cadence™

Allegro®
SI SigXplorer User Guide

Product Version 16.2
November 2008

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Contents

1

Introduction

Topics in this chapter include

Finding Information about SigXplorer

The SigXplorer information set consists of online books accessible from CDSDoc in both HTML and PDF formats. You can 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*.
Allegro PCB SI User Guide	This book contains reference information about the analysis engine used by SigXplorer.
Allegro Constraint Manager Tutorial	This book contains lessons in using SigXplorer and Constraint Manager.

Introducing SigXplorer

You use SigXplorer to create, modify, simulate, and save prototypes of net topologies. You explore these topologies by modifying circuit parameters, simulating, and examining reports and waveforms, and repeating this process to modify circuit parameters for optimum results.

SigXplorer graphically displays simulation results as spreadsheet data and waveforms in SigWave. SigXplorer uses familiar user interface controls. For complete information on the menus, icons, and views of SigXplorer, see the *Allegro SI SigXplorer Command Reference*.

Starting SigXplorer

You can start SigXplorer with a clean canvas or by extracting a topology from the Allegro PCB Editor.

Invoking SigXplorer with a Clean Canvas

➤ In Windows, choose *Start – Run – Sigxp*.

-or-

➤ In UNIX, type *sigxp* in a Shell window.

The SigXplorer window opens with a clean canvas.

-or-

➤ From Allegro SI, choose *Tools – Topology Editor*.

The SigXplorer window opens with a clean canvas.

Invoking SigXplorer by Extracting a Topology

1. In Constraint Manager, select one of the Net Worksheets from the worksheet.
2. Select the net of the topology you want to extract, and then right-click to display the context-sensitive menu.
3. Select the SigXplorer option.

   SigXplorer launches and the topology appears in the Topology Canvas.

   -or-

4. In Allegro SI, choose *Analyze – SI/EMI Sim – Probe*.

The Signal Analysis dialog box appears.

5. Select the net of the topology you want to extract.
6. Click *View Topology.*

   SigXplorer launches and the topology appears in the Topology Canvas.

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

Using the SigXplorer User Interface

SigXplorer lets you 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 the *Allegro SI SigXplorer Command Reference*.

Using commands accessed through the Menus and the Toolbar, you set up and perform simulations as shown in Table 1-1.

Table 1-1 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.

Manipulating the Canvas View

Panning

You can pan a topology (the action of moving across a topology in the canvas) to view different parts of it. To pan a topology, hold the cursor inside the canvas, 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. You cannot drag the cursor outside the boundaries of the topology, if *Canvas* mode is active in the *Edit* menu.

Moving

You can move a topology on the canvas by dragging the cursor to surround the topology. When you have surrounded it within a box, you can move the topology up, down or sideways on the canvas.

Re-sizing the spreadsheet

You can move the border between the spreadsheet and the canvas to increase or decrease the view. Grab the edge of the spreadsheet border with the cursor, move the spreadsheet up or down, according to where you want it.

2

Working with Topologies

Topics in this chapter include:

Introduction

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 an Allegro PCB Editor or Allegro SI database.

Creating a Topology

You create a topology by:

