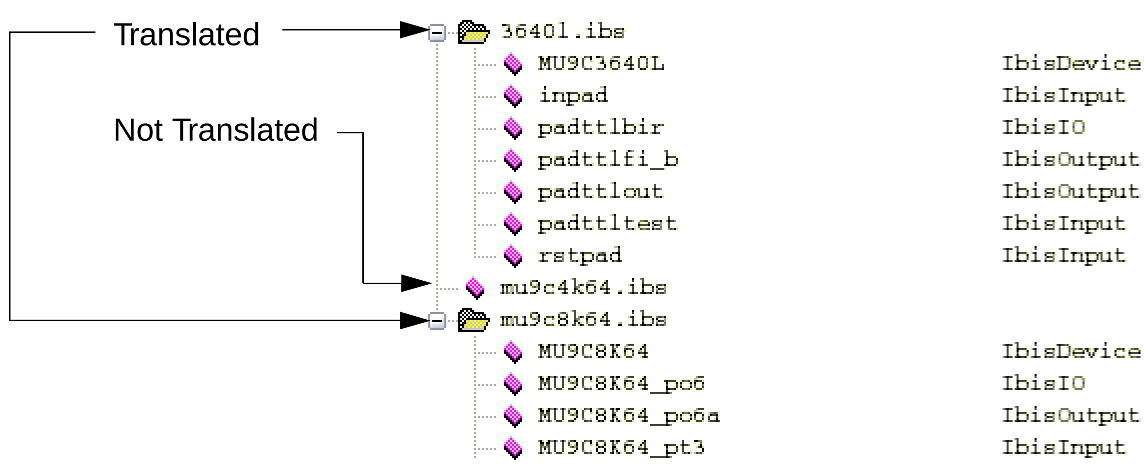
Figure 8-9 Qualifying File Names



At this point you have the option of selecting a buffer model directly from SI Model Browser dialog or a specific pin (packaged buffer model) from the IBISDevice. In either case, after a selection is made in SI Model Browser, you need to click the *Select* button

Once a signal model is translated, its icon is promoted to a folder, with translated DML models shown as children, along with the model's purpose (see Figure 8-10 on page 152).

Figure 8-10 Model Translation Nodes (IBIS shown)



To delete a model,

➡ Select the model in the list and click the *Delete Derived Models* button.

This deletes all the child DML models associated with an IBIS model.

If the selected model fails to translate, you are prompted to edit the model. If you choose *Yes*, the model is opened in Model Editor. After you save the edits, the model is re-translated after you close Model Editor.

Tip

You can also select model and click the *Model Editor* button to edit the model in its native format.

Caution

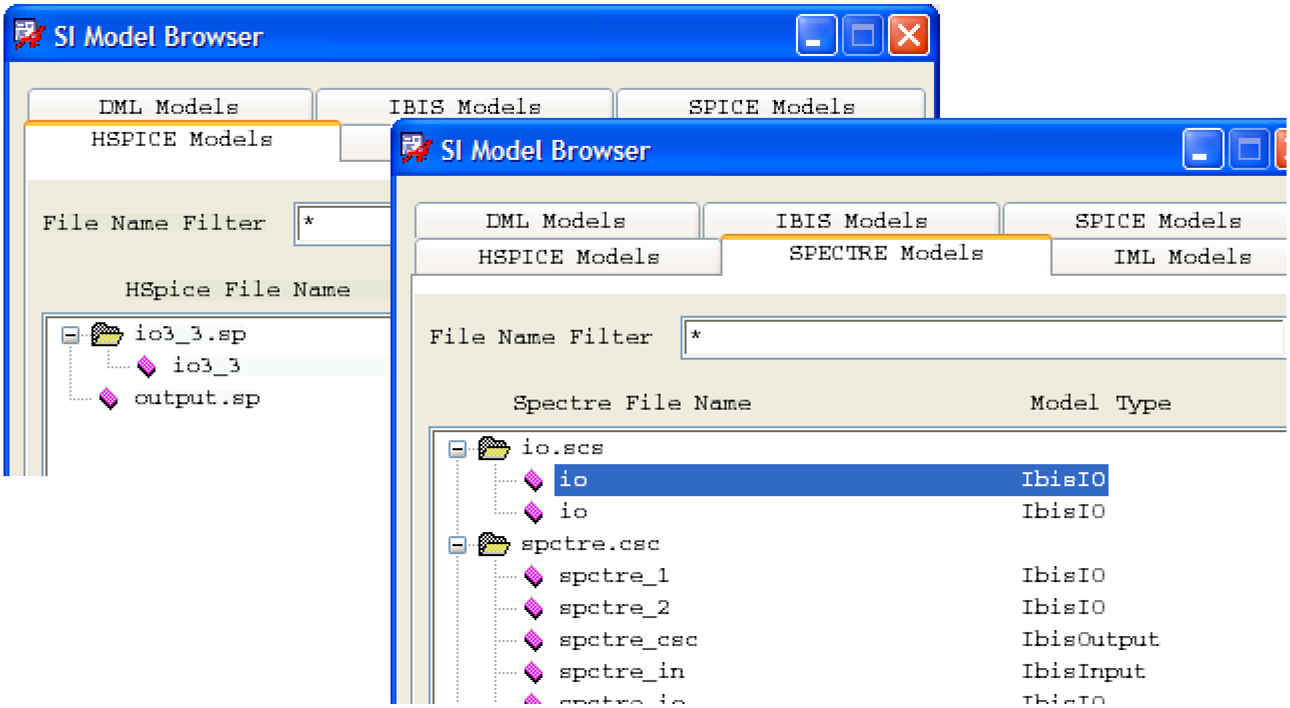
If you edit a DML model (that was translated from an IBIS file) from the DML Models tab, the model is no longer consistent with the data in the IBIS file, and it will be removed from the list of DML files shown under the IBIS tab. It will, however, remain in the list of models shown under the DML Models tab.

Translating HSPICE and Spectre Models

HSPICE and Spectre translations are more involved than IBIS and Spice. SigXplorer's model translation wizard, however, guides you through this translation, step-by-step.

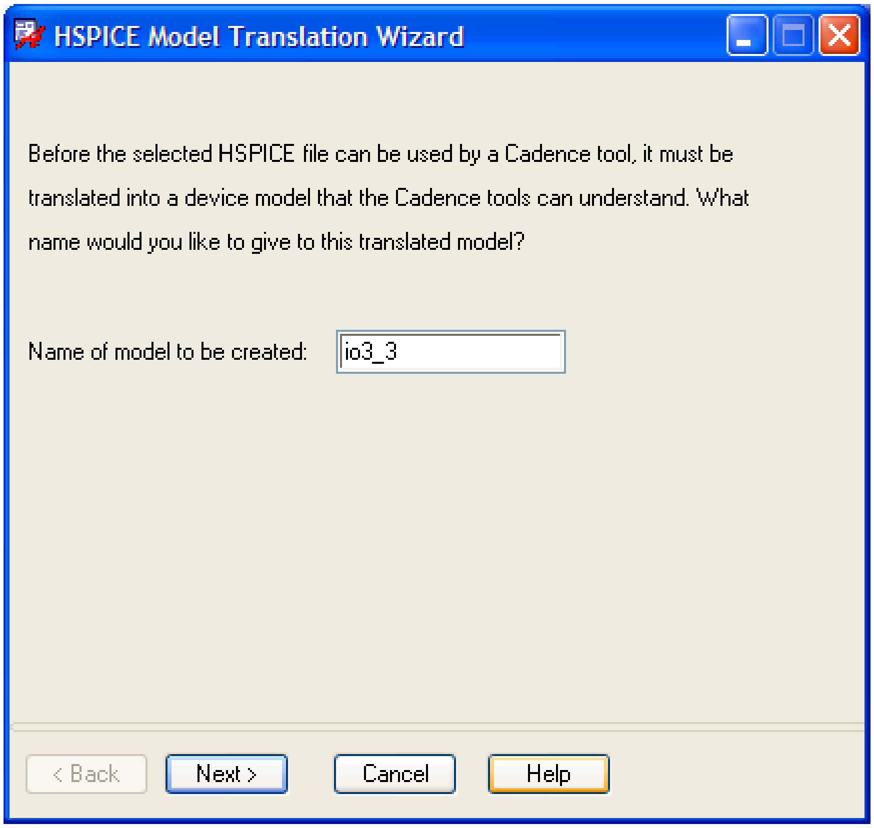
To translate an HSpice or Spectre model:

1. Click the model name in the appropriate tab, HSPICE or Spectre.



2. Click *Translate*.

A translation wizard starts. The first window gives an option to rename the model.



In the remaining pages of the wizard, you need to do the following:

- Select the type of translation you want to perform: Black Box model or IOCell model.
- Choose the type of IOCell model out of six model types and then select subcircuits in the model. One subcircuit must be selected as the Master and the rest can be selected for inclusion in the model.
- Identify the function of each subcircuit terminal.
- Specify how each Signoise IO cell terminal is to be treated.
- Specify the DC voltage of the IO cell, and the voltage ranges for the input stimulus signal and the Enable signal
- Specify reference values for power, ground, pull up and pull down devices. You can choose to copy from an existing model as well.
- Specify high and low input logic threshold voltages.
- Specify if you want to include any files in the models being created.

- Specify if you want to include package parasitics. If you choose *Yes*, you will need to specify how to define package parasitics.

When the final page of the wizard displays, click *Finish* to translate the model. The translated model appears in the appropriate tabbed page. See Figure 8-12

You might also want to view the models to be created. You can do so by clicking the *View Model(s) To Be Created* button. The model opens in Model Editor.

Figure 8-11 Final Page of the Model Translation Wizard

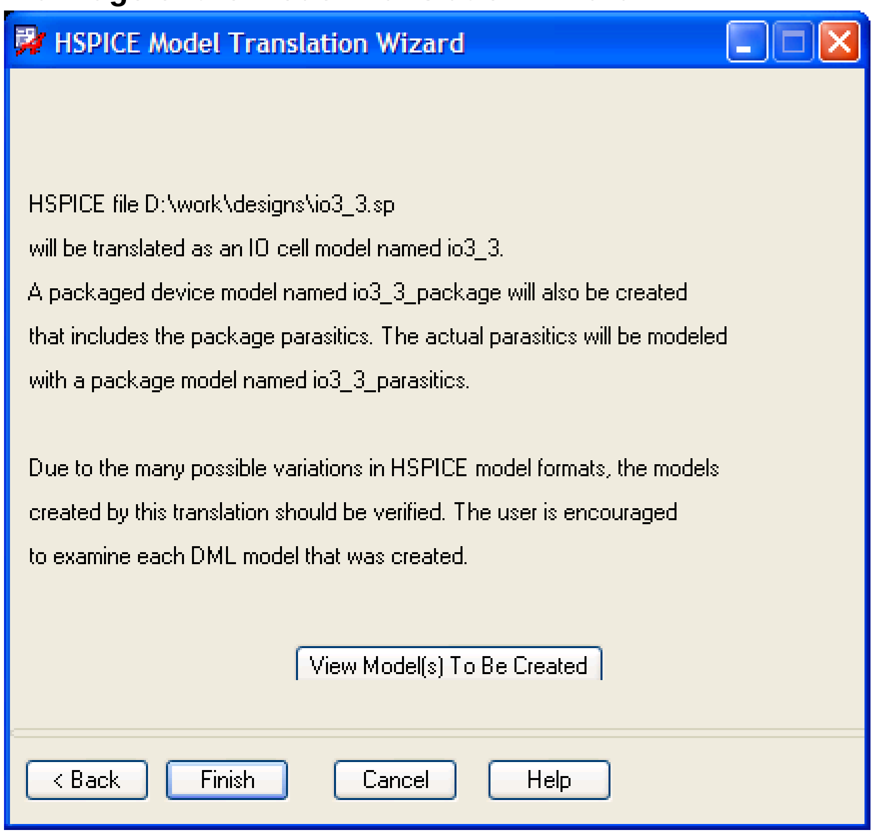
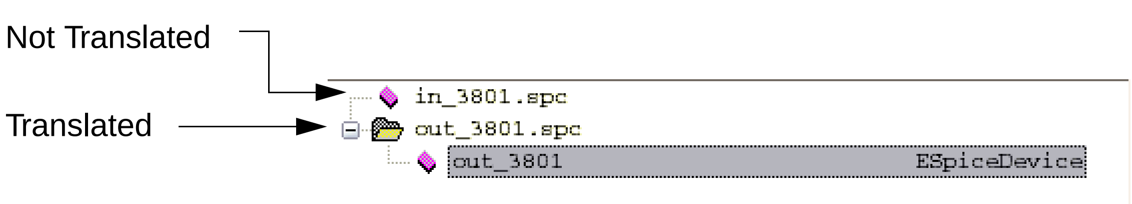


Figure 8-12 Translated Model



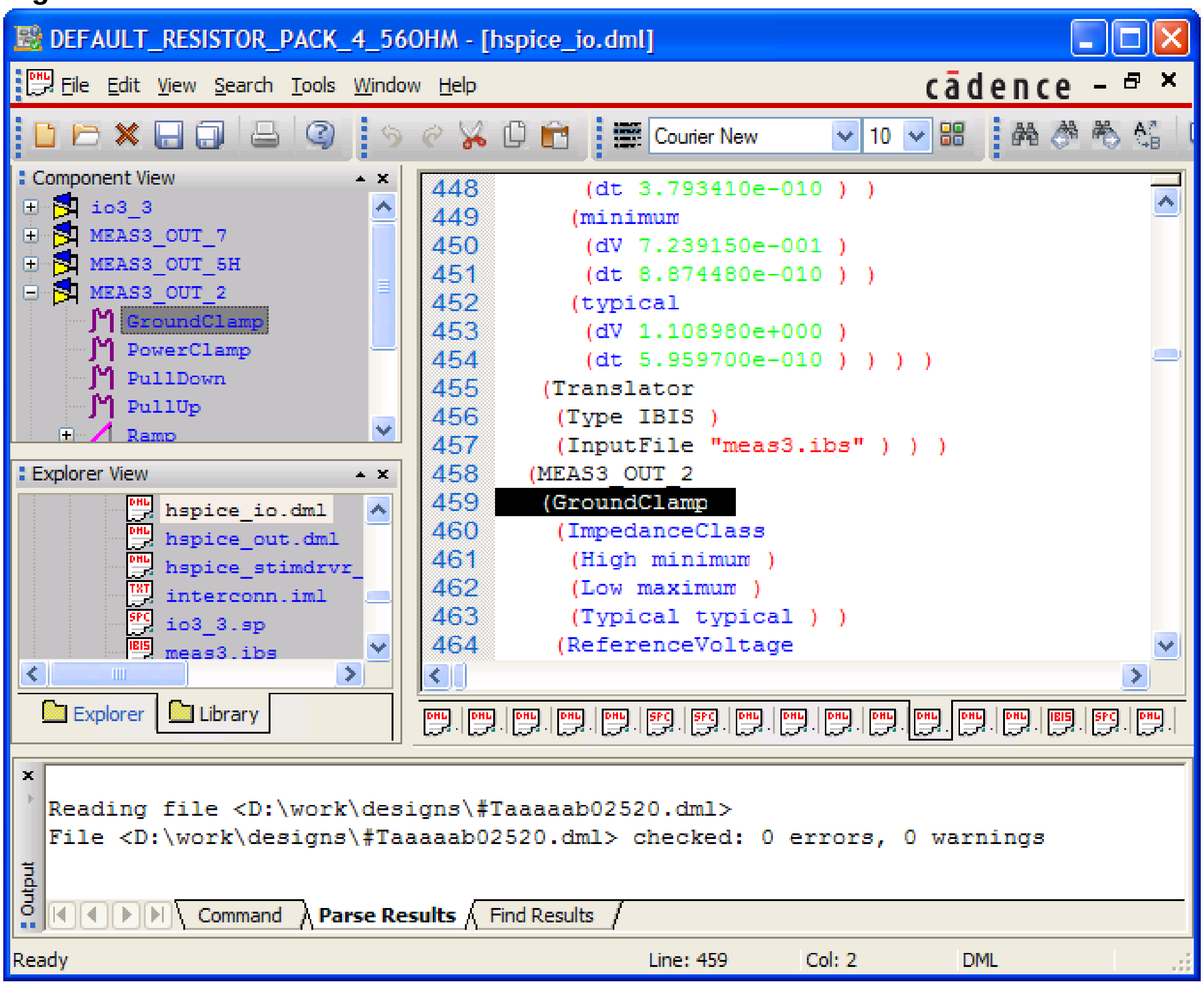
9

Working with Model Editor

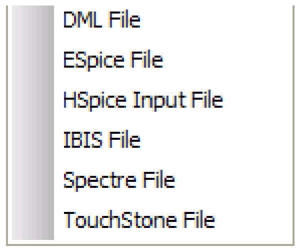
Model Editor

Model Editor is a high-speed design editing tool that helps you ensure the integrity of the model data required for high-speed circuit simulations. It lets you create, manipulate, and validate models quickly in an easy-to-use editing environment.

Figure 9-1 Model Editor



Model Editor provides a model browser and syntax checker (parser) for models written in IBIS as well as for advanced models written in DML. The following device model formats are supported in Model Editor:



This section covers:

- Launching Model Editor
- Explorer View
- Component View
- Editor View
- Output Window
- Setting Color and Font Options
- Associating File Extensions with Models

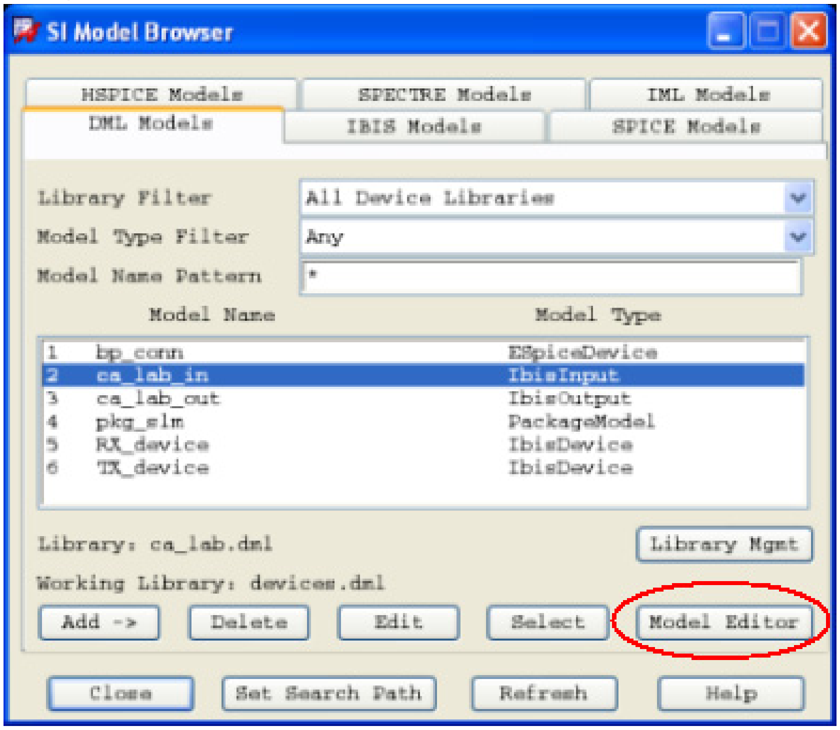
Launching Model Editor

You launch Model Editor from SI Model Browser for any of the models listed there, either native dml or generated by translation.

1. Select a model.
2. Click the *Model Editor* button,

   The model is opened for editing in the Model Editor window.

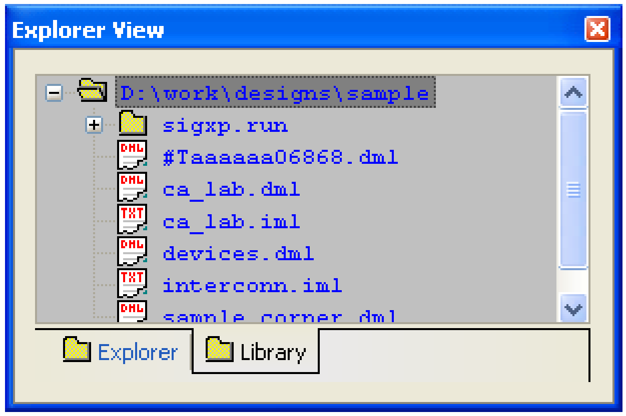
Figure 9-2 launching Model Editor



Explorer View

The model library directories are displayed and explored in this dockable window. You can specify the directories containing all the model files using the system environment variable `SI_MODEL_PATH`. This window displays all the model files and directories in a tree structure.