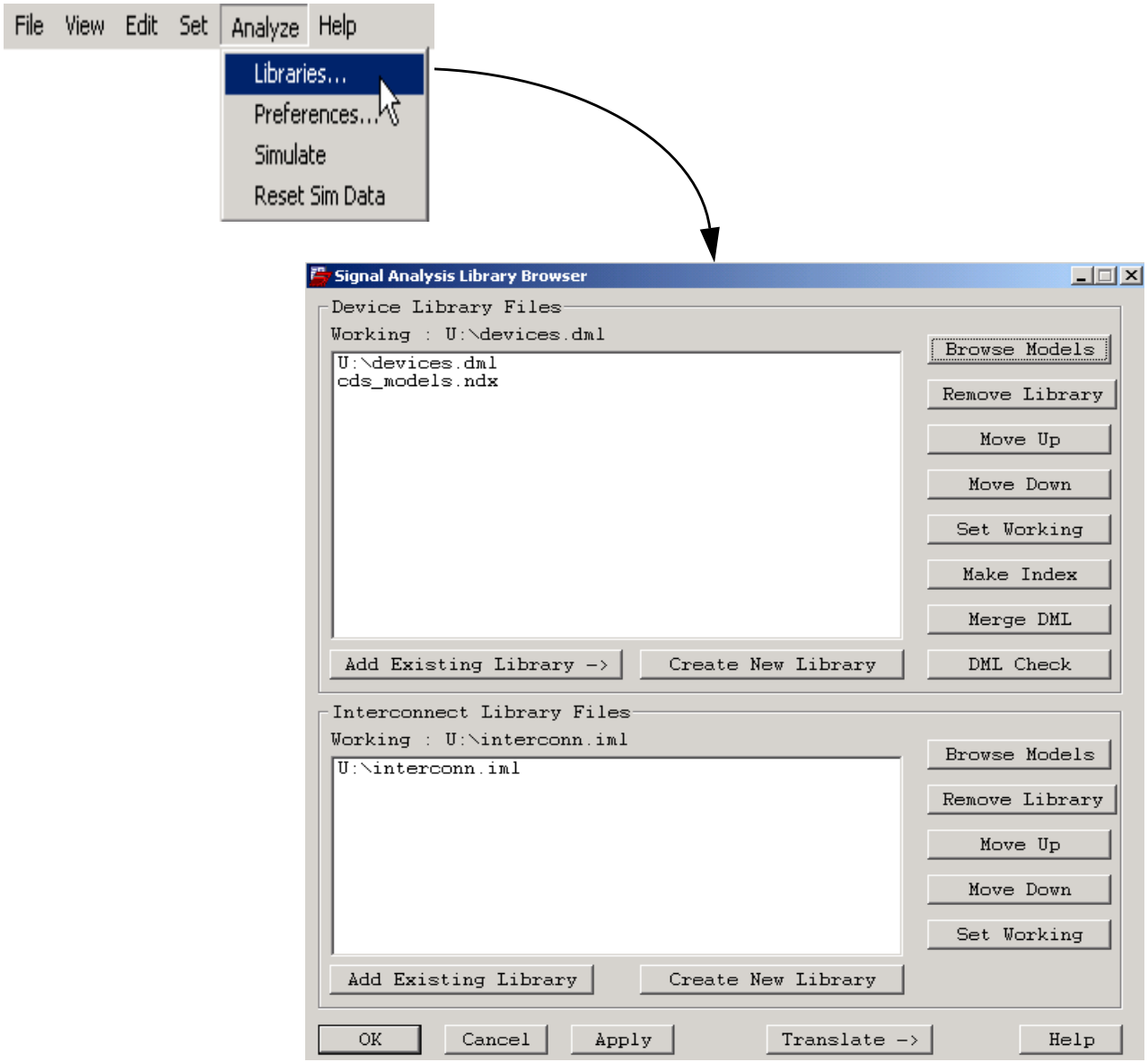
1. Selecting the individual elements from the Cadence default libraries or from user-defined libraries.
2. Placing each element in the SigXplorer canvas.
3. Wiring the elements, as required.

Using Sigxplorer, you capture the net schedule, 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 which applies to every net, eliminating the need to create a topology for each net of the bus.

Adding Parts

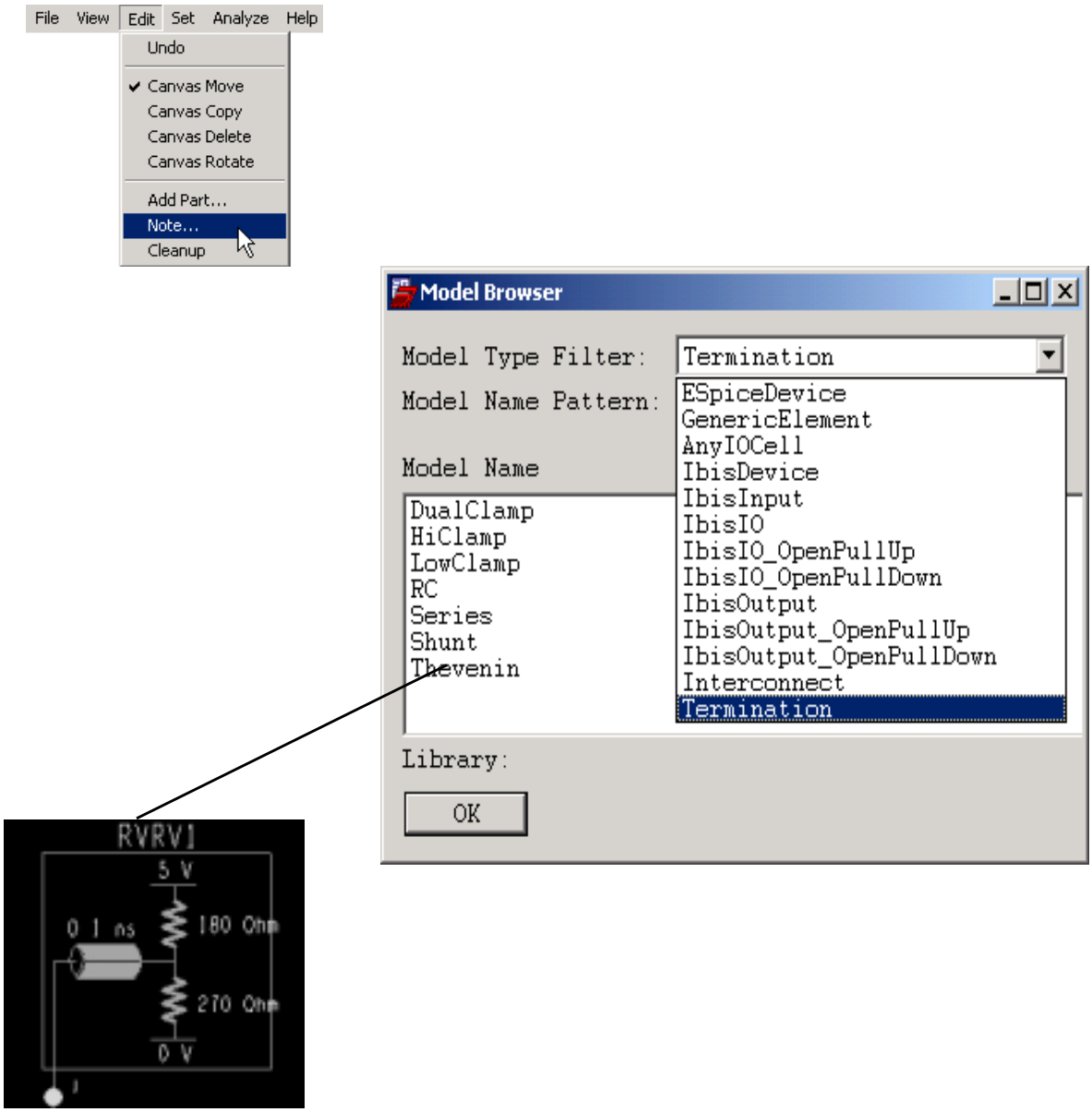
You add parts (discretes, IO buffers, packages) to the topology using the *Parts Browser* in Allegro SI. Before you can add parts, you must make them available by specifying the signal integrity model libraries (*.dml* files) that you want to use through the use of the *Signal Analysis Library Browser* (Figure 2-1). The signal integrity model libraries contain electrical models for components that subsequently appear in the *Model Browser* (Figure 2-2).

Figure 2-1 Signal Analysis Library Browser



You add a part to the topology canvas by selecting it in the *Model Browser,* Figure 2-2 (the part symbol attaches to the end of your cursor) and dragging it to the canvas. This puts you in *add part mode* where the part is instantiated with each click that you make in the canvas.

Figure 2-2 Model Browser



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

Signal Analysis Models and Model Libraries

SigXplorer maintains the part models in the signal analysis device and interconnect model libraries used by Allegro SI and Allegro PCB Editor.

Use the Signal Analysis Library Browser (*Analyze – Librairies)* to load and examine Model libraries. You can see which model libraries are open and accessible. SigXplorer can access any of the libraries listed in the Library Browser to obtain part models.

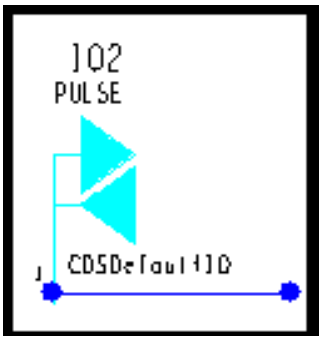
The libraries listed in the two working fields are the only two libraries where SigXplorer saves any new or edited models. Use *Add Existing Library* to load additional libraries.

Use the Signal Analysis Model Browser to edit device and interconnect part models. Use *Browse Models* in the Signal Analysis Library Browser to open the Model Browser. Adjust the filter fields to control which models you see. Use *Edit* to examine a selected model. Depending on the model you select, *Edit* opens the model data in a text editor or in a GUI specific to the model type. You can examine the model, make changes to the model data, and save the modified model in the working library.