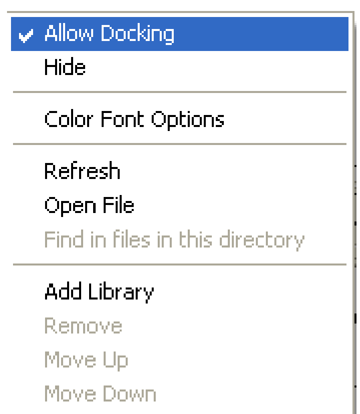
Figure 9-3 Explorer View



All the supported model files with previously-registered file extension in system environment `SI_MODEL_FILE_EXT` are listed in the explorer tree. You can open a model file by double-clicking the model file in the tree.

On an RMB action in the explorer view, a pop-up menu is displayed as shown in Figure 9-4 .

Figure 9-4 Explorer View RMB Menu



Menu Option	Description
Color Font Options	Opens the Color Font Option Dialog Box where you can specify color and font options for the different views and text elements such as keywords, numbers, operators, and so on in Model Editor.
Refresh	Refreshes the explorer view.

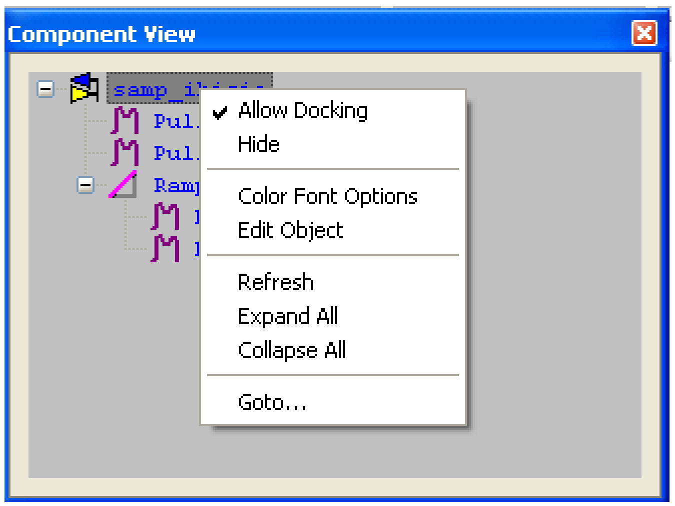
Open File	Opens the selected model file in the Editor View on the right pane.
Find files in this directory	Finds matched filename in the current directory.
Add Library	Adds directory to the model library path.
Remove	Removes the selected directory from the model library path.
Move Up	Moves the file one place up in the list.
Move Down	Moves the file one place down in the list.

Component View

You use the Component view to navigate to a specific model or subsection of a model. The component view is associated with the active opened component and acts as a quick navigator for various data and components in model data. When the model file is opened in the editor view, the component hierarchy is displayed in the component view as a tree.

On an RMB action in the component view, a pop-up menu is displayed.

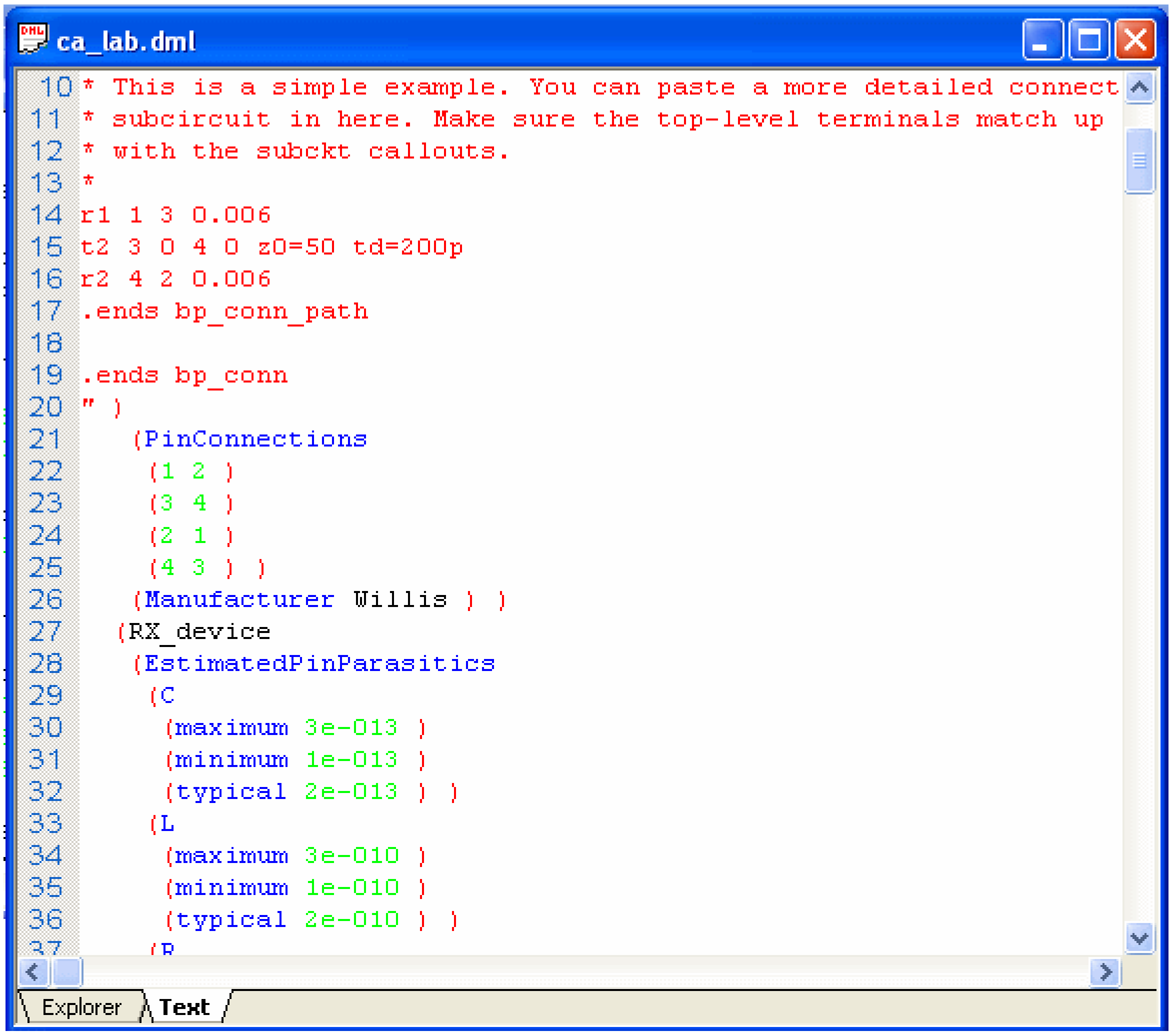
Figure 9-5 Component View RMB Menu



Menu Option	Description
Hide	Hides the Component view.
Color Font Options	Opens the Color Font Option Dialog Box where you can specify color and font options for the different views and text elements such as keywords, numbers, operators, and so on in Model Editor.
Edit Object	Opens the IOCell Object Editor where you can edit and manage various parameters of the selected IOCell model.
Refresh	Refreshes the explorer view.
Expand All	Expands the hierarchy of the active component.
Collapse All	Collapses the hierarchy of the active component.
Goto	Navigates to the selected element of the component.

Editor View

Model Editor provides you with the ability to view and edit model files in a convenient way. It uses color coding of text to distinguish among model data elements and components, such as comment, keyword, operators, or a preprocessor symbol. Syntax-coloring enables you to identify various elements of the model data quickly and correctly.



The primary features of the editor view include:

- Text selection, insertion, and deletion
- Customizable syntax color highlighting
- Full support for text find and replace
- Support for multiple undo/redo

- Support for drag-and-drop
- Support for drawing line index, and customized flags in the gutter (the space to the left of the editor view).
- Line highlighting

Output Window

You can find and replace various texts on active or all opened documents in Model Editor. All search or replace information are displayed in the Output window.

Parsing Models

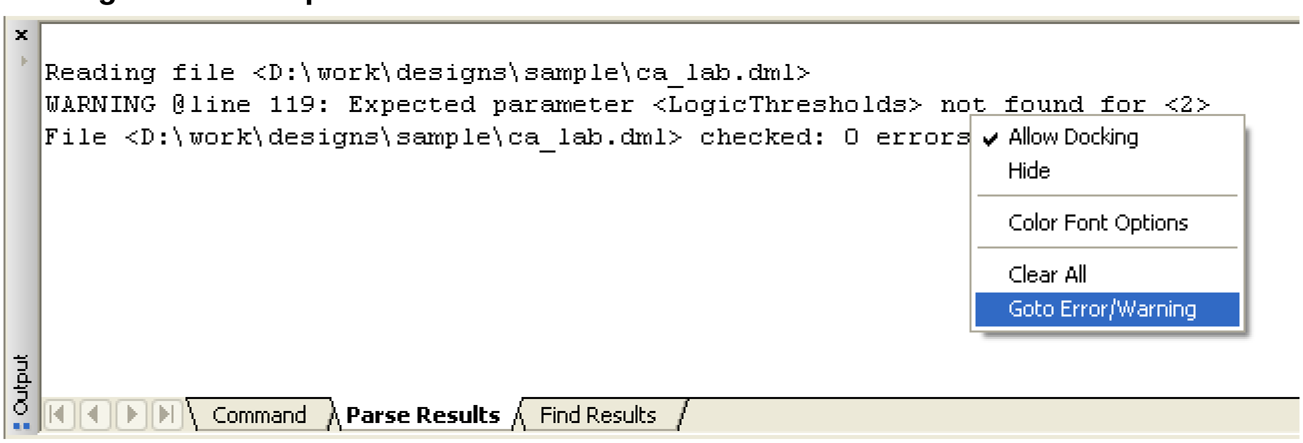
In addition to opening and editing model files in Model Editor, you can also parse model (`.dml`) files to determine any syntax errors. When you open a valid model file in Model Editor, it is automatically parsed using the parser appropriate for the file type.

You can also request to parse an open file at any time while you work with it in Model Editor.

➡ Choose *Tools – Parse* in the Model Editor window.

When parsing of the file completes, Model Editor displays errors or warnings in the Output window to mark any syntactical problems encountered within each model object contained in the file. The Output window pop-up menu provides the option to jump to any error or warning flagged in the model file. You can also double-click the message to quickly locate the content in Editor View.

Figure 9-6 Output Window



Setting Color and Font Options

You can set the text color and font schema to display elements in the different views of Model Editor in the Color Font Option dialog box.

Figure 9-7 Color Font Option Dialog Box

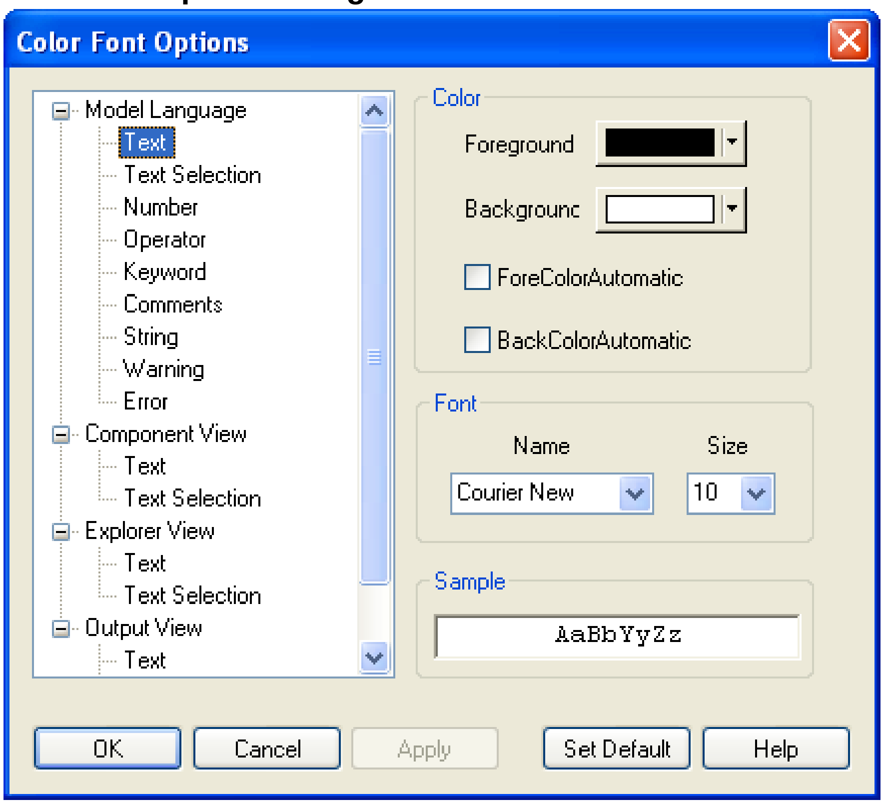


Table 9-1 Color Font Options

Option	Description
Model Language	Specify the color for the following in Editor view: ■ Text ■ Text Selection ■ Number ■ Operator ■ Keyword ■ Comments ■ String ■ Warning ■ Error
Component View	Specify the color for the following in Component view: ■ Text ■ Text Selection
Explorer View	Specify the color for the following in Explorer view: ■ Text ■ Text Selection
Output View	Specify the color for the following in Output view: ■ Text ■ Text Selection
Color	Set the color of the element selected in the left pane. ■ **Foreground**: Set the text color. ■ **Background:** Set the color of the background. ■ **Fore Color automatic:** Specifies whether to use the default color for text or selected text. ■ **Back Color automatic:** Specifies whether to use the default color for the background of text or selected text.

Option	Description
Font	Set the font name and size for the element selected in the left pane.
Set Default	Discard changes and reset it to default setting.

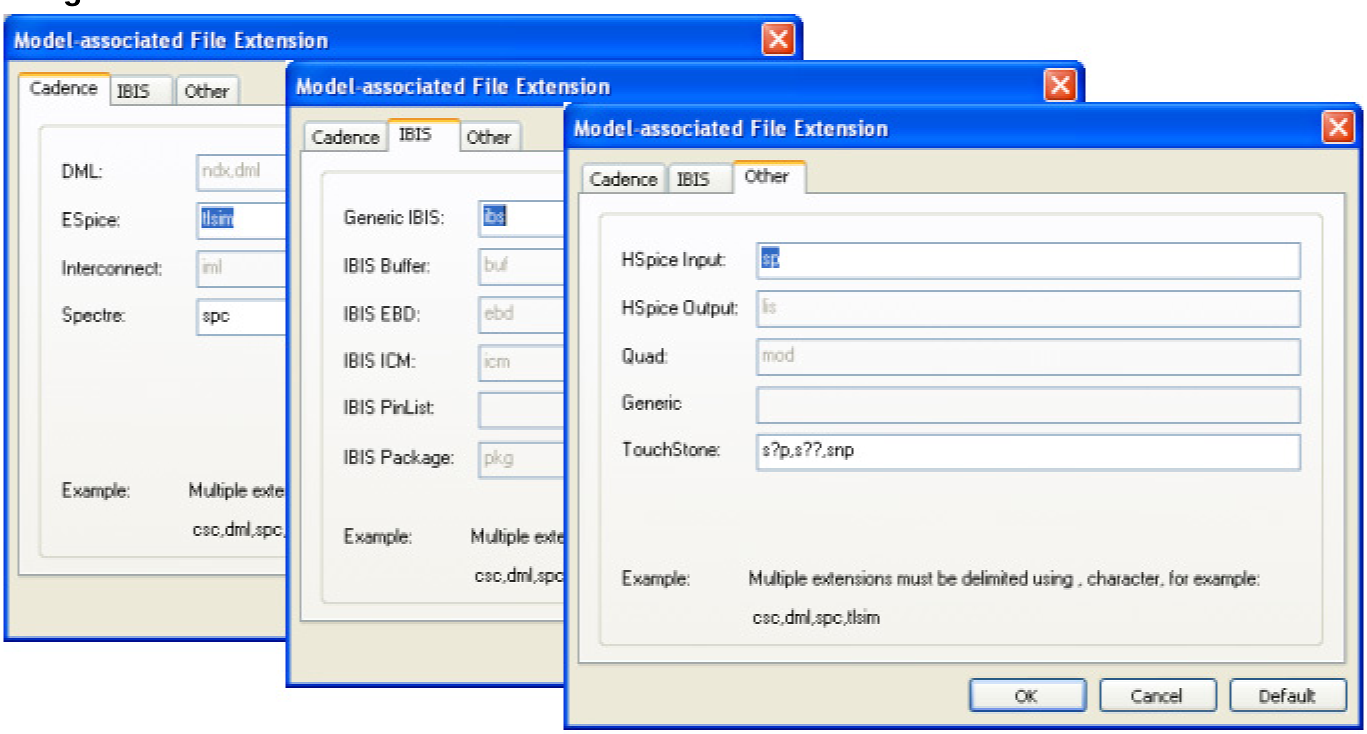
Associating File Extensions with Models

Model Editor determines the model format based on a given file extension. The mapping between the file extension and the model format are stored in the system variable, `SI_MODEL_FILE_EXT`. You can change this mapping in the Model Associated File Extension dialog box.

➡ Choose *File – File Extension* to launch the dialog box.

The updated mapping is stored in Allegro local environment file `<LOCALENV>/env` and shared with SI Model Browser.

Figure 9-8 Model Associated File Extension



You can associate file extensions for the following models in the three tabbed pages of the Model Associated File Extensions dialog box:

- Cadence
  - DML
  - ESpice
  - Interconnect
  - Spectre
- IBIS
  - Generic IBIS
  - IBIS Buffer
  - IBS EBD
  - IBIS ICM
  - IBIS PinList
  - IBIS Package
- Other
  - HSpice Input
  - HSpice Output
  - Quad
  - Generic Spice
  - Touchstone

Object Editor

The IOCell Object Editor provides you an interface to edit and manage various parameters of the selected IOCell model. You can edit any existing IOCell model that has been created and added to a library. You need to edit a model if you created it by cloning (copying) an existing model, so that it characterizes the device you are modeling. Also, if you create a model from scratch, it will contain default values that you may want to edit.

This section covers:

- Launching Object Editor
- General Information
- Output Information
- Delay Measurement
- Pullup Waveform
- Pulldown Waveform
- Input Information
- Power Clamp
- Ground Clamp
- Rising Waveform/Falling Waveform

The pages displayed may differ depending on whether the wizard is launched on an IbisInput, IbisOutput, IOCell, or a pure IBIS model.