Modifying Parameters for Topology Elements

SigXplorer maintains the data for the circuit and the part parameters in the spreadsheet. When you place a part in the topology, you see both the element symbol for the part and associated text fields that describe default parameters and other information associated with the topology element, as seen in Figure 2-3. After simulating, you edit this data to further refine the circuit design.

Figure 2-3 A Placed Part in the Canvas



Use the *Set Parameter* dialog box to modify numeric parameters for part models. See the *SigXplorer Command Reference* for more information.

You can modify the following information:

- Name (Reference Designator or RefDes)

  If you click TL1 to select the RefDes for a transmission line, Name T1 opens in the spreadsheet to display all parameters associated with the part and the value for each. The RefDes parameter, T1, is highlighted and ready for editing. After entering a new RefDes in the spreadsheet, the RefDes changes in the canvas, as well.

- Parameter value

  Click the part value in the canvas to highlight the associated spreadsheet row. Click the icon in the spreadsheet value cell to display the Set Parameter dialog box. Enter the new parameter value and click *OK*. After you complete the edit, the data changes in both the spreadsheet and the canvas.

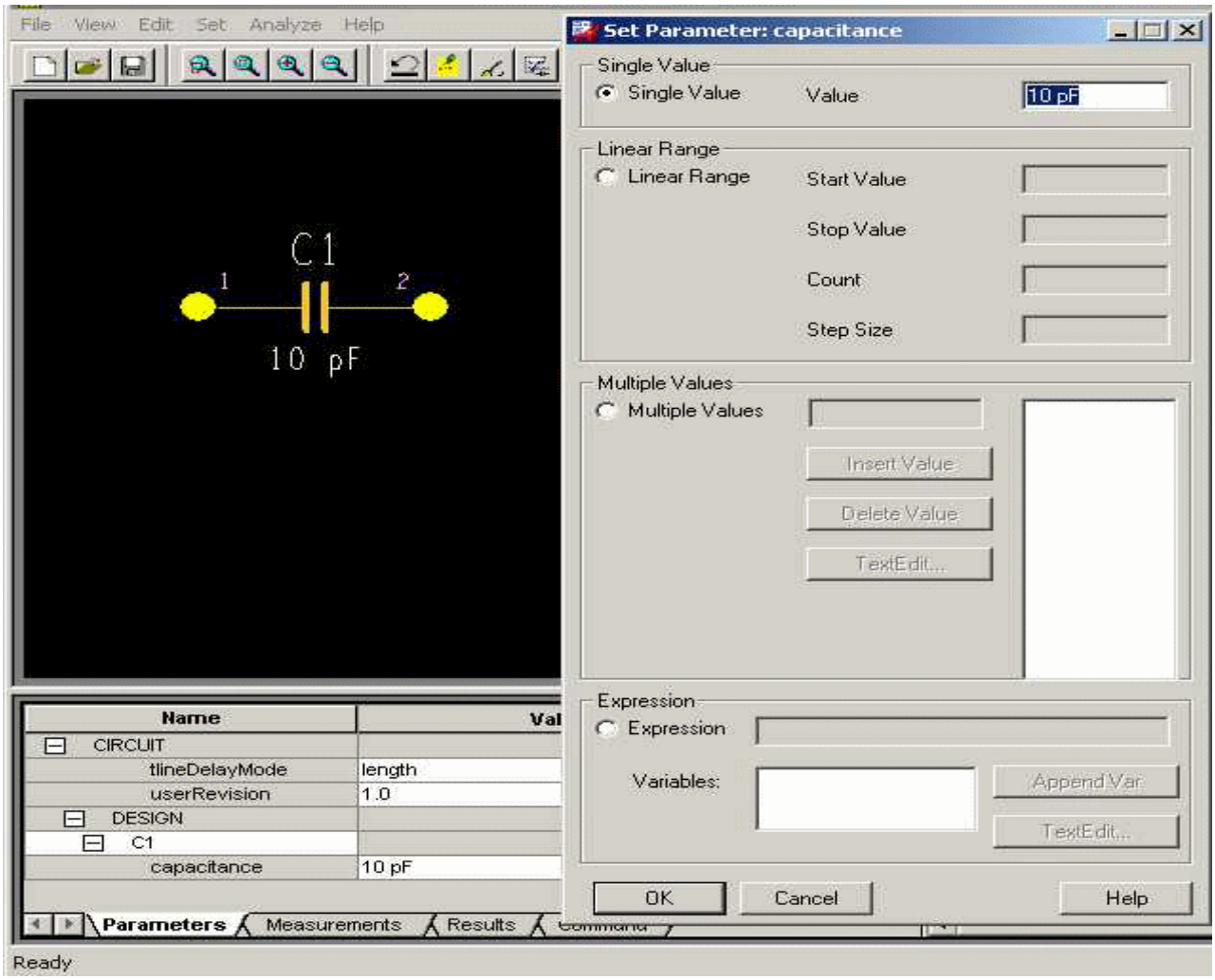
- Stimulus associated with an IOCell, or buffer

  The spreadsheet does not contain the stimulus data for IOCell parts. You use the IOCell Stimulus Editor to modify this information. See the *SigXplorer Command Reference* for more information.

When you select a label of a symbol in the canvas, the *Parameters* tab of the spreadsheet opens to the data for editing, as shown in Figure 2-4.

You click on the part's reference designator to edit the RefDes in the Parameters tab.

Figure 2-4 The Parameters Tab with Displayed Data



Default Values for Parameters

When you select a part from the Model Browser and add it to the canvas, SigXplorer assigns a default set of parameter values, .

You can modify these defaults for component models, interconnect models, and terminator models, using the Default Values dialog box (*Set – Defaults*). For more information, see the *SigXplorer Command Reference*.

Note: When you modify a default value, it becomes the value of that part whenever you place the symbol.

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 part and the default value associated with each parameter.
- Modify the default parameter values.

Understanding Transmission Line Models in SigXplorer

Transmission line (TL or Tline) elements in SigXplorer are characterized by *impedance*, *propDelay*, *traceGeometry*, and *velocity*, values of which you can view in the *Parameters* tab of the SigXplorer spreadsheet. Of these four, the “fundamental” parameters are *impedance* and *velocity*; *propDelay* as an indication of trace length is defined in nanoseconds as opposed to mils, inches, or other physical unit of measurement. It is somewhat equivalent, therefore, to *velocity*, whose value default is 5600 mils per nanosecond. *traceGeometry* does not affect simulations, but does allow you to set default velocity values to any microstrip or stripline TLs 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 (lossless) transmission line.

Advantages of using a lossless TL in pre-layout simulations are its 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 TLs for microstrip or stripline models.

Wiring the Topology

To wire a topology, click the terminal of a topology element (see Figure 2-5). As you move the cursor, a wire moves with the cursor away from the terminal. Click the destination terminal on another part. A wire is drawn between the parts. Click another terminal to extend the wire. When you click a terminal, the wire starts or extends, regardless of the current edit mode.

To delete a wire from the topology, click on the wire anywhere except at its terminals. The wire disappears, regardless of the current edit mode.

Figure 2-5 Wiring a Topology

Select symbols from the model browser and place on canvas in a semi-orderly arrangement