Launching Object Editor

To launch Object Editor:

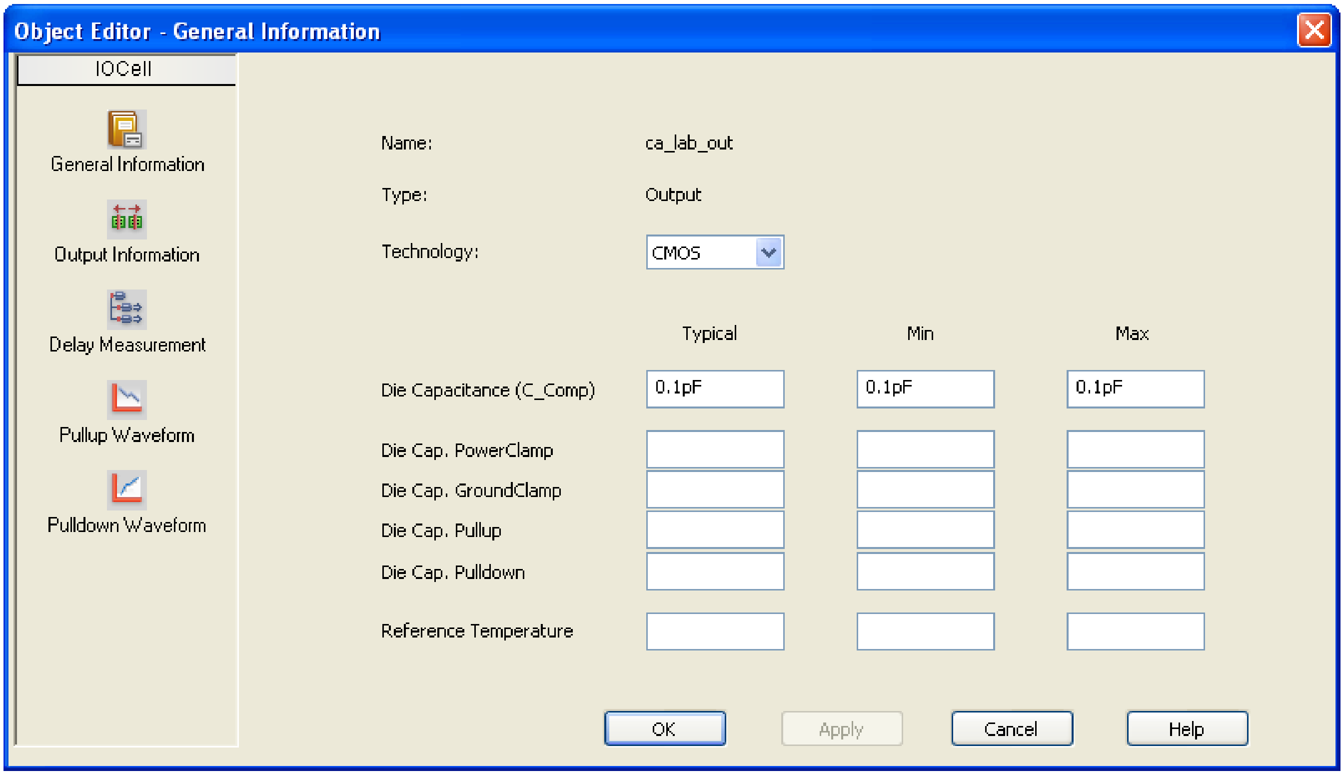
➡ In the Model Editor window, double click the IOCell model.

OR

➡ Right-click the IOCell model in the Component View and choose *Edit Object* from the pop-up menu.

The IOCell Object Editor is displayed.

Figure 9-9 IOCell Object Editor



General Information

In the General Information page, you edit general attributes of the selected IOCell model.

Table 9-2 Object Editor - General Information

Option	Description
Name	The name of the IOCell model you are editing.
Type	The model type of the IOCell model.
Technology	The technology family of the IOCell model. The following three technologies are supported for IOCell models: ■ CMOS ■ TTL ■ ECL

Option	Description
Die Capacitance (C_Comp)	The typical, minimum, and maximum values for the die capacitance in the IBIS data file. The die capacitance values are listed under c_comp. Die capacitance is measured in `pF` (picoFarads).
Die Cap. PowerClamp	The die capacitance of power clamps.
Die Cap. GroundClamp	The die capacitance of ground clamps.
Die Cap. Pullup	The die capacitance of pullup resistors.
Die Cap. Pulldown	The die capacitance of pulldown resistors.
Reference Temperature	The reference temperature value. The reference temperature is typically 50 degrees, with a minimum default of 0 degrees and a maximum default of 100 degrees. The voltage and current values for the Power Clamp, Ground Clamp, Pull Up, and Pull Down VI curves that you specify elsewhere in the IOCell editor correspond to the reference temperature you specify here.

Output Information

Figure 9-10 IOCell Object Editor - Output Information

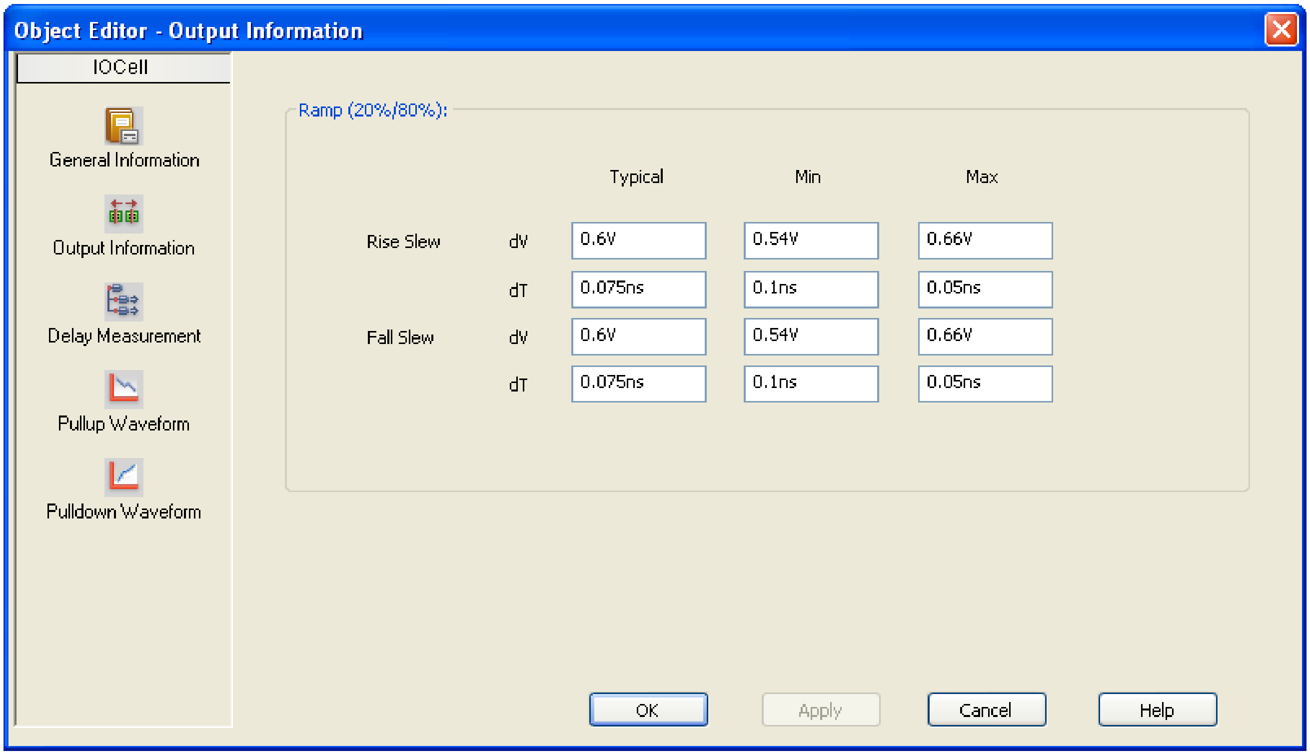


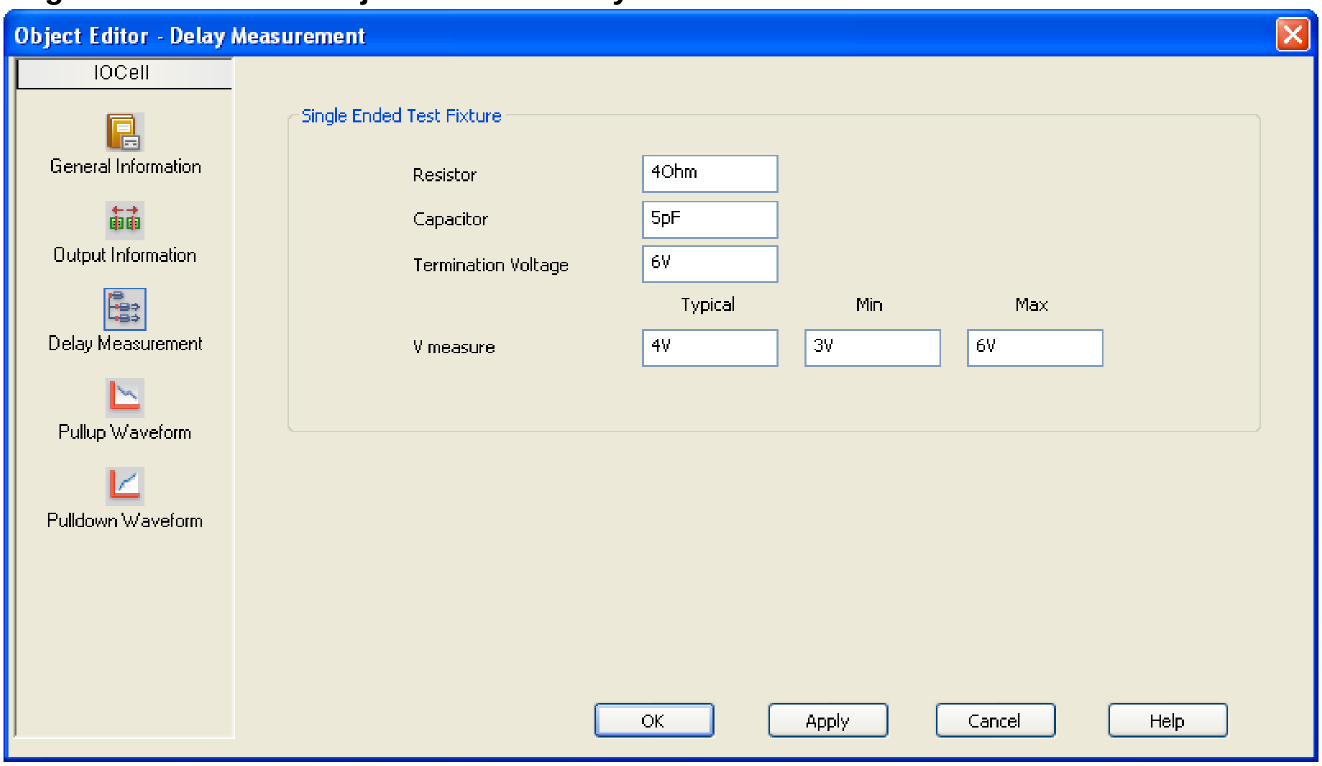
Table 9-3 Object Editor - Output Information

Option	Description
Rise Slew/Fall Slew	Specify minimum, typical, and maximum values for ramp rates associated with the IOCell model.
	By IBIS convention, the ramp rate values (the dV/dT values or slew rates) are required to be 20%/80% values. This implies that the rise time and the fall time are defined as the time it takes the output buffer to go from 20% of its final value to 80% of its final value.
	You must also correctly categorized ramp rate values as minimum, typical, and maximum slew rates. The minimum value is the slowest slew rate and the maximum value is the fastest.
dV	Represents the difference between 20% and 80% of the actual voltage swing.

Option	Description
dT	Represents the actual time taken for the 20%/80% voltage swing.

Delay Measurement

Figure 9-11 IOCell Object Editor - Delay Measurement



The Delay Measurement page contains the test fixture data used when measuring buffer delays for rising and falling drivers (output buffers).

If you are defining information for single-ended test fixtures but your output model does not contain that information, you need to populate the Resistor, Capacitor, and Termination Voltage, and V Measure fields.

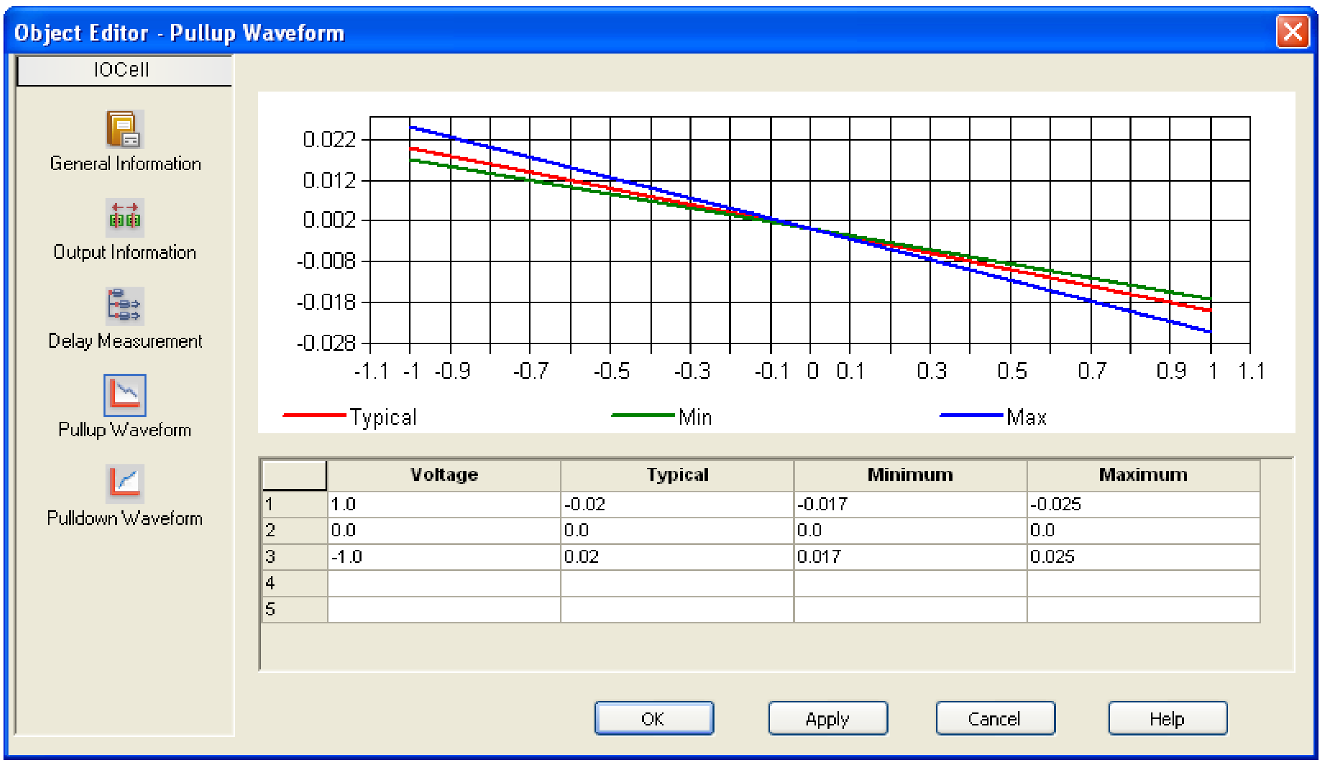
Table 9-4 Object Editor - Delay Measurement

Option	Description
Resistor	Displays the test fixture value for resistance.

Option	Description
Capacitor	Displays the test fixture value for capacitance.
Termination Voltage	Displays the test fixture value termination voltage.
V measure	Displays the reference voltage.

Pullup Waveform

Figure 9-12 IOCell Object Editor - Pullup Waveform



The Pullup Waveform/Pulldown Waveform page consists of the following information:

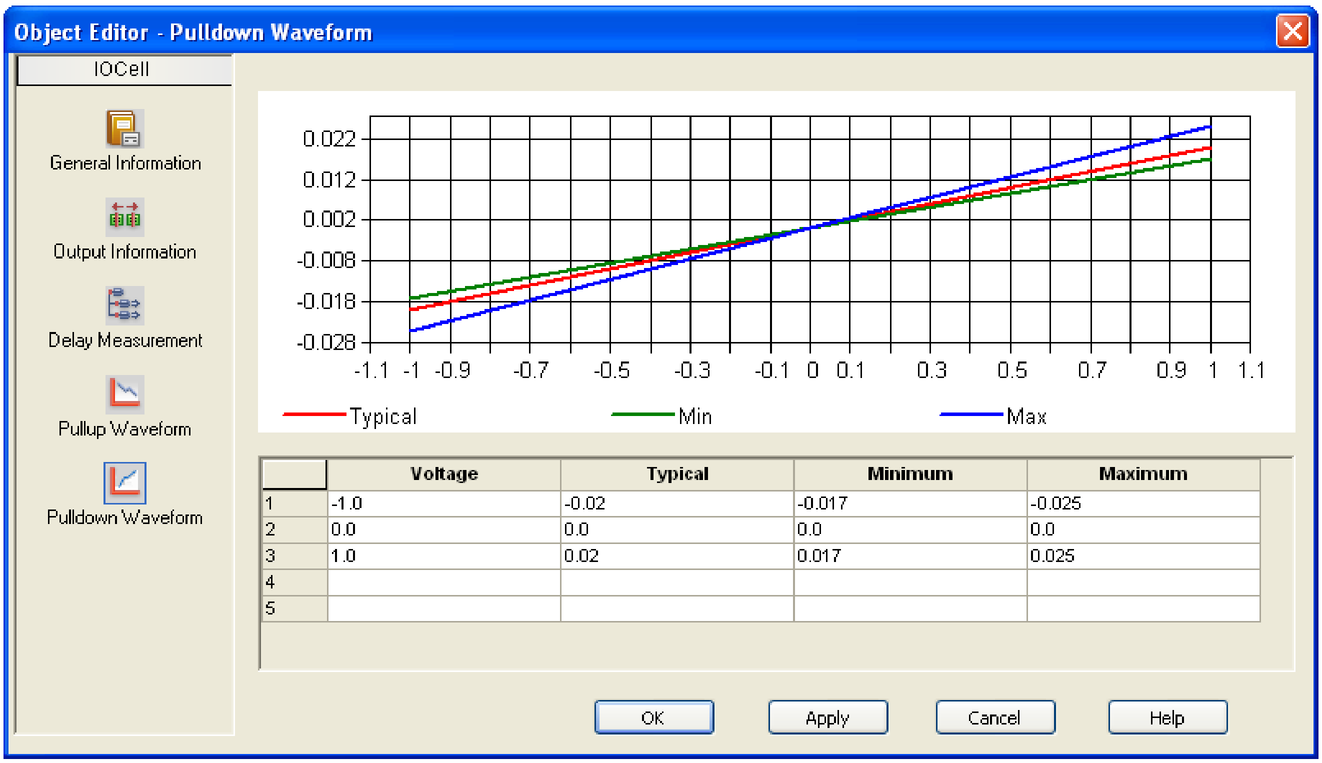
- Waveform chart
- Waveform table data:
  - Voltage
  - Typical
  - Minimum

- ❑ Maximum

In the current release, you can only view the waveform data. To change the curve, you need to manually edit the TEXT file for the model.

Pulldown Waveform

Figure 9-13 IOCell Object Editor - Pulldown Waveform



	Voltage	Typical	Minimum	Maximum
1	-1.0	-0.02	-0.017	-0.025
2	0.0	0.0	0.0	0.0
3	1.0	0.02	0.017	0.025
4				
5				

The Pulldown Waveform page consists of the following information:

- Waveform chart
- Waveform table data:
  - ❑ Voltage
  - ❑ Typical
  - ❑ Minimum
  - ❑ Maximum

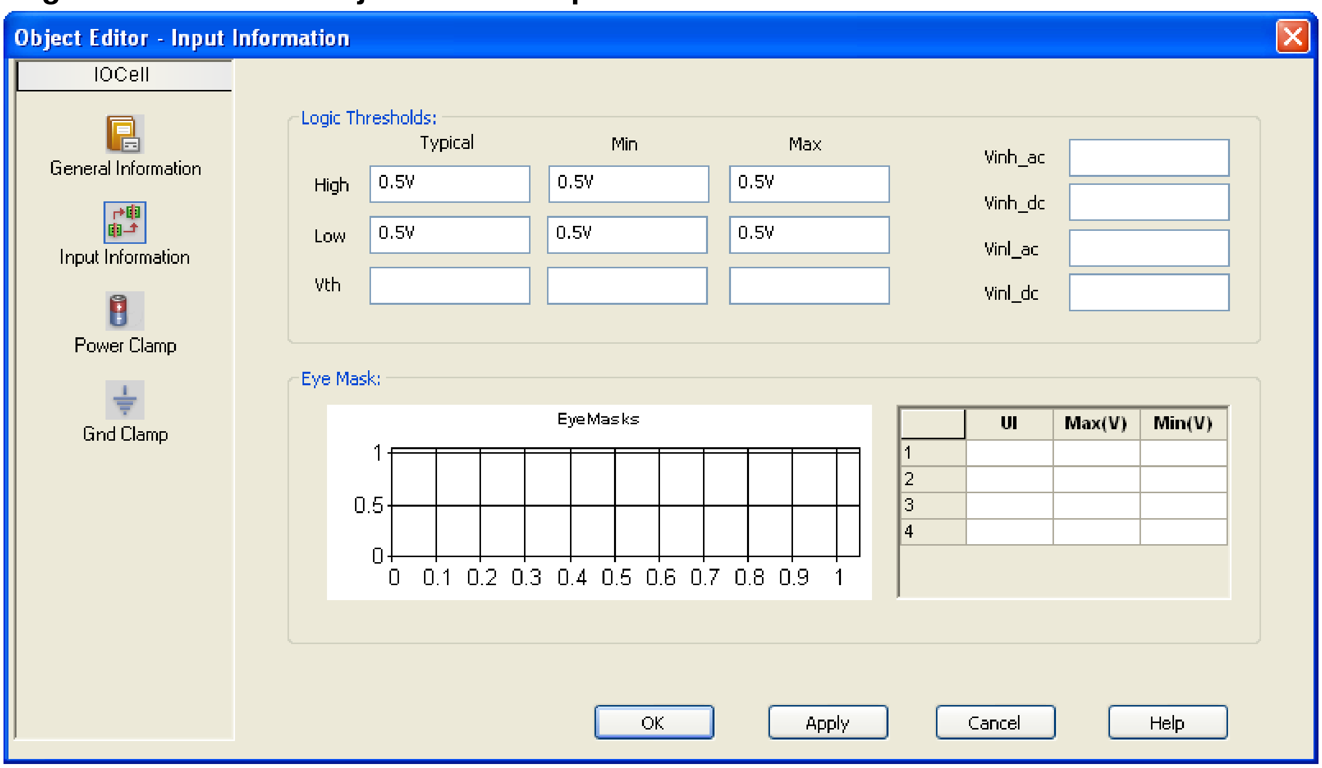
In the current release, you can only view the waveform data. To change the curve, you need to manually edit the TEXT file for the model.