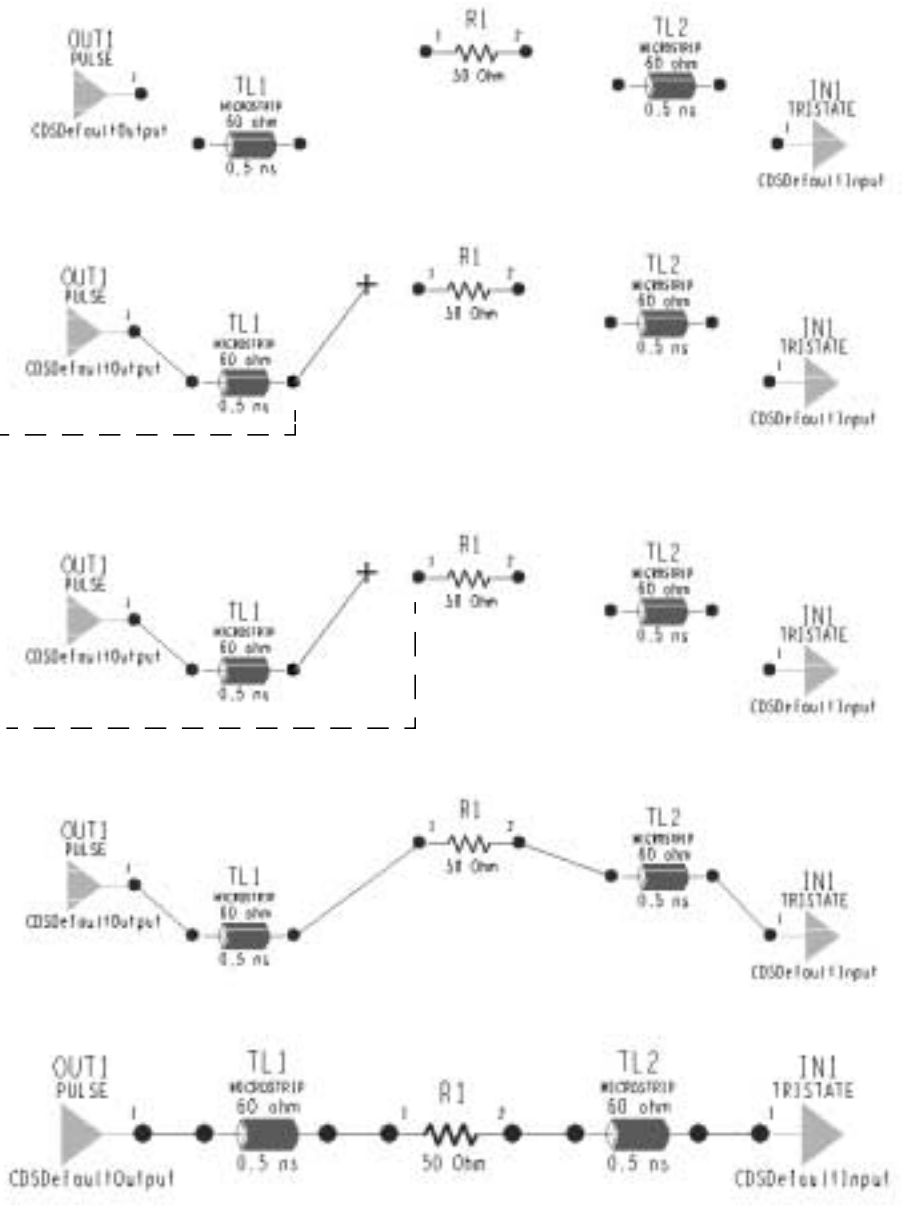
Click the source pin on the symbol to begin the connection

Drag the wire from the source pin of one symbol to the target pin of the next symbol

Double-click the target pin, or click on the pin followed by an off-pin click on the canvas to end the connection

Finish wiring the topology

Organize the topology (cleanup command) to assert a left-to-right, top-to-bottom appearance