Input Information

For an IbisInput model type, the following additional pages are displayed:

- Input Information
- Power Clamp
- Gnd Clamp

Figure 9-14 IOCell Object Editor - Input Information



The Input Information page consists of the following information:

- Logic Threshold
  - High Voltage
  - Low Voltage
  - Threshold Voltage

- ❑ Vinh_ac – input threshold voltage that a low-to-high input waveform must reach to ensure that the receiver's output has changed state.
- ❑ Vinh_dc – input threshold voltage that an input waveform must remain above to ensure that a receiver’s output will not change state.
- ❑ Vinl_ac – input threshold voltage that a high-to-low input waveform must reach to ensure that the receiver’s output has changed state.
- ❑ Vinl_dc – input threshold voltage that an input waveform must remain below to ensure that the receiver’s output will not change state.

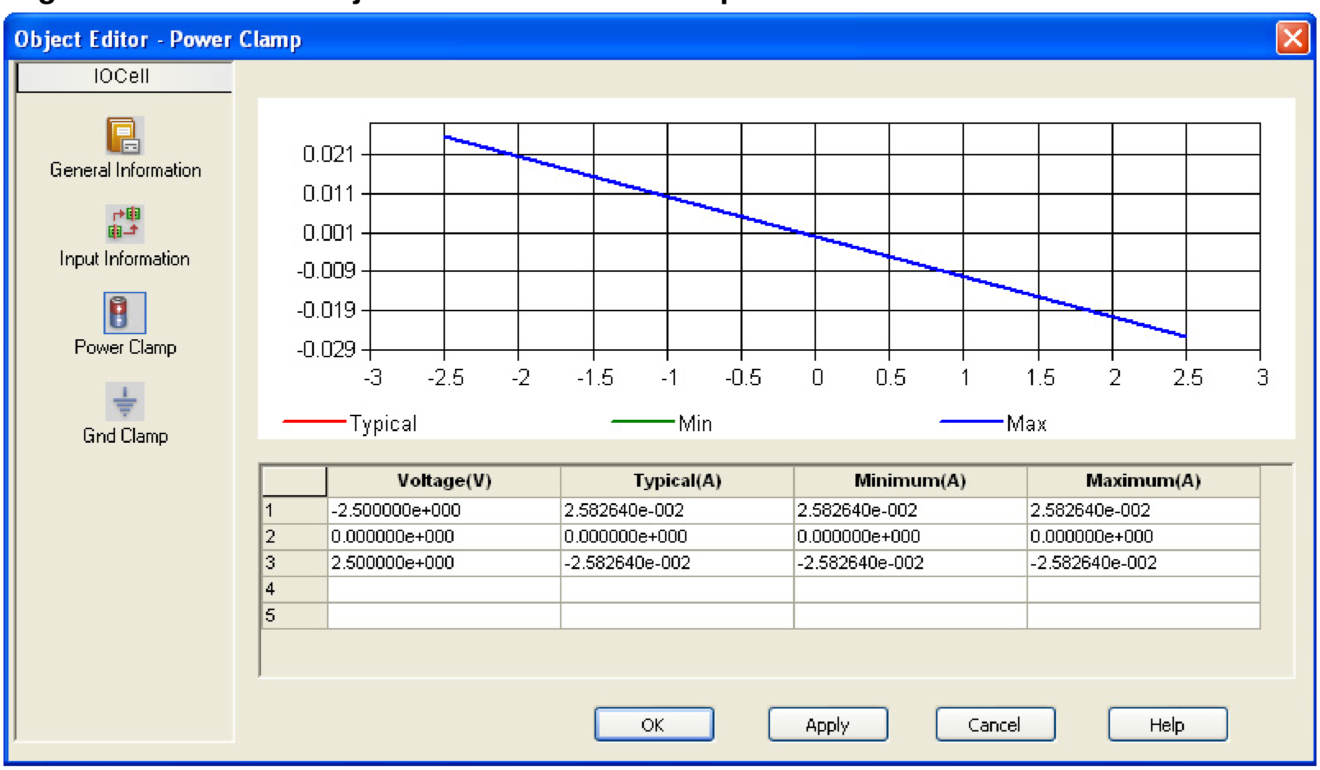
- Eye Mask – Displays the waveform of the eye mask saved as a `.sim` file in the Eye Diagram mode.

  Note: An eye mask lets you specify the acceptable parameters for what an eye should look like in order to extract clock transmissions and high-speed data to buffer models.

Depending on the model, IbisInput can also include *Rising Waveform* and *Falling Waveform*.

Power Clamp

Figure 9-15 IOCell Object Editor - Power Clamp



	Voltage(V)	Typical(A)	Minimum(A)	Maximum(A)
1	-2.500000e+000	2.582640e-002	2.582640e-002	2.582640e-002
2	0.000000e+000	0.000000e+000	0.000000e+000	0.000000e+000
3	2.500000e+000	-2.582640e-002	-2.582640e-002	-2.582640e-002
4				
5				

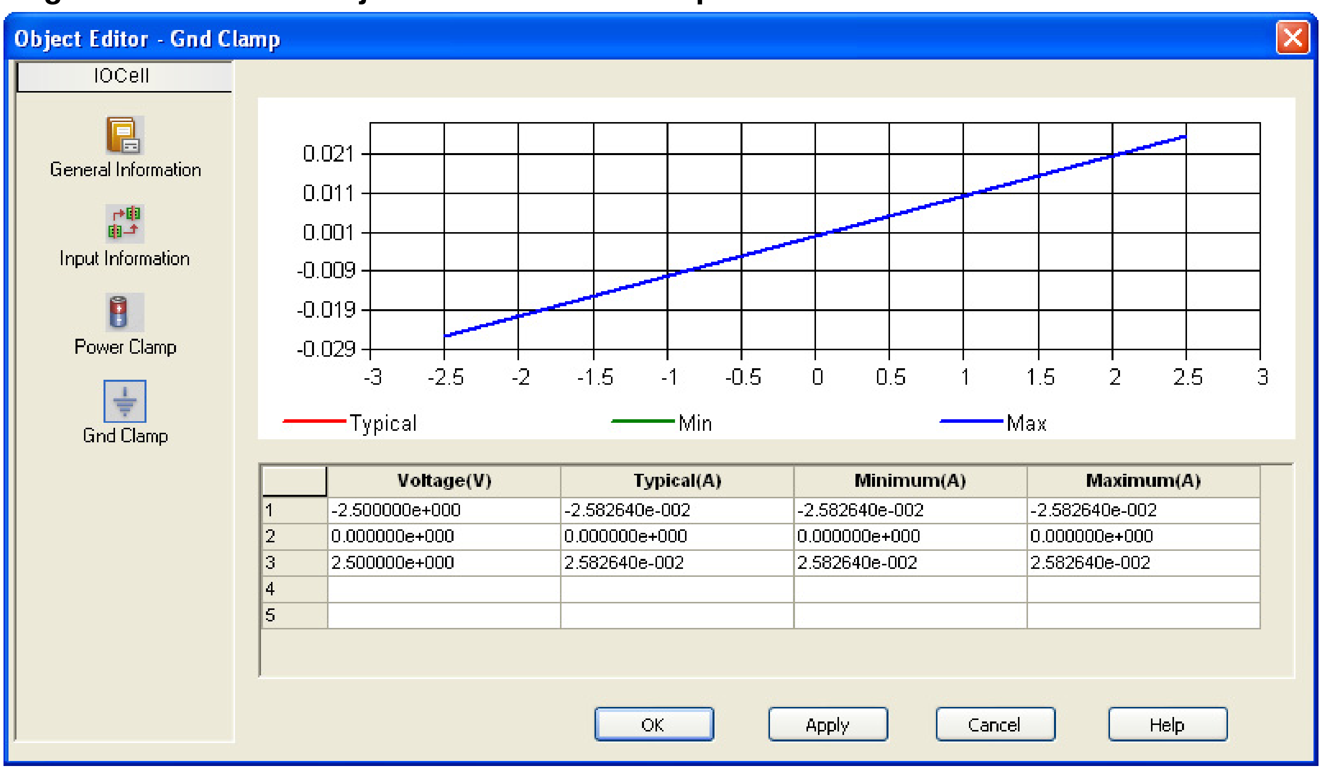
The Power Clamp page consists of the following information:

- Waveform chart
- Waveform table data:
  - Voltage
  - Typical
  - Minimum
  - Maximum

In the current release, you can only view the waveform data. To change the curve, you need to manually edit the TEXT file for the model.

Ground Clamp

Figure 9-16 IOCell Object Editor - Gnd Clamp



The Gnd Clamp page consists of the following information:

- Waveform chart

- Waveform table data:
  - Voltage
  - Typical
  - Minimum
  - Maximum

Note: In the current release, you can only view the waveform data. To change the curve, you need to manually edit the TEXT file for the model.

Rising Waveform/Falling Waveform

The Rising and Falling Waveform pages consist of the following information:

- Waveform chart
- Sub waveform selection
- Test package R C L value
- V/T curve test fixture R C L and V value
- Waveform table data

Note: In the current release, you can only view the waveform data. To change the curve, you need to manually edit the TEXT file for the model.

10

Device Modeling

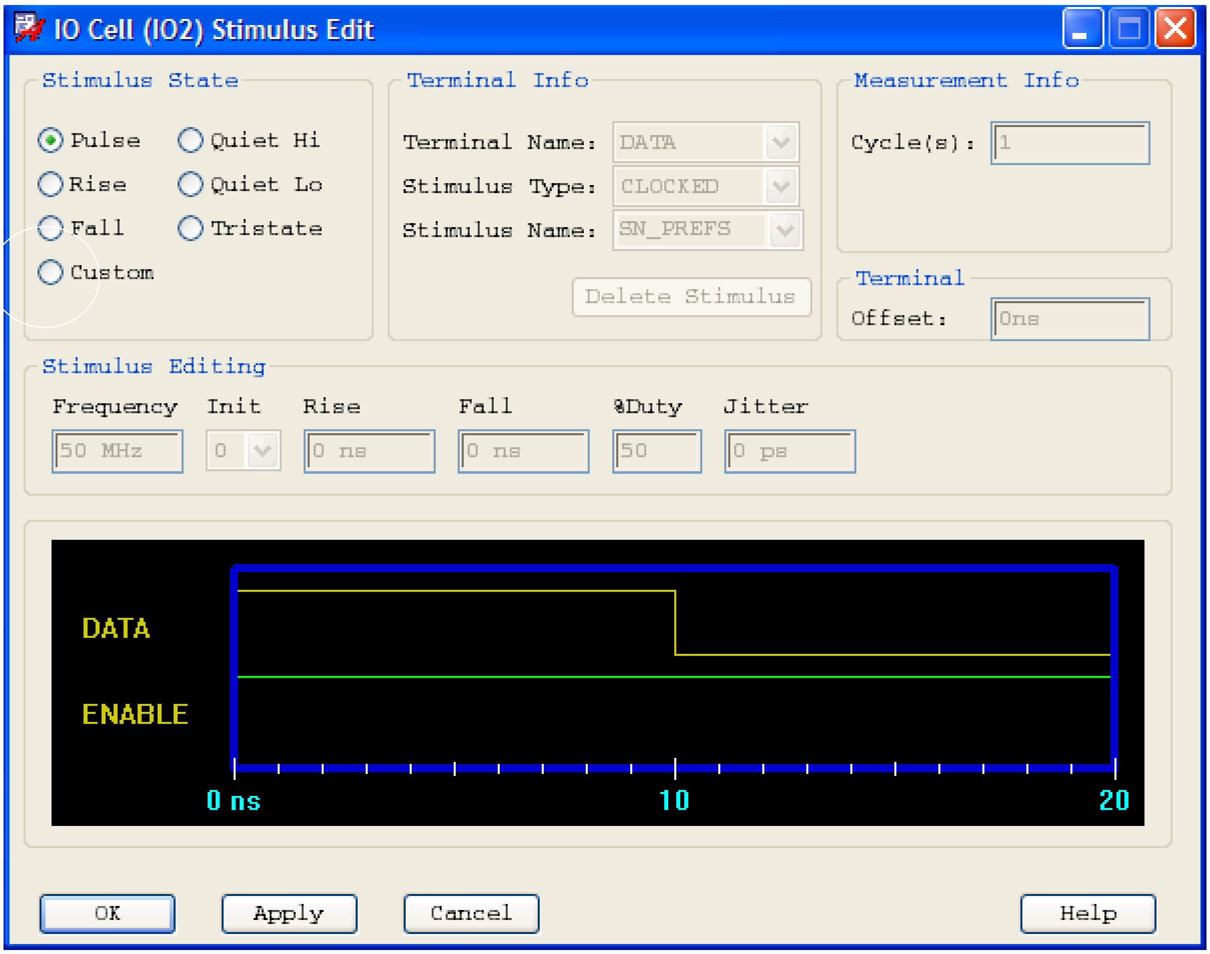
IOCell

Topics in this chapter include:

Introduction

For both pre-defined and custom stimuli, you use the *IO Cell Stimulus Edit* dialog box to change the stimulus associated with an IOCell and view the waveform. You can also define the characteristics of a custom stimulus.

Figure 10-1 IO Cell Stimulus Editor



The stimulus state assigned to the IOCell determines the stimulus state applied to an IOCell during simulation. You choose from six pre-defined stimulus states and a customizable stimulus state.

- Pulse
- Rise
- Fall
- Quiet Hi

- Quiet Lo
- Tristate
- Custom

The pre-defined stimulus states derive their stimuli from parameters set in the Analysis Preferences dialog box. You define the Custom stimulus in the Stimulus Editor. The default stimulus state for an IOCell is Tristate.

You use the IOCell Stimulus Edit dialog box to modify the stimulus state associated with an IOCell, or buffer symbol in the canvas. The associated stimulus applies to the IOCell when you perform simulation.

➡ To display the IOCell Stimulus Editor, click the stimulus associated with an IOCell symbol.

For more information on the editor, see the *SigXplorer Command Reference*.

Tip

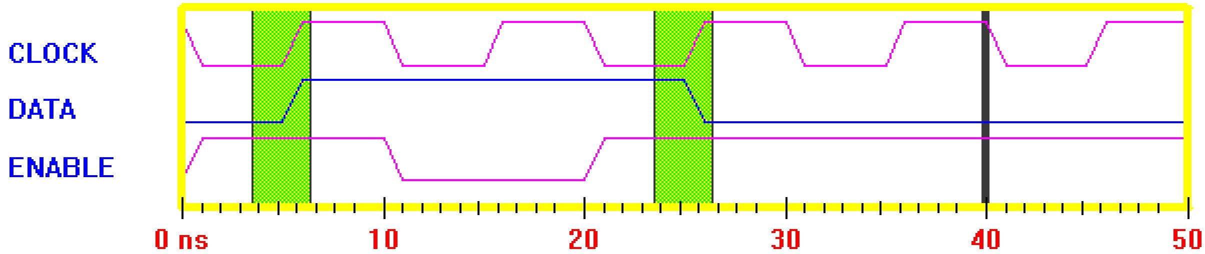
You save and recall stimulus vectors for all IOCell drivers in the topology with the *Setup – Vectors* command.

When you select *Custom* in the Stimulus State area, you activate all areas for defining the Custom stimulus. With the editing areas active, you can define the Custom stimulus in the Stimulus Editor. The default values for a Custom stimulus populate from parameters set in the Analysis Preferences dialog box.

Viewing the Waveform for a Stimulus

Waveforms for each IOCell terminal appear at the bottom section of the Stimulus Editor.

Figure 10-2 The Stimulus Editor Waveform



For both pre-defined and custom stimuli, a reference data clock waveform aids you in defining data stimuli. The initial frequency of this waveform comes from the frequency specified in the Analysis Preferences dialog box.

The following defines the signal color in the Waveform Display:

■ Yellow	Current signal you are editing.
■ Red	Signal that you cannot edit.
■ Green	Signal that is editable but is not active.
■ Purple bands	Setup and hold margins.
■ Gray vertical markers	Simulation run time.

Defining Terminal Information

Terminal information for an IOCell symbol comes from the IOCell buffer model associated with the IOCell symbol based on the following criteria.

IOCell Type	Terminal
Output (only)	Only the data terminal defines the stimuli.
Bi-directional	Both the Data and Enable terminals define stimuli.
Clocked IO	Clock, Data, and Enable terminals define the stimuli.
MacroModel definition	Each terminal, except for Data and Enable, defines the stimuli named in the PinTerminalsMap of the MacroModel definition.

Frequency and Pattern

For Periodic and Synchronous stimuli, *Frequency* and *Pattern* specify the frequency of the stimulus reference clock and the bit pattern used for the stimulus. For a Synchronous stimulus, click *Random* to generate a random bit pattern of a specified length. The default value for *Frequency* comes from the Clock Frequency Analysis Preference.

Init, Switch Times, and Switch At

For the Asynchronous stimulus, *Init* and *Switch Times* specify the initial state for the stimulus and the times at which the stimulus switches. Enter a list of switch times separated by spaces.

For the Synchronous stimulus, *Init* and *Switch At* specify the initial state for the stimulus and the point at which the stimulus transitions; on the rising edge, the falling edge, or on both edges.

Tr and Tf

For all stimuli, *Tr* and *Tf* display the transition rise and fall times. *Tr* or *Transition Rise Time* displays the time it takes the signal to transition from the low state to the high state. *Tf* or *Transition Fall Time* displays the time it takes the signal to transition from the high state to the low state.

Rise and Fall

For all stimuli, *Rise* and *Fall* display the transition rise and fall times. *Rise* displays the time it takes the signal to transition from the low state to the high state. *Fall* displays the time it takes the signal to transition from the high state to the low state.

% Duty

For the Clocked stimulus, *%Duty* sets the percentage of time that the clock signal is high in a single clock cycle. For example, *50* represents equal high and low periods of the cycle. The default value for *%Duty* is taken from the Duty Cycle Analysis Preference.

Jitter

For the Clocked stimulus, *Jitter* sets the time period for variations between system clock cycles at an IOCell pin. *Jitter* is the potential for a single-cycle narrowing of the clock period. For example, a *10 ns* cycle with *10 ps* of jitter could result in a potential cycle time of *9.99* to *10 ns*.

In a perfect network, the system clock always arrives exactly on time (one clock period from the end of the previous clock cycle at a given IOCell pin). In reality, the clock may not arrive exactly on time, but may arrive earlier or later than expected.

Clock jitter describes the variations in clock edges as the clock signal travels through the electronic components in the circuit. For example, for a clock period of *20 ns*, a jitter value of *10 ps*, rise and fall times of *1 ns*, and a *duty cycle of 50%*, the clock edges can fall within the following time ranges:

1st rising edge	-0.005ns to 0.005ns
1st falling edge	9.990ns to 10.010ns

Jitter can be coherent and incoherent in the following situations. Jitter is coherent when the clock instances have the same distribution, resulting in identical clock cycles. This is typical of a central clock generator where jitter is shared by all receivers. Jitter is incoherent when clock instances exhibit independent jitter. This is typical of receivers clocked from different sources.

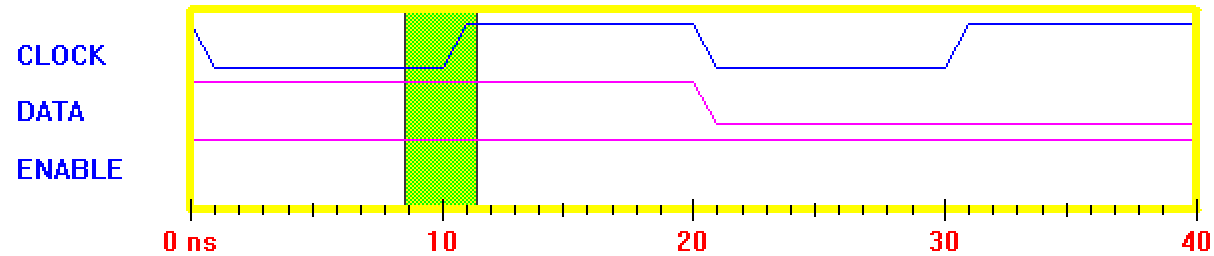
Defining Measurement Information for Custom and Tristate Stimuli

For Custom and Tristate stimuli associated with a clocked IOCell MacroModel, you can set *Cycles*, *Setup* and *Hold* in the Measurement Info area.

- In *Cycles*, specify the clock pulse at which to make measurements. At a minimum, the simulation runs for the highest specified cycle number. The default value for *Cycles* propagates from the Measurement Cycle Analysis Preference.
- In *Setup*, set the desired settle time for the data signal so it can latch on a synchronous device on the next clock cycle. You use *Setup* to determine if a signal arrives too late.
- In *Hold*, set the time for a signal to remain latched at one synchronous device before launching to a second synchronous device. You use *Hold* to determine if a signal arrives too early.

You use *Setup* and *Hold* to stabilize a data signal at a synchronous device, so that the data signal latches in a predictable state when the clock edge arrives. Figure 10-3 illustrates sample *Setup* and *Hold* margins.

Figure 10-3 Sample Setup and Hold Margins



For a Tristate stimulus not associated with a clocked IOCell MacroModel, you can set only the Measurement Cycle value. In the Measurement Info area, specify the clock pulse at which to make measurements. At a minimum, the simulation runs for the highest specified cycle number. The default value for *Cycles* comes from the Measurement Cycle Analysis Preference.

Defining Terminal Offset and Skew for Custom Stimuli

In the Terminal area, enter the offset value, the launch time for the arrival of the stimulus at the IOCell input pin. The default value for *Offset* comes from the Offset Analysis Preference.

In the Terminal area, enter the skew value, the clock latency for an IOCell. The Skew value shifts the clock stimulus.

Editing a Custom Stimulus

For a Custom stimulus state, you can use the Stimulus Editor to:

- Select the Custom stimulus state, if necessary (in the Stimulus State area)
- Edit the Custom stimulus (in the Stimulus Editing area)
- Specify terminal information (in the Terminal Info area)
- Specify cycles (in the Measurement Info area)
- Specify terminal offset (in the Terminal area)
- View the waveform associated with the Custom stimulus (in the waveform display)

To enter multiple values for simulation sweeping, use the:

- *Linear Range* to enter a range of values by specifying *Start* and *Stop* and a step size for iterating from the start value to the stop value.

- *Multiple Values* to enter a list of values by specifying a list of discrete number values.
- *Expression* to enter an expression by specifying an expression string composed of operators, functions, and references to other parameters.

ESPICE Device Models

Topics in this chapter include:

Introduction

You use multi-terminal ESpice models for simulation to support advanced analysis using externally developed SPICE models. For example, a wire bond model created by an external 3D engine. Symbols for these devices occur dynamically, according to the number of terminals in the model source file. Simulating a topology containing these devices, creates simulator input files, as required, providing the needed circuit builder support.

Setting Pin Order

The pin order of the model defines the location of the port on the blackbox. You can edit the pin order in SigXplorer for better readability in Model Editor.

The following is an example of the text describing the pin order of the ESpice Device model Figure 10-4.

```
("espice.dml"

(PackagedDevice

(BP_CONN

(ESpice ".subckt BP_CONN DC1 MB1 DC2 MB2 <= this order defines left,
right, left, right from top to bottom
```

Figure 10-4 Example of pin order on an ESpice Device

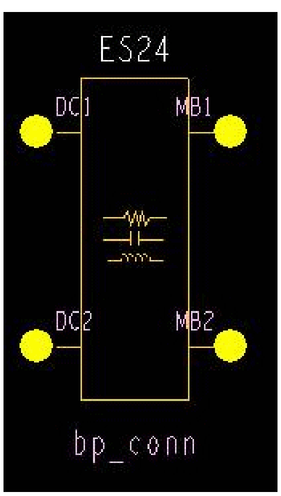
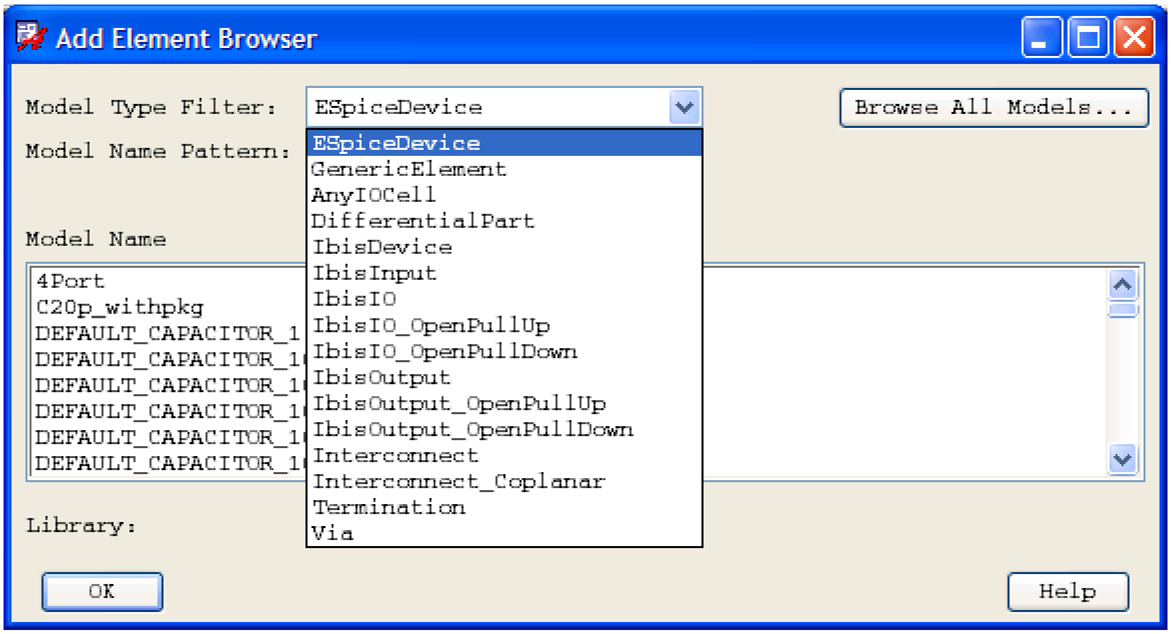


Figure 10-5 shows the selection of a user-defined multi-terminal ESpice from the Add Element Browser.

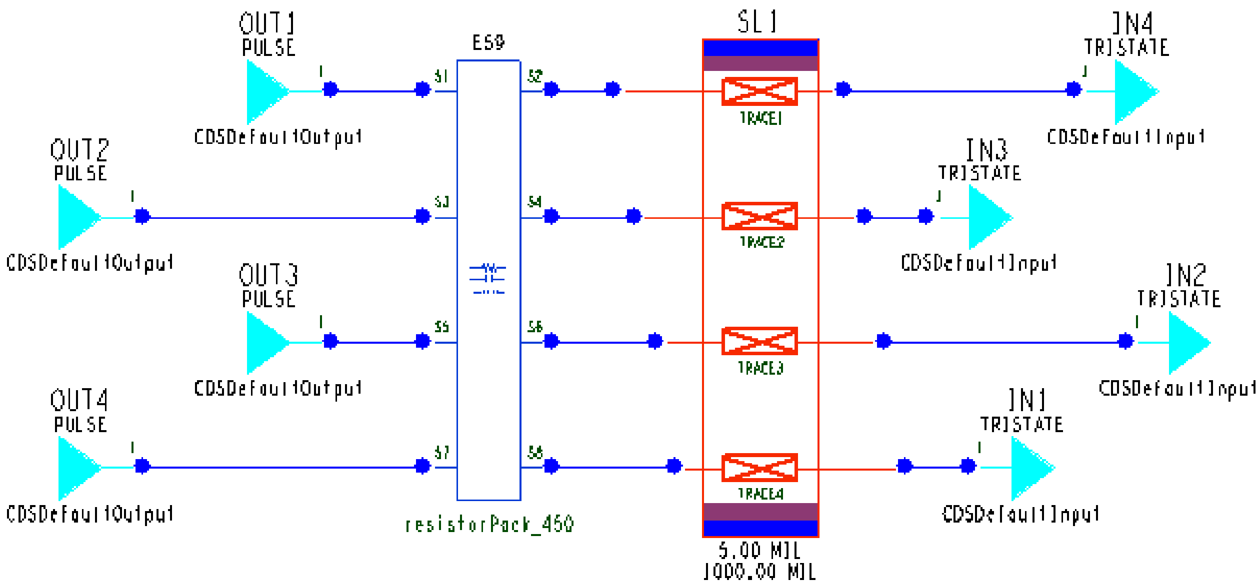
Figure 10-5 Selection of a Multi-terminal ESpice Device in the Add Element Browser



If you assign a multi-pin ESpice device to a part in SI, a subset model generates when extracting. For example, if pins S7 and S8 were in the net, SigXplorer creates an ESpice device, including the two pins. You could then reapply the topology with that subset model.

Figure 10-6 shows a user-defined multi-terminal ESpiceDevice and its use within a sample topology.

Figure 10-6 Sample Multi-terminal ESpice Topology



11

Interconnect Modeling

Topics in this chapter include

Introduction - Etch

Coupled trace interconnect models and simulation sweeping in SigXplorer allow exploration of the electromagnetic-coupling behavior of PCB traces.

Coupled trace part models with two to six traces are available from the Add Element Browser for both Microstrip and Stripline layer stackups. As with other part models, coupled trace part models are user-definable. Embedded (dual) microstrip and stripline coupled trace parts are available to support broadside-coupled differential pairs. These parts currently support single traces on adjacent layers.

To determine acceptable parallelism rules, you can perform reflection, crosstalk, EMI, or custom simulation sweeps of the coupled trace part model parameters, including trace width, spacing between adjacent lines, and offset between line centers. To determine appropriate manufacturing tolerances, you can sweep layer stackup parameters (dielectric constant and thickness, and trace width and thickness) to explore potential layer stackup possibilities.

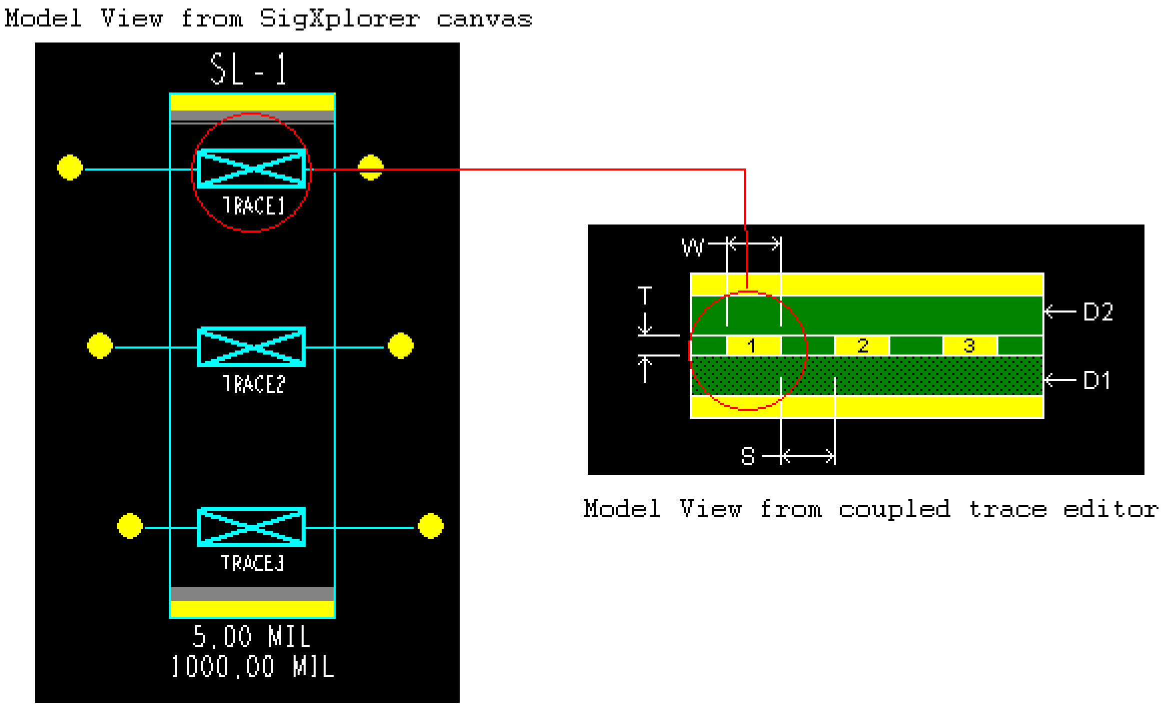
You define stimuli for victim and aggressor net IOCells using the IOCell Stimulus Editor.

Coupled Trace Symbols

Coupled trace symbols appear in the canvas as individual traces enclosed in a bounding rectangle. Colored bands across the top and bottom (yellow bands represent the shield/plane layers while green bands represent the dielectric layers) graphically indicate if the coupled-trace part is microstrip or stripline.

Figure 11-1 represents a three-trace stripline geometry coupled trace symbol as viewed from the canvas (stacked vertically) and from the View Trace Model Parameters dialog box (shown horizontally).

Figure 11-1 Viewing Trace Model Parameters



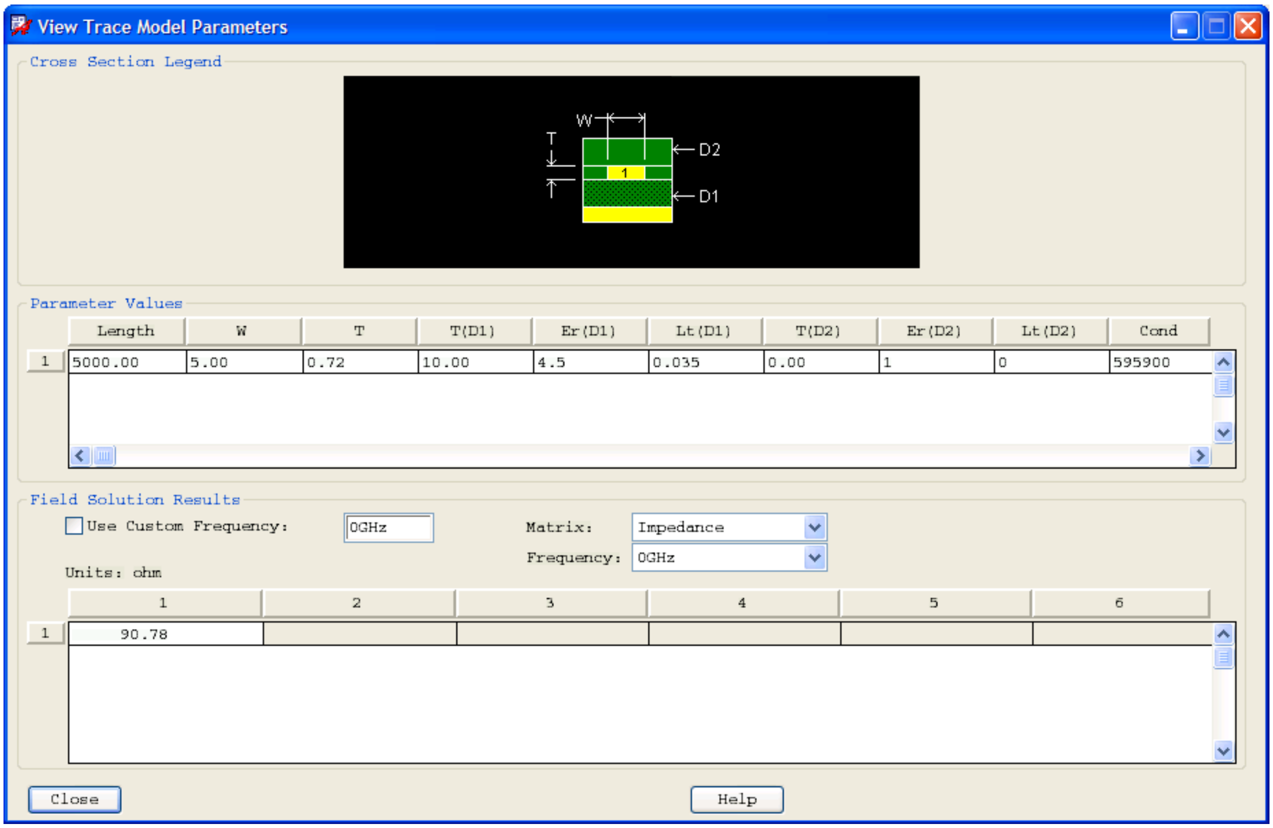
To invoke the View Trace Model Parameters dialog box, do the following:

1. Position your cursor over a trace name or one of its values in the Parameters tab of the SigXplorer spreadsheet.
2. Right-click to display the *View Trace Parameters* button and click it.

Tip

You can also right-click the trace on the canvas and choose *View Trace Model Parameters* from the pop-up menu to open the View Trace Model Parameters dialog box.

Figure 11-2 View Trace Model Parameters Dialog Box



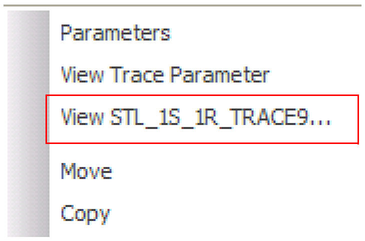
From this dialog box you can view and explore the following:

- Cross section view detailing the orientation of the traces and dielectric layers within the trace or coupled trace part model in the stackup
- Parameter values for the trace or coupled trace part model
- Spreadsheet display of field solution data for the trace or coupled trace part model. You can control display of data by selecting the parameter value set, Field Solver Cutoff Frequency and the impedance matrix for which to display data.