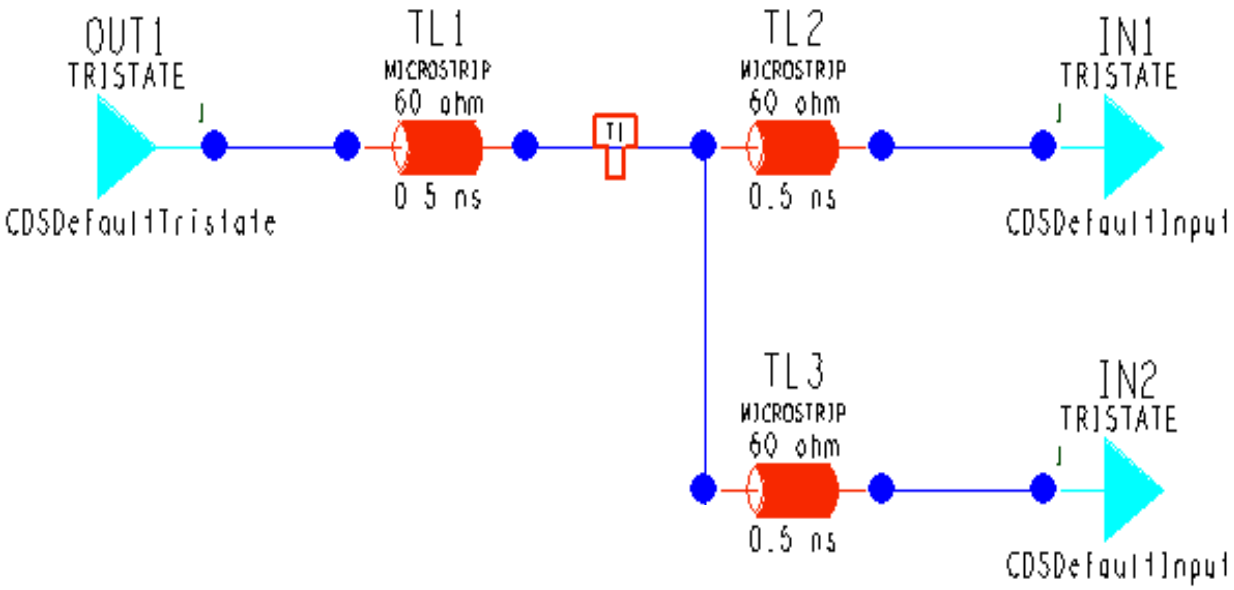


T-point Elements

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 Figure2-6.

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. 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, 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.

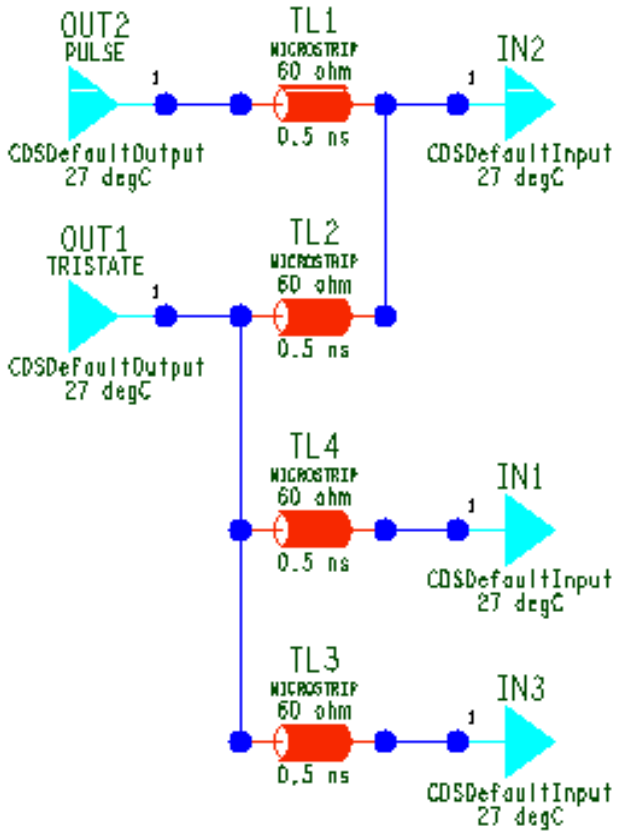
The following are the schedule types available in SigXplorer:

- Min Spanning Tree
- Daisy Chain
- Source Load Daisy Chain
- Star
- Far End Cluster

Min Spanning Tree Schedule

A min tree schedule consists of connecting all of the 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 have been scheduled. Following is an example of a Min Tree schedule.