Associating a Trace Model

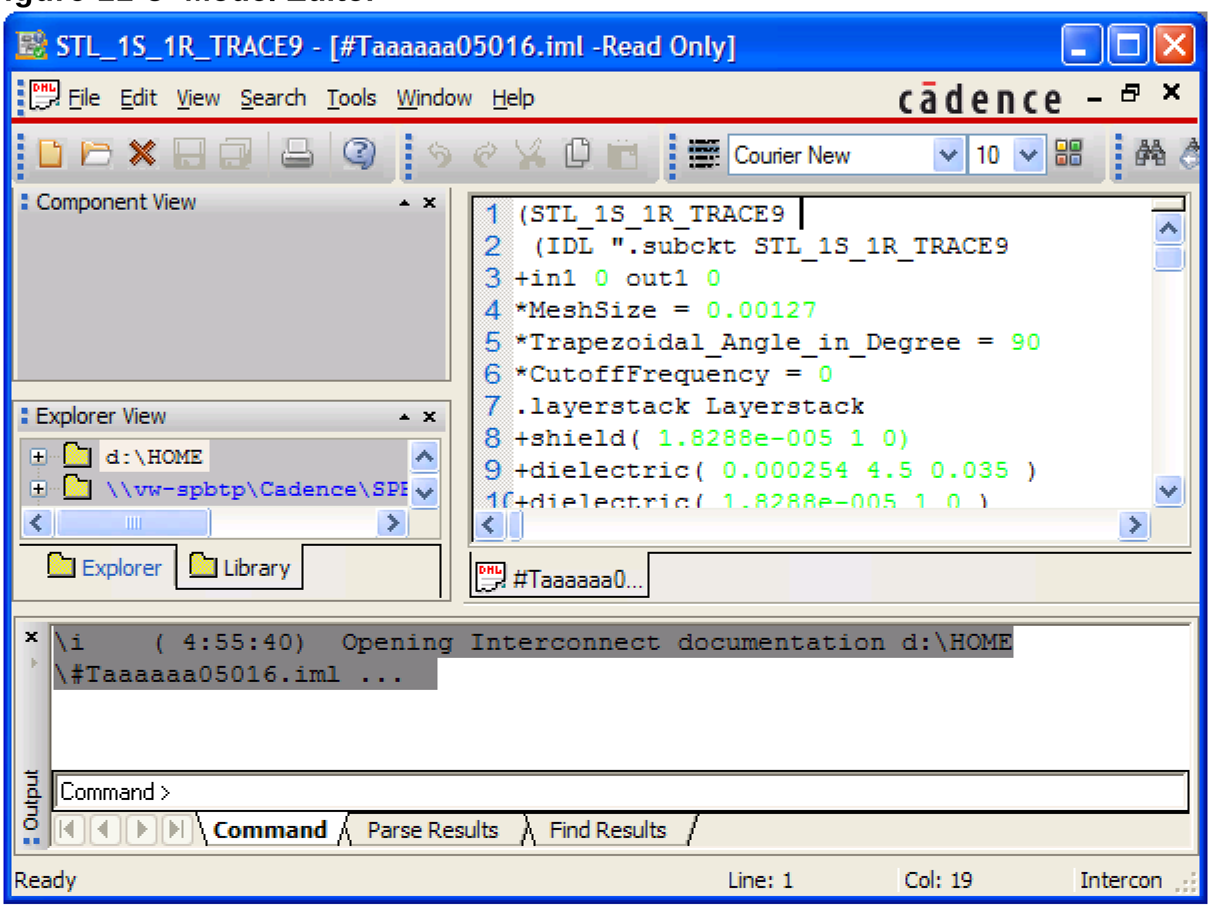
To associate a trace model symbol in the canvas to a specific trace model in the Interconnect library, do the following:

➡ Right-click a trace and select the trace model name from the pop-up menu.



The field solved model for the trace is displayed in Model Editor.

Figure 11-3 Model Editor



Model Editor makes it easy to associate a trace model symbol in the canvas to a specific trace model in the Interconnect library.

Exploring Topologies Containing Coupled Trace Models

For early topology exploration (pre-part placement), you can select IOCell and coupled trace parts from the Add Element Browser to create a topology in SigXplorer.

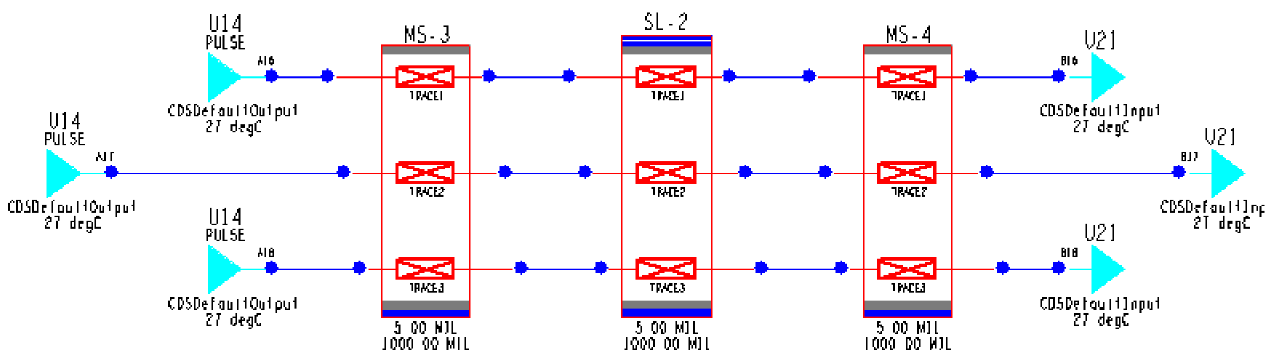
For post-part placement, you can extract a net from a routed board into the canvas, and then replace the interconnect with the appropriate coupled trace models. You can then experiment in the canvas with trace length, spacing and layer stackup parameters to derive a topology that satisfies the target design requirements.

You use a coupled trace symbol to model the crosstalk between lines of different logic levels or families. For example, `TTL` and `ECL`. You also explore the crosstalk of up to three differential pairs, whether edge-coupled or broadside-coupled.

Sample Coupled Trace Circuit

This circuit (Figure 11-4) uses three-trace microstrip and stripline coupled traces to model a bus net from one surface-mount IO pin to another. In this example, the signal travels from the top layer, through a stripline layer, back to the top layer.

Figure 11-4 Sample Coupled Trace Circuit



Adding a Coupled Trace Model

To add a coupled trace model:

1. Choose *Edit – Add Element*.

   The Add Element Browser dialog box appears.

2. From the Model Type Filter drop-down list, select *Interconnect*.
3. Select the desired coupled trace model and drag it to the canvas for placement.
4. Right-click on the trace model and select *End Add* from the pop-up menu.
5. Click *OK* to close Add Element Browser.

Editing a Coupled Trace Model

For a coupled trace part, you edit the following parameter values in the spreadsheet Parameters tab:

- RefDes for the coupled trace part.
- Thickness and dielectric constant for each dielectric layer.
- Trace spacing, length, width, and thickness.

To modify RefDes

1. In the canvas, select the RefDes or part name, above the coupled trace symbol.

   The Parameters tab opens with the data for the selected coupled trace part expanded and the RefDes in the Attribute column highlighted for editing.

2. Specify the new RefDes and press *Enter*.

   The new RefDes replaces the old in both the Attribute column of the spreadsheet and above the part symbol in the canvas.

To modify the thickness or dielectric constant for the dielectric layers

1. Position your cursor over a trace name or one of its values in the Parameters tab of the SigXplorer spreadsheet.
2. Right-click to display the *View Trace Parameters* button, then click the button.

The Parameters tab in the spreadsheet opens with the data for the selected coupled trace part expanded.

The View Trace Model Parameters dialog box opens with the following information:

- Detailed cross-sectional view of the trace model
- Parameter values associated with the trace model
- Parasitic information for the trace model

3. In the *Value* column of the spreadsheet Parameters tab, click in the cell to the right of the dielectric constant or dielectric layer thickness whose value you want to edit. Then click the icon () that appears to the right in the cell.

   The Set Parameter dialog box appears for the selected attribute.

4. In the Set parameters dialog box, specify the new values.

   a) To specify one value for a single simulation, use *Single Value* and specify one number value.

   b) To specify multiple values for simulation sweeping:

   - Use *Linear Range* to specify a range of values by providing start and stop values and a step size for iterating from the start value to the stop value.
   - Use *Multiple Values* to specify a list of values by providing a list of discrete number values.
   - Use *Expression* to specify an expression by providing an expression string composed of operators, functions, and references to other parameters.

5. In the Set Parameter dialog box, click *OK*.

   The new parameter values appear in both the canvas and the Parameters tab of the spreadsheet.

   The change appears in *Parameter Values* of the View Trace Model Parameters dialog box as well. SigXplorer adds a new line of data to the Parameter Values spreadsheet for each parameter value entered.

To view trace parasitics

1. In the canvas, click the trace length or trace spacing text below the coupled trace symbol.

   The attribute highlights in the spreadsheet.

2. Position your cursor over the attribute in the spreadsheet.
3. Right-click to display the *View Trace Parameters* button, then click the button.

   The View Trace Model Parameters dialog box opens with parasitic information displayed in the *Field Solution Results* area.

Simulating a Coupled Trace Model

To simulate a coupled trace circuit, the victim net is held at a non-switching (quiet hi or quiet lo) state while the aggressor nets drive the simulation.

To edit the stimulus on the victim net driver

1. Click the stimulus state text on the driving IOCell of the victim net.

   The IOCell Stimulus Editor appears.
2. In *Stimulus State*, select a non-switching state (either Quiet Hi or Quiet Lo).

   The data displayed in the IOCell Stimulus Editor changes to reflect the new stimulus state. The Stimulus field for the IOCell symbol is also updated in the canvas.

To edit the stimulus on the aggressor nets drivers

Edit the stimulus state associated with each aggressor net driver in turn.

1. Click the stimulus state text for a driving IOCell on an aggressor net.

   The IOCell Stimulus Edit dialog box appears.
2. In *Stimulus State*, select an appropriate stimulus state for each driver (Pulse, Rise, Fall, Custom, or Tristate).

   The data displayed in the IOCell Stimulus Editor changes to reflect the new stimulus state. The *Stimulus* field for the IOCell symbol updates in the canvas.

To modify simulation parameters

1. If required, use *Analyze – Preferences* to display the Analysis Preferences dialog box.
2. In the Analysis Preferences dialog box, specify the required analysis parameters.

The modified analysis parameter values display in the Analysis Preferences dialog box. The Cutoff Frequency value also displays in the View Trace Model Parameters dialog box.

To select measurements

1. Click to select the Measurements tab.
2. In the Measurements tab, select the Measurements to be made during simulation and the types of simulations to perform.

To perform the simulation

1. Use *Analyze – Simulate* to perform the simulation.
2. When prompted, select a receiver of the victim net.
3. If you are performing simulation sweeps, the Sweep Sampling dialog box appears. Confirm the information it supplies and click *Continue*.

   The Command tab displays messages during the simulation.

   When the simulation is complete, the Results tab displays the simulation result data.

Note: Once you have derived the parallelism rules, you can set them (*Setup – Constraints*) in your target topology and subsequently apply the topology (*File – Update Constraint Manager*) in Allegro SI to each bit of a bus.

Viewing Parameters and Field Solution Results for Trace Models

Use the View Trace Model Parameters dialog box to display the following information for a trace or coupled trace symbol selected from the canvas:

- Cross section view detailing the orientation of the traces and dielectric layers for the trace or coupled trace part model in the stack up
- Sets of parameter attribute values for the trace or coupled trace model
- Spreadsheet display of field solution data for the trace or coupled trace model.

Viewing Parameter Attribute Values

The *Parameter Values* area displays the sets of parameter attribute values to sweep at simulation time. You set and modify these values through the Parameters tab of the spreadsheet. When you change parameter values from the Parameters tab, the *Parameter Values* area updates.

Exploring Field Solution Data

Selecting one of the numbered parameter attribute value sets that appear in the *Parameter Values* area, calculates and displays the field solution data for that set of parameter attribute values.

You can further explore field solution results for individual parameter sets by selecting or entering field solver cutoff frequencies at which to recalculate field solution data. Select existing cutoff frequencies from the *Frequency* pulldown. Enter new cutoff frequencies in *Field Solver Cutoff Frequency*.

Changes you make to the *Field Solver Cutoff Frequency* while exploring field solution results are local to this dialog box. You must establish the *Default Cutoff Frequency* used during simulations from the Analysis Preferences dialog box. The Ems2d field solver also uses this value *unless* a different cutoff frequency is specified in the EMS2D Preferences dialog box.The *Default Cutoff Frequency* established in the Analysis Preferences dialog box is among those listed in the *Frequency* pull-down.

You can also change the spreadsheet display of field solution results by selecting the field solution matrix for which to display data. Use the *Matrix* pull-down menu to select a matrix from one of the following: Capacitance, Inductance, Modal Velocity, Admittance, Near-end

Coupling, Impedance, and Modal Delay. In addition, when the cutoff frequency is above 0 GHz, you can also select *Linear Resistance* and *Dielectric Conductance*.

To view Cross Section, Trace Model Parameters, and Field Solution Results

1. In the canvas, click the trace length or trace spacing text below the coupled trace symbol.

   The attribute highlights in the spreadsheet.

2. Position your cursor over the attribute in the spreadsheet.

3. Right-click to display the *View Trace Parameters* button, then click the button.

   The View Trace Model Parameters dialog box appears with the cross section, parameter values, and field solution results for the trace model.

4. In the *Parameter Values* area, click within a spreadsheet row to calculate and display the field solution data for that set of parameter attribute values. Field solution is performed at the Cutoff Frequency displayed in the Field Solver Cutoff Frequency field.

   The spreadsheet in the Field Solution Results area changes to display the field solution data for the matrix displayed in the Matrix field.

To explore Field Solution Results

You explore different field solutions by changing the cutoff frequency and displaying different matrices.

1. Use the Matrix pull-down menu to change the data displayed in the Field Solution Results spreadsheet.

   The following choices are available for all cutoff frequencies:

   Impedance, Capacitance, Inductance, Modal Velocity, Admittance, Near-end Coupling, and Modal Delay. At cutoff frequencies above 0 GHz, Dielectric Conductance and Linear Resistance are also available.

   The Field Solution spreadsheet changes to display the new matrix. The *Units* field above the spreadsheet displays the units used for the spreadsheet data.

2. Use the *Frequency* pull-down to select an existing cutoff frequency or enter a new value in *Field Solver Cutoff Frequency*.

   The field solver recalculates for the selected parameter values and cutoff frequency. The spreadsheet displays the field solution for the selected matrix.

The new cutoff frequency value is added to the *Frequency* menu. Changes to cutoff frequency made in this dialog box are local and are not preserved or used for simulations. Use the Analysis Preferences dialog box to set the cutoff frequency used for simulation.

Vias

Topics covered in this section:

Introduction - Vias

For multi-gigahertz designs, it is critical to model via structures accurately over a very high frequency range. Vias often represent some of the most significant discontinuities on PCB, package, and IC structures. Given their inherent 3D nature, they can cause severe signal integrity and EMI issues. In addition to signal degradation on the host net, via excitation of waveguide modes can propagate and radiate energy to neighbor nets and into space as well.

The via modeling capability currently available in SigXplorer (Allegro PCB SI GXL only) is accurate well into the GHz frequency range.

The electrical via model formats include:

- Narrowband
- Wideband
- Scattering parameters (S-parameters)

Via model types include:

- Single vias
- Coupled vias - signal, signal-and-ground, and signal-and-power

In SigXplorer, you can:

- create via models in SigXplorer
- add them as parts to a topology, perform what-if simulations,
- analyze the results using SigWave before committing to a PCB layout.

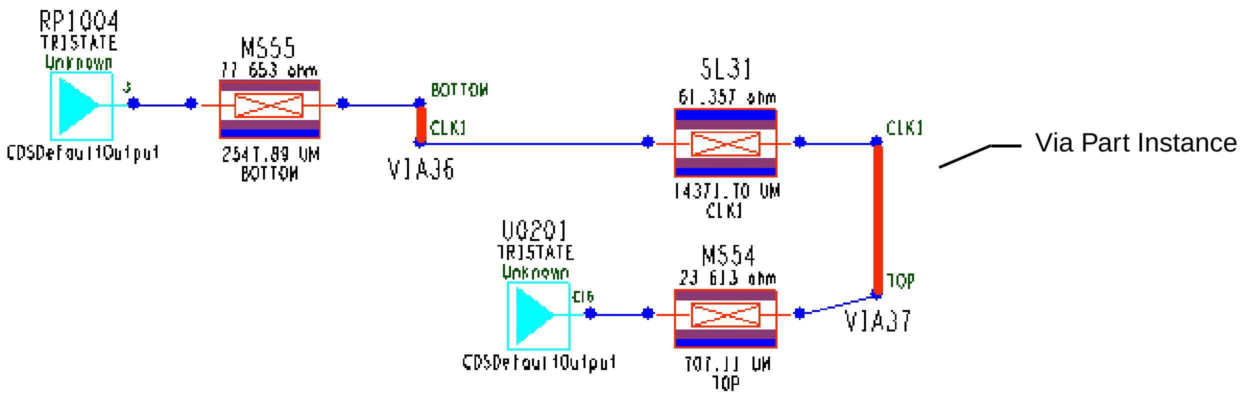
Important

Plated-through holes (PTHs) associated with component pins in Allegro (for example, PTHs for a backplane connector) are *not* seen as *vias* and are *not* automatically extracted into SigXplorer. You must add these structures manually using the Via Model Generator and include them for simulation accuracy.

Net Extraction and Via Models

In Allegro PCB SI GXL, when you extract a net topology from your PCB design into SigXplorer, any Closed Form via models associated with the interconnect are displayed in the canvas as via part instances, as shown in Figure 11-5 on page 210. Once available in SigXplorer, these via models can be upgraded interactively to one of the other available via model formats (S Parameter, Wideband or Narrowband) and used for advanced exploration. For further details, see the procedure for editing via models in the *SigXplorer Command Reference*.

Figure 11-5 Extracted Topology with Vias

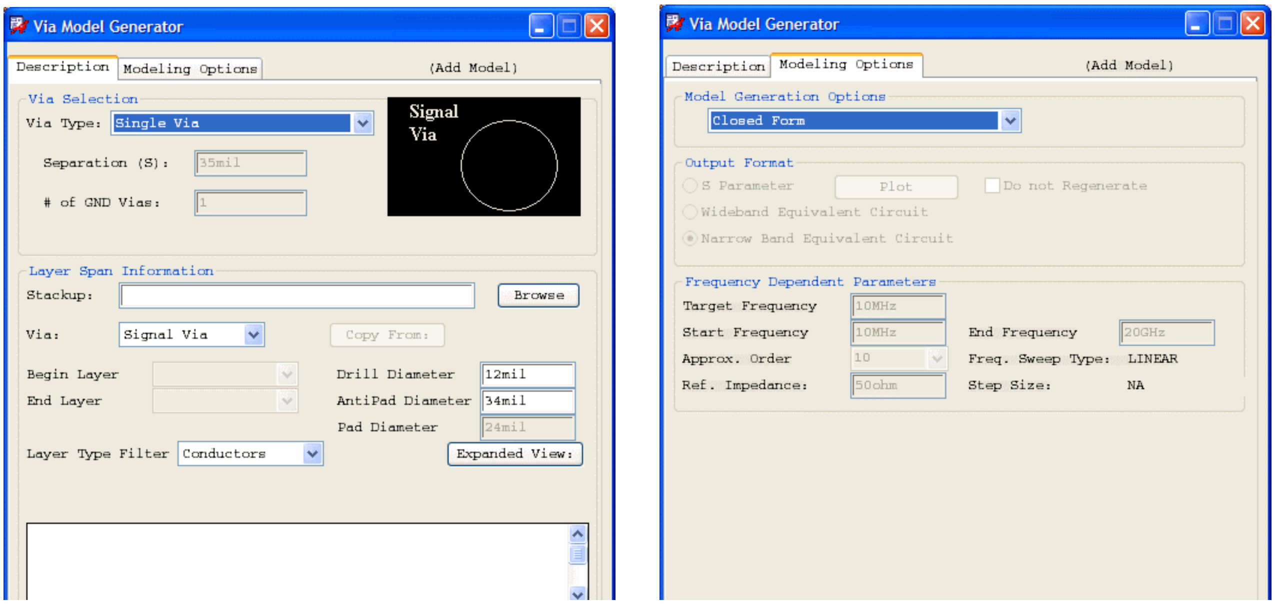


Via Model Generation

You can modify an existing via model or create one from scratch using the Via Model Generator dialog box shown in Figure 11-6.

Note: Before you create a new via model, be sure that the library you want to add it to is designated as the *working* library. For details on how to set the working interconnect library, refer to the procedures for working with libraries in the *SigXplorer Command Reference*.

Figure 11-6 Tabbed Via Model Generator Dialog Box



To access the Via Model Generator dialog box

➡ Choose *Analyze – Via Model Generation*.

For information on how to create or edit a via model, refer to the via modeling procedures in the *SigXplorer Command Reference*.

Via Model Formats

S-Parameter Format Details

- This is the most accurate via format. It should accurately capture the via behavior over the entire frequency range.
- Expect slower performance compared the circuit-based formats as more processing is required.
- *Start Frequency* for MGH applications is recommended at `10MHz`. If DC convergence issues occur, you can drop to `1MHz` (but no lower than `0.1MHz`).
- *End Frequency* should be about 2/t_rise (1/t_rise minimum). Go up to 5/t_rise for greater accuracy, similar to when you use a fine waveform resolution like 5ps or 10ps.
- *No. of Freq Points* should be 128 points for most via models (this is the default value)

Note: If you include S-Parameter via models in larger S-Parameter circuits, their accuracy must be similar to that of the final circuit.

S-Parameter Settings Example

Edge Rate	Start Freq.	End Freq.	Bandwidth	No. of Freq. Points
100 ps	10 MHz	20GHz	20 GHz	128

Wideband Equivalent Circuit Details

- *Start Frequency* for MGH applications is recommended at 0MHz.
- *End Frequency* should be about 20 GHz.
- Leaving *Approx Order* set to 10 is recommended. You can increase it to 12 if *End Frequency* goes beyond 20GHz for improved accuracy.

  Note: Setting *Approx Order* greater than15 is not recommended.
- There is some loss of accuracy compared to the S Parameter format. However, simulation time is significantly faster.
- Convergence issues are possible if the frequency range is stretched too far.

Narrowband Equivalent Circuit Details

- The narrowband model is derived from the *Target Frequency*.
  - Use a target frequency that is near the middle of the energy content.
  - A good rule of thumb is 1/(1000*risetime). For a driver with `100ps` rise times, a target frequency of `10MHz` is recommended.
  - If the *Target Frequency* is too high, then low frequency (DC losses) are dramatically overestimated.
  - If the *Target Frequency* is too low, then high frequency effects (skin effect and dielectric loss) are underestimated. However, these are small effects in a via.
- This is the least accurate of the via model formats. However, it is very stable and simulates very quickly.

Via Model Types

In addition to single vias, SigXplorer lets you generate and add coupled vias. The three general types are

- Signal-and-signal
- Ground-and-signal
- Power-and-signal

  Note: Power-and-signal vias require an external voltage source.

Coupled via symbols are distinguished from single signal via symbols, as shown below.

Figure 11-7 Single Signal Via Symbol

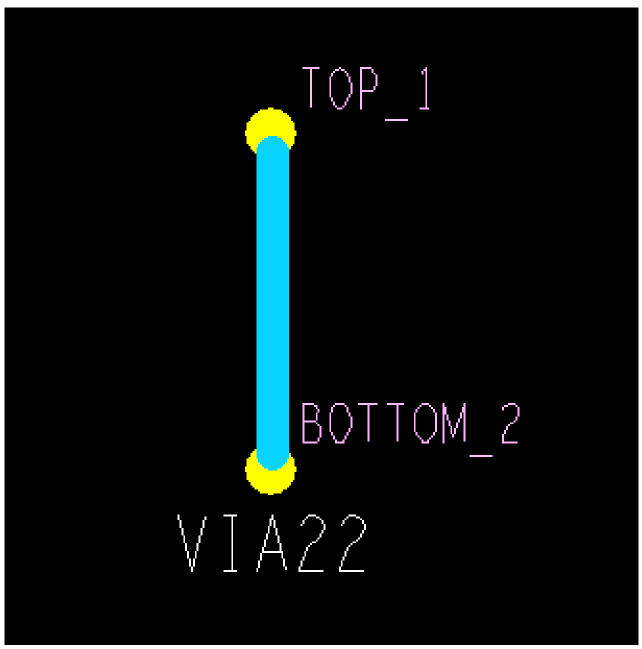
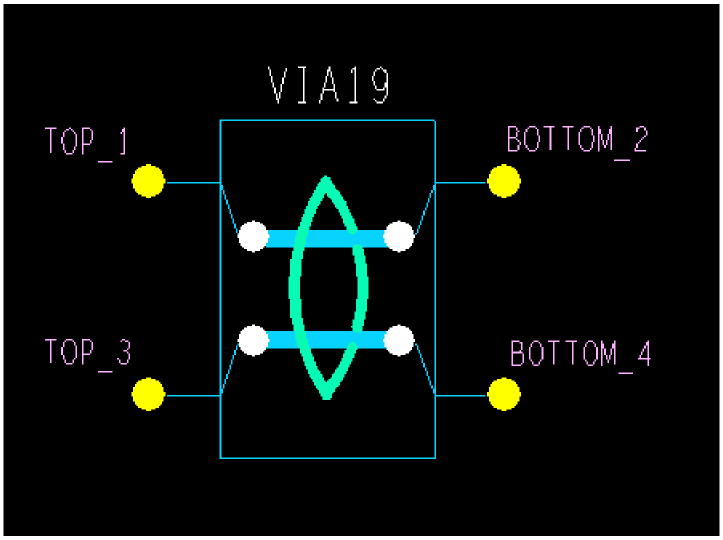
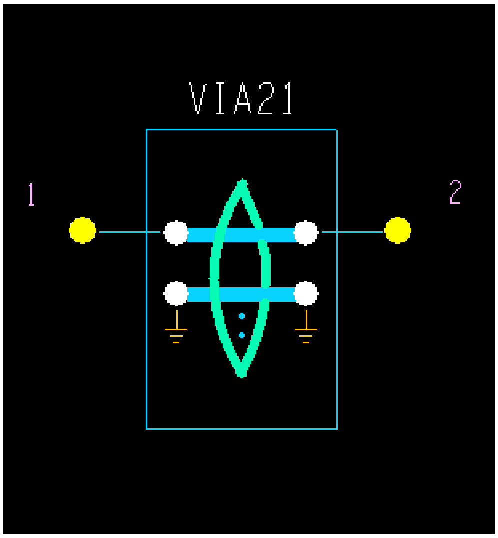


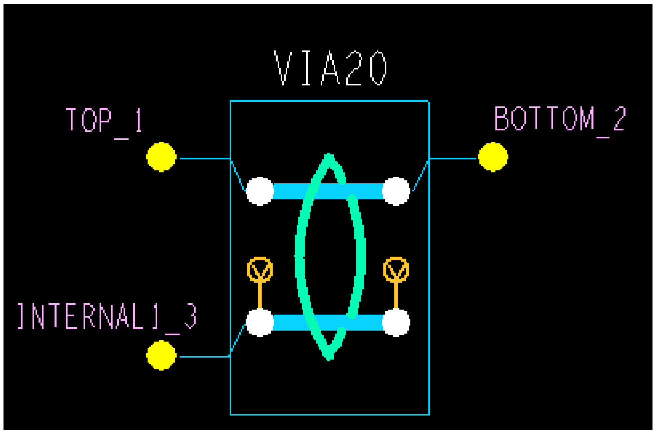
Figure 11-8 Coupled Via Symbols



Signal-and-signal Via



Ground-and-signal



Power-and-signal Via

12

S-Parameters

Topics in this chapter include:

Introduction

Scattering parameters (S-parameters) are mathematical expressions used to define the relationships of traveling waves between ports of a black box. When a signal enters one port, with other terminated ports, S-parameters describe how the traveling waves transmit to and reflect from the various ports of the black box. S-parameters are the reflection and transfer coefficients for the network. S-parameters can characterize the behavior of these structures over a wide frequency range.

S-parameters are used to:

- Analyze frequency characteristics of a complex network.
- Represent a complex network with a single black box.
- Incorporate S-parameter lab measurements in simulations with other circuit elements.

Substituting an S-parameter model for a topology or elements of a topology is not always the best option, because they:

- Are behavioral models, so they lose physical association with the topology.
- Depend on measurement techniques or generation input parameters.
- Are slower to simulate, because they require more processing, depending on the simulator and how much data is in the model.

S-Parameter Generation

Use SigXplorer for S-parameter generation to:

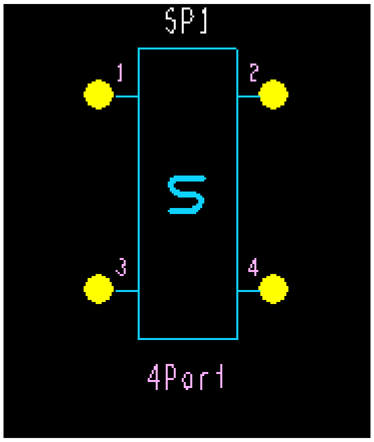
- Evaluate channel loss in the frequency domain to determine if the topology meets the loss budget.

Tip

Experiment with the frequency range and number of frequency points to gauge different responses. For example, using higher End Frequency and more Frequency Points improves accuracy (at the cost of slower performance). See example Viewing Frequency Response Using S-Parameters on page 224

- Create an ESpice black box model of the S-parameter data for use in Time Domain Analysis.

Figure 12-1 S-Parameter Model with 4 Ports (Maximum of 12 Ports Allowed)



Defining Ports

You can define ports for all diodes, IOCells, non-zero voltage sources, and other nodes of interest. Voltage sources without ports become part of the S-parameter black box model. Non-zero voltage sources and IOCells without ports are open circuited during S-parameter generation. If there are no defined ports, an error message appears, and S-parameter generation aborts.

You define ports in the S-Parameter Generation dialog box.

To define ports, do the following:

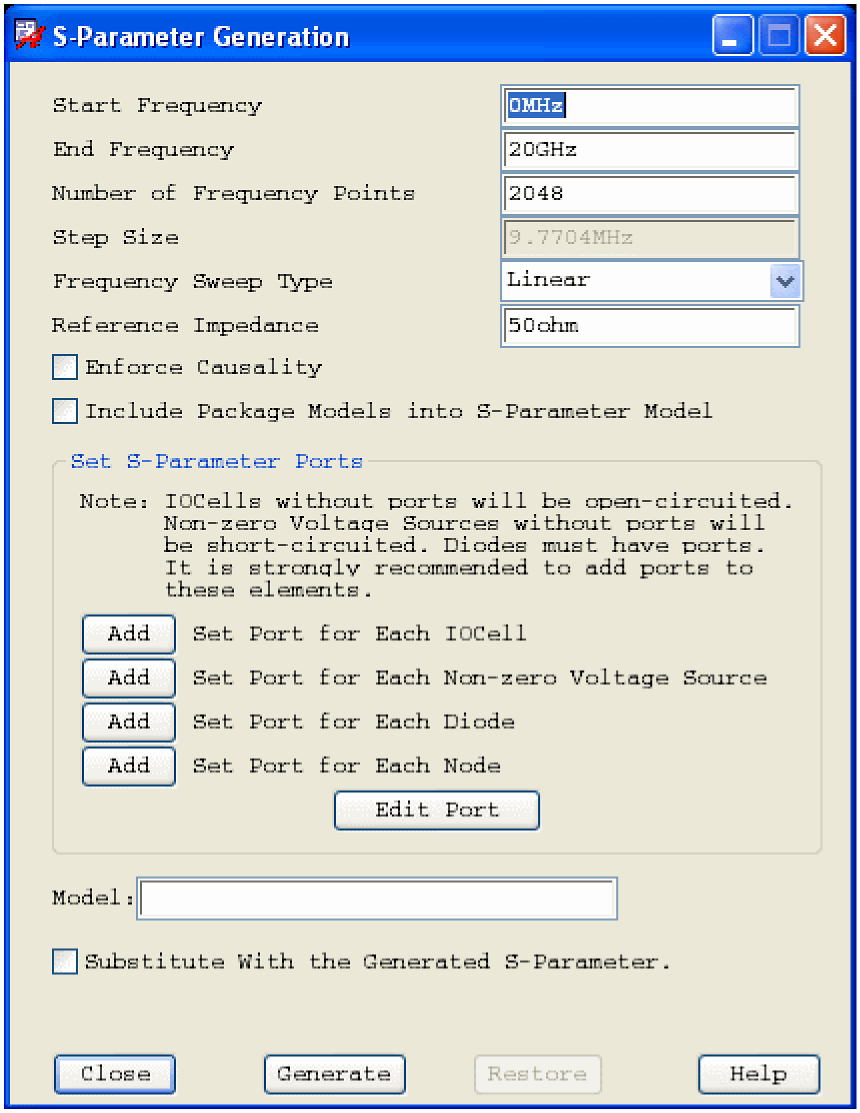
1. Choose *Analyze – [S] Generation...*

   The S-Parameter Generation dialog box is displayed.

2. Click *Add* to automatically generate ports for:

   - IOCells
   - Non-zero voltage source
   - Diodes
   - Nodes

Figure 12-2 S-Parameter Generation



For automatic port setting, set port names with a Refdes_PinNumber which you can change later in the Port Editing dialog box.

For manual port setting, enter the port name for each port. The ports appear on the SigXplorer canvas as you edit them.

Tip

Automatically set ports when you want to look at the loss end frequency of the whole channel. To only focus on a portion of the topology, manually place the ports.

Important

> If you want to place ports at a node in the middle of a topology, someplace other than at IOCells and sources, isolate the item (that you are trying to capture as S-params) from the rest of the circuit to avoid including them in the black box model.

See *S - Parameter Generation Dialog Box* for detailed information on the various options in this dialog box.

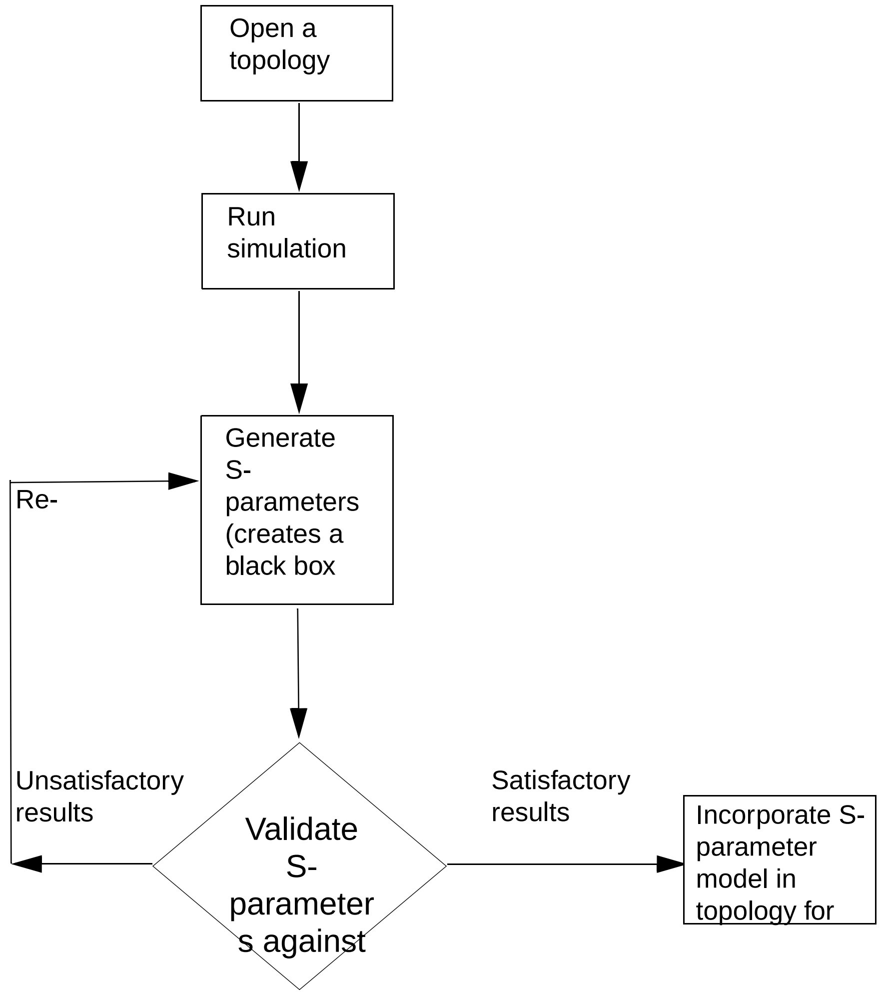
Time Domain Analysis

S-parameter black boxes provide a complete description of the behavior of a given network as seen at its ports, enabling time domain analysis (TDA) of a circuit. The circuit simulator uses the black boxes without knowing the topology of the network.

You can use S-parameters for TDA to:

- Incorporate measured data for known or fixed elements in the signal path for simulations. For example, a legacy backplane.
- Provide a black box model for a complex section of topology for use in simulation by another user. For example, a package vendor provides the black box model to a system engineer.

Figure 12-3 S-Parameter Validation Flow for Time Domain Analysis



Tip

When using S-parameters for time domain analysis:

- Follow the usage recommendations for Start Frequency, End Frequency, and number of points, as described following this tip.
- Use Linear sweep; not Logarithmic.
- Ensure that via models support the requested S-parameter bandwidth, if they exist in the topology.

Tip

The recommended Start Frequency is `0MHz`. The end frequency should be about 2/t_rise, such as you use for fine waveform resolutions of `5` or `10 ps`. The number of frequency points should be a power of 2, with a frequency step of about `10MHz`.

S Parameter Settings Example

Edge Rate	Start Freq.	End Freq.	Bandwidth	Freq. Step	No. of Freq. Points
100 ps	0 MHz	20GHz	20 GHz	10 MHz	2048

After generating the DML ESpice model for the S-parameter data, use the S-parameter black box in the same way that you use an ESpice black box. The S-parameter black box use model is as follows:

- Generate DML models for the S-parameter data, as shown in Generating an S-Parameter Black Box on page 226
- Load the DML library that contains the S-parameter models, if it is not present.
- Add the S-parameter black box part to the canvas. Use *Add Part* and then select an ESpice device that contains S-parameter data.
- Connect any element supported in SigXplorer to the S-parameter black box (including non-linear IBIS I/O buffers, transmission lines, lumped elements) and simulate.
- Edit the simulation preferences, if necessary, and validate the simulation results against the source topology.

The S-parameter black box symbol automatically appears when you select an S-parameter DML model from the *Add Part* menu. The S tag that appears in the middle of the symbol represents the S-parameter black box, as shown in Figure 12-1 on page 217.

The ESpice device model name and the outer `.subckt` name for the black box must be the same. The maximum number of ports of the S-parameter data is twelve. The number of terminals of the generated symbol is equal to the model's subckt terminal count (which is equal to the number of ports). The black box terminal names are the same as the outer `.subckt` terminal names in the DML model and appear on the symbol from left-to-right, top-to-bottom.

Tip

To control the order of the terminals on the SigXplorer canvas, edit the outer `.subckt` terminal names in the DML model, and then map the outer `.subckt` terminals to the corresponding inner `.subckt` terminals when you instantiate the inner `.subckt`.

Typical Use Models

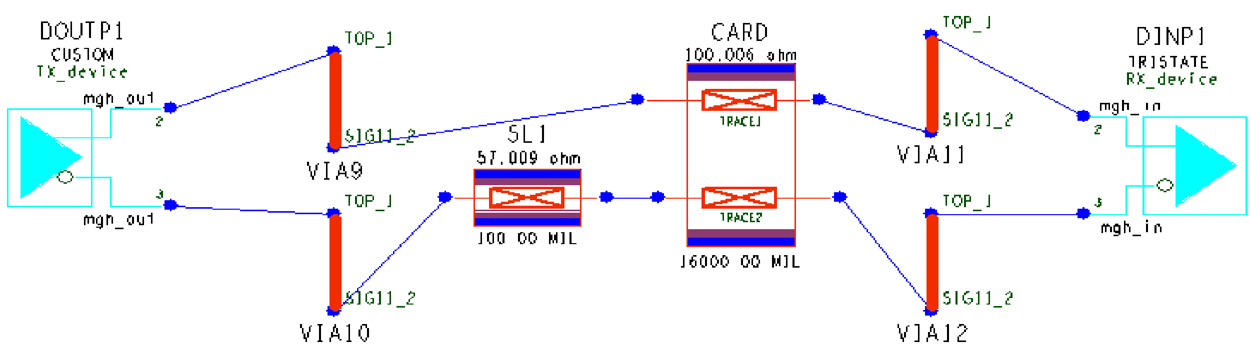
Viewing Frequency Response Using S-Parameters

One of the primary benefits of using S-parameters is the ability to plot them and examine the loss exhibited by the interconnect. This example is a typical use model.

1. In SigXplorer open a topology similar to Figure 12-4.

   The backplane and receiving card appear in detail.

Figure 12-4 Source Topology



2. Choose *Analyze – Preferences*.
3. In the *Simulation Parameters* tabbed page, set the *Waveform Resolution* to `10 ps`.
4. Set the *Default Cutoff Frequency* to `10GHz`.

Important

Change the Default Cutoff Frequency to avoid modeling a lossless case.

5. Choose *Analyze – [S] Generation*...

   The S-Parameter Generation dialog box appears.

6. Click *Add* next to the *Set Port for each IOCell* option.

   The ports automatically appear in the topology with names.

7. Specify the following values:

Start Frequency	`1 Hz`
End Frequency	`10 GHz`
Frequency Points	`1024`
Model	`16inch`

8. Click *Generate*.

   The S-Parameter Generation log appears.

9. Examine the Port Index to determine which port numbers to look at to see the transmission.

10. In SigWave, turn off *Re*, *Im*, and *Ph*. Turn on *Ma* (magnitude).

11. Click the *push pin icon* to keep these settings.

12. Turn off all sub-items in SigWave and turn on the transmission plot per the port index.

13. Put a vertical marker at `4GHz` and zoom in at the crossing point.

    The loss is greater than the loss budget of `10dB`. The simple solution is to reduce the trace length.

14. In SigXplorer, close the S-Parameter Generation dialog box.

15. Change the length of the coupled trace from `16inches` to `10inches (`*Parameters tab)*.

16. Click *Analyze – [S] Generation...*.

    The S-Parameter Generation dialog box appears.

17. Enter `10inch` in the Model field.

18. Click *Generate* to re-generate the S-parameters.

19. In SigWave, overlay the two waveforms and compare.

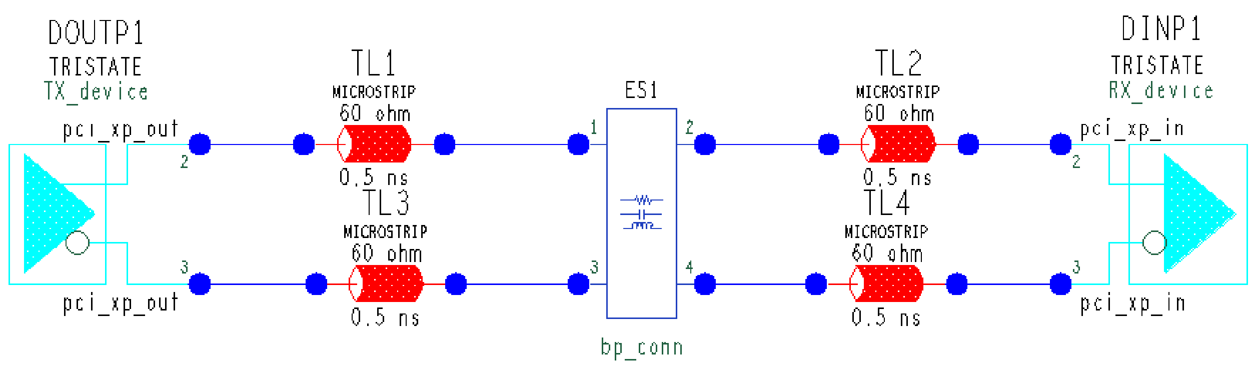
    The trace length now meets the loss budget.

Generating an S-Parameter Black Box

You generate an ESpice Device model black box to replace a topology, or some elements of the topology, for use in what-if situations.

1. In SigXplorer, open an existing topology, as seen in Figure 12-5.

Figure 12-5 Topology without Generated S-Parameters



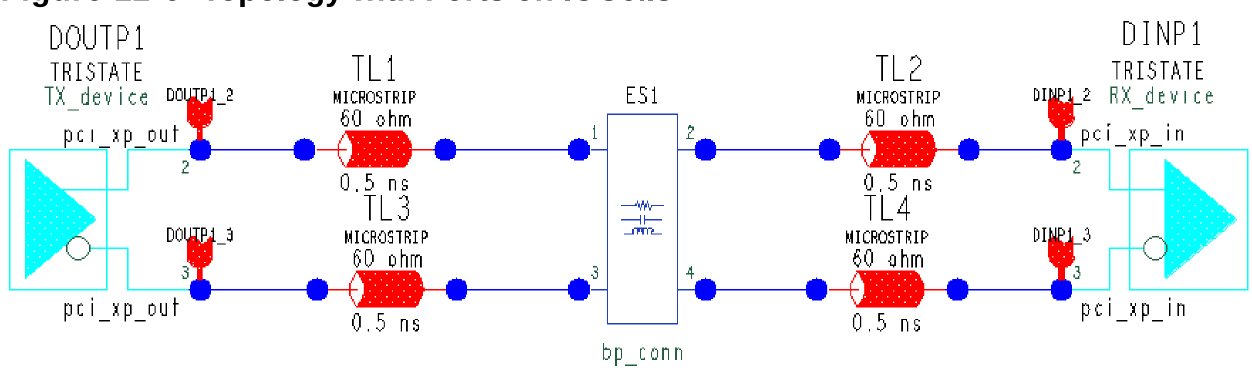
2. Choose *Analyze – [S] Generation...*

   The S-Parameter Generation dialog box appears.

3. Click *Add* next to the *Set Port for each IOCell* option.

   The added ports appear in the canvas, as seen in Figure 12-6.

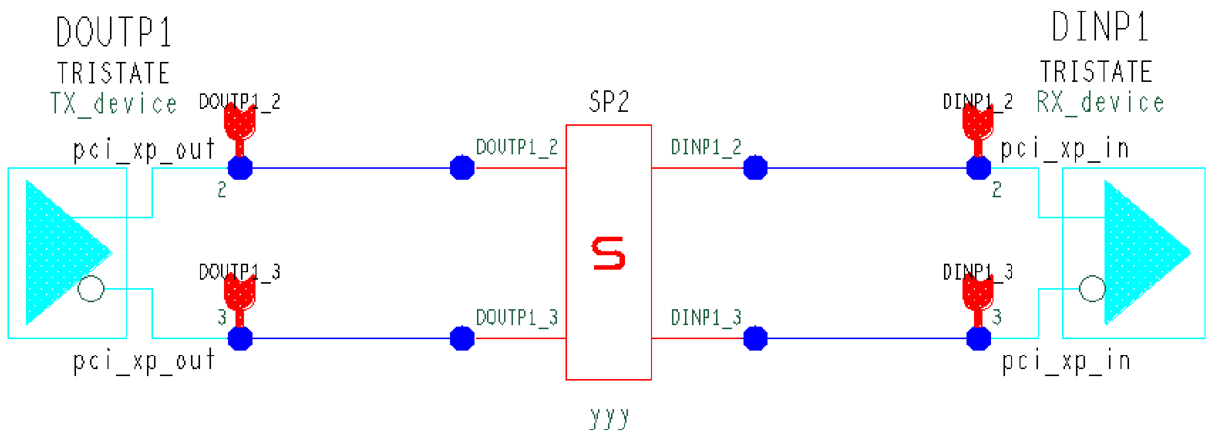
Figure 12-6 Topology with Ports on IOCells



4. Select the *Substitute with the Generated S-Parameter* option.
5. Click *Generate*.

   The original topology updates with the generated S-parameter black box, as seen in Figure 12-7.

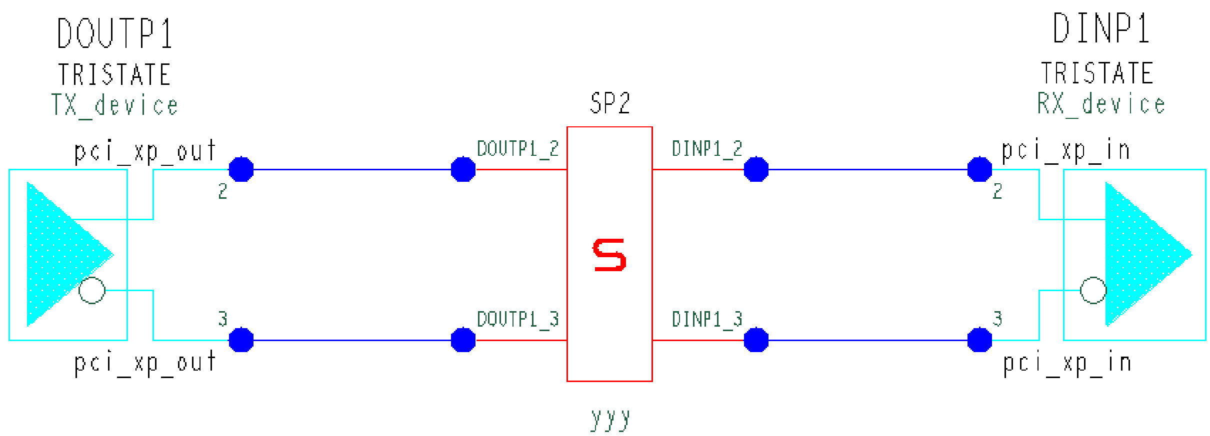
Figure 12-7 Generated S-Parameter Black Box



6. Click *Close*.

   The dialog box closes and all ports disappear from the topology. See Figure 12-8. The new S-parameter model appears in the working DML library as a Touchstone file.

Figure 12-8 Updated Topology



13

Custom Measurements

Topics in this chapter include:

Introduction

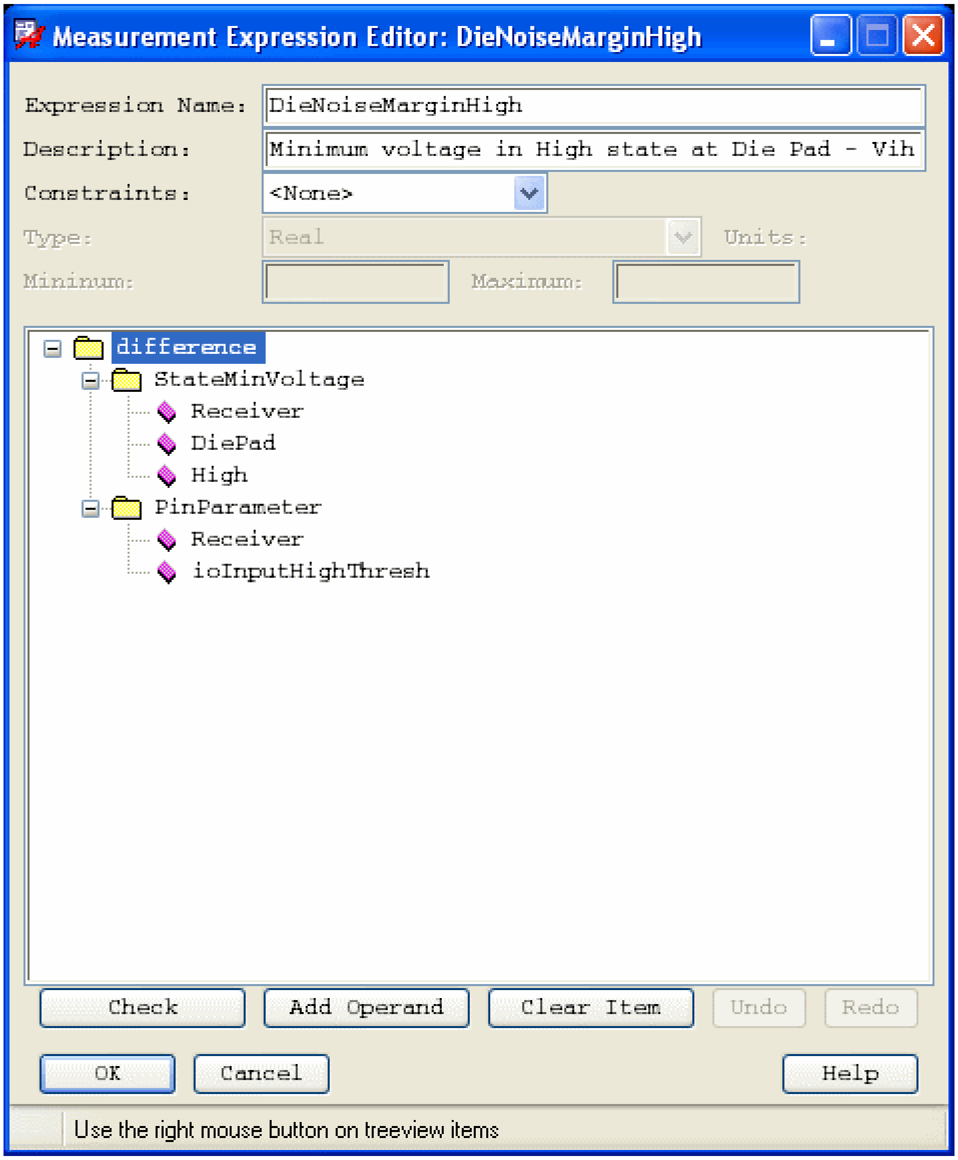
By defining and evaluating custom measurements, you can obtain new data from signal integrity simulations. Customized expressions provide the simulation data that fits your requirements.

You use the Measurement Expression Editor to build syntactically correct measurement expressions. In a custom measurement expression, you use parameter substitution, math functions, predefined measurements, and references to other custom measurement expressions you create. You evaluate the measurement expressions during simulation.

Note: You can only have one Measurement Expression Editor dialog box open at a time.

Measurement Expressions

Figure 13-1 Measurement Expression Editor



Use this dialog box to create and modify customized measurement expressions for SigXplorer. When you open the Measurement Expression Editor to edit an existing expression, the name of the expression appears in both the Expression Name field and in the title bar. For more information about the editor, see the *SigXplorer Command Reference*.

Number	Enter literal numbers in custom measurement expressions in decimal or scientific notation. They can have units or Spice scaling suffixes (f, p, n, u, m, k, meg, g, t), but they cannot include white space. The units.`dat` file defines usable unit suffixes.
Reference to another measurement expression	Reference standard and user-defined measurements by their measurement expression names. When you select from the tree view MEASUREMENT_TBD, or any other placeholder that you replace with a measurement, the pull-down menu includes all standard measurements available in the Reflection, Crosstalk, and EMI sections of the Measurements tab, as well as all currently available custom measurement expressions. User-defined custom measurements appear alphabetically, following the standard measurements. **Note:** To avoid recursive expressions, the measurement you are currently editing does not appear on the list.
Pin parameter reference	Obtain numeric data from the library definitions of pins and IOCells available in measurement expressions. You access the data using the PinParameter function, which has the following arguments: ■ PIN_TBD ■ PARAMETER_TBD **Note:** The values for some parameters are sensitive to the current Fast/Typical/Slow settings for the pin.
Waveform measurement function call	Specify the waveform function by pin and node name. Pin names are: ■ Receiver - name of the measured receiver pin ■ Driver - name of the active driver pin ■ Strobe - name of the strobe pin associated with the receiver ■ "comp.pin" - exact pin name in dot notation (use double quotes)

Math function call	Specify the following math functions to perform calculations. All arguments are numeric. ■ Min Function - return is the lesser of one or more values. ■ Max Function - return is the greater of one or more values. ■ Sum Function - return is the sum of two or more values. ■ Difference Function - return is the difference between two or more values. ■ Product Function - return is multiples of two or more values. ■ Quotient Function - return is the division of two or more values. ■ Abs Function - return is the absolute value of the input argument.

Note: In functions, all arguments evaluate as either a single number or as NA. Run mode ignores NA returns from functions and appears as NA in the simulation results data.

Function calls introduce hierarchy by calling nested arguments.

The following table lists the standard measurements that can be referenced by name in measurement expressions. These same standard measurements appear on the Measurements tab of the spreadsheet.

Measurement	Type	Description
Crosstalk	Voltage	Maximum voltage excursion on crosstalk victim net. (Crosstalk measurement)
NoiseMargin	Voltage	Minimum of NoiseMarginHigh or NoiseMarginLow (Reflection measurement)
NoiseMarginHigh	Voltage	Minimum noise margin (voltage) in the high state (Reflection measurement)
NoiseMarginLow	Voltage	Vilmax minus the maximum noise margin (voltage) in the low state (Reflection measurement)
SettleDelay	Time	Maximum of SettleDelayRise and SettleDelayFall (Reflection measurement)

SettleDelayRise	Time	Final time to settle high above Vihmin minus BufferDelayRise (Reflection measurement)
SettleDelayFall	Time	Final time to settle low below Vilmax minus BufferDelayFall (Reflection measurement)
SwitchDelay	Time	Minimum of SwitchDelayRise and SwitchDelayFall (Reflection measurement)
SwitchDelayFall	Time	First time to switch low below Vihmin minus BufferDelayFall (Reflection measurement)
SwitchDelayRise	Time	First time to switch high above Vilmax minus BufferDelayRise (Reflection measurement)
OvershootHigh	Voltage	Maximum voltage seen in High state (Reflection measurement)
OvershootLow	Voltage	Minimum voltage seen in the Low state (Reflection measurement)
PropDelay	Time	Calculated transmission line propagation delay (Reflection measurement)
PulseFreq	Frequency	Frequency of the excitation pulse (EMI measurement)
Monotonic	0 or 1	Monotonic switching check of Rising and Falling edges (Reflection measurement)
MonotonicRise	0 or 1	Monotonic switching check of Rising edge (Reflection measurement)
MonotonicFall	0 or 1	Monotonic switching check of Falling edge (Reflection measurement)
FirstIncidentRise	0 or 1	First Incident Switching check of Rising edge (Reflection measurement)
FirstIncidentFall	0 or 1	First Incident Switching check of Falling edge (Reflection measurement)
BufferDelayRise	Time	Buffer Delay for Rising edge (Reflection measurement)
BufferDelayFall	Time	Buffer Delay for Falling edge (Reflection measurement)
VoltageSwing	Voltage	Peak to Peak voltage of the excitation (EMI measurement)

RiseTime	Time	Minimum of the rise and fall times of the excitation (EMI measurement)
PeakEmission	DBuV/m	Peak Radiated Electric Field dBuV/m (EMI measurement)
PeakFrequency	Frequency	Frequency at which PeakEmission occurs (EMI measurement)
EMIStatus	0 or 1	PASS/FAIL check of EMI regulation compliance (EMI measurement)

Exporting and Importing Custom Measurements

You save custom measurements as part of the topology where you created them. In order to use the custom measurement expressions with another topology, you must export the custom measurement expressions for the first topology to a text file. You can then import the text file to another topology, where you can edit the individual expressions.

Custom Measurement Editor Message Reference

When you check a custom measurement expression during an editing session, the following error and warning messages appear:

ERROR: 'VALUE_TBD' must be replaced with a valid value.

> **Probable Cause:** There are _TBD placeholders in the expression.
>
> **Suggested Solution:** Replace the indicated placeholder with a valid argument.

ERROR: 'VoltageAtTime' argument 3 must be a number, measurement name, parameter name, or function call

> **Probable Cause:** You entered an invalid numeric function argument. You probably imported a corrupted custom measurement expression file.
>
> **Suggested Solution:** Replace the invalid argument with a number, or with a legal measurement name, parameter name or function call.

ERROR: 'FooBar' is not a recognized parameter or measurement name

> **Probable Cause:** The referenced parameter or measurement, FooBar, does not exist. You probably imported a corrupted custom measurement expression file.

Suggested Solution: The check operation verifies that you only use valid and available parameters and measurements in expressions. Verify that the parameter or measurement is not gone, and that the name is spelled correctly.

ERROR: Unable to convert '3.25smoots' to a number

Probable Cause: The argument 3.25smoots appears to be a number, but it is not.

Suggested Solution: Anything that starts off looking like a number is parsed as a number.

ERROR: 'factorial' is not a recognized function name

Probable Cause: The first word following a left parenthesis must be a function name.

Suggested Solution: Enter a valid function name or remove the parenthesis character.

ERROR: First argument to 'CrossingTime' function must be a pin designator

Probable Cause: This error can occur when an expression that is specific to one topology (for example, it uses explicit pin names) is used in another topology.

Suggested Solution: This check is made for all waveform function calls. Modify the expression so that its arguments reflect the topology you are using it with.

ERROR: Third argument to 'CrossingTime' function must be a number

Probable Cause: The expression contains a parameter or function call in a position where only a number is valid.

Suggested Solution: Replace the parameter or function call with a number.

WARNING: No pins with 'input' trace used in 'CrossingTime' function

Suggested Solution: Special nodes defined in MacroModels will apply only to pins that use that MacroModel. The evaluator returns nil for other pins. This warning tells when a node is not found.