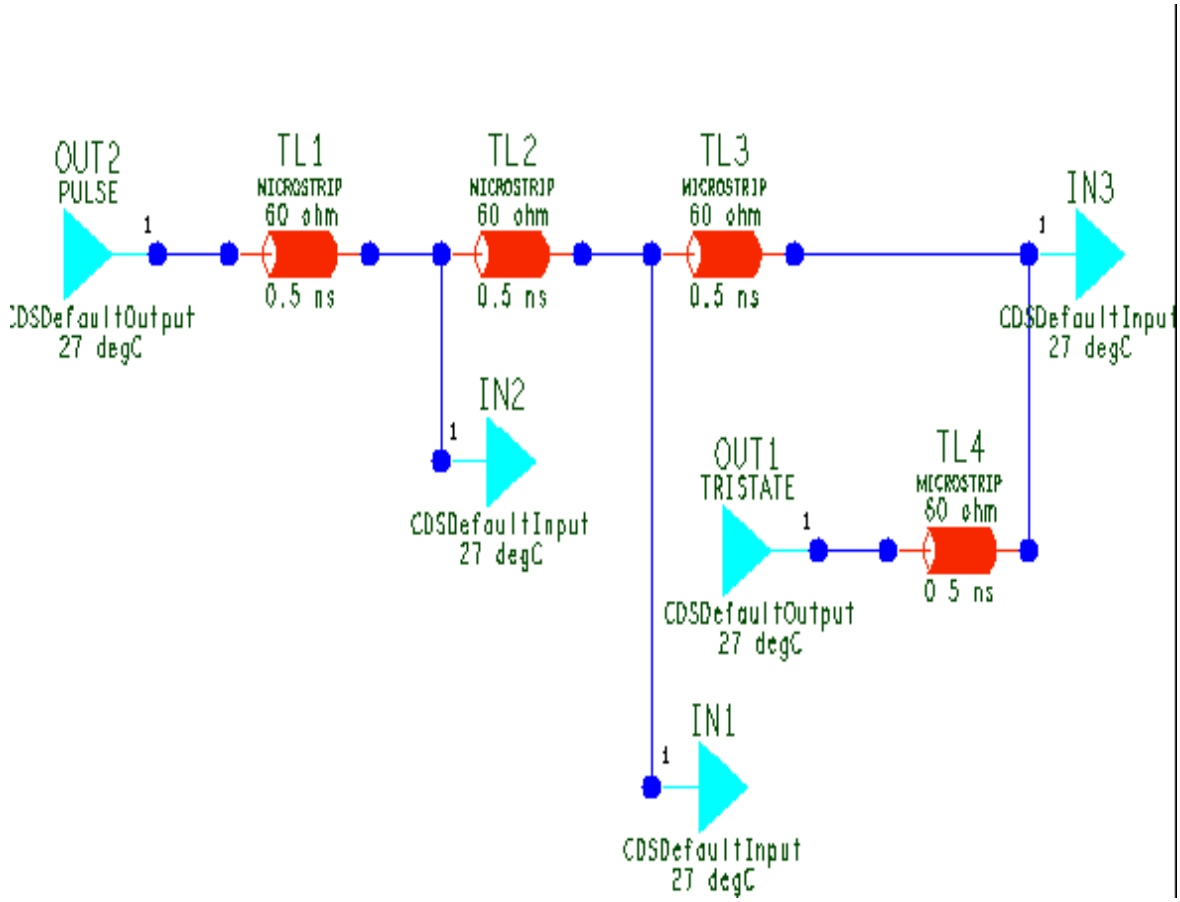
Figure 2-7 Sample Min Tree Schedule



Daisy Chain Schedule

A daisy chain schedule consists of connecting the pins of the topology with minimum connection lengths, but 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 connects with a TLine. The closest pin to the last pin scheduled connects with a TLine. Continue until all of the pins connect. Following is an example of a daisy chain schedule.

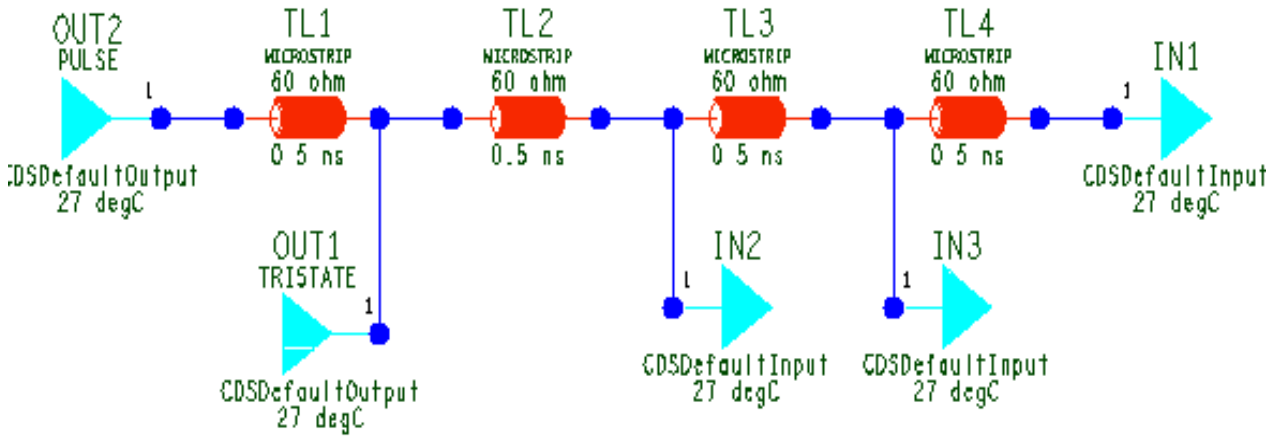
Figure 2-8 Sample Daisy Chain Schedule



Source Load Daisy Chain Schedule

A source load daisy chain schedule is similar to a daisy chain schedule except that all driver pins connect first, followed by all receiver pins. Following is an example of a source load daisy chain schedule.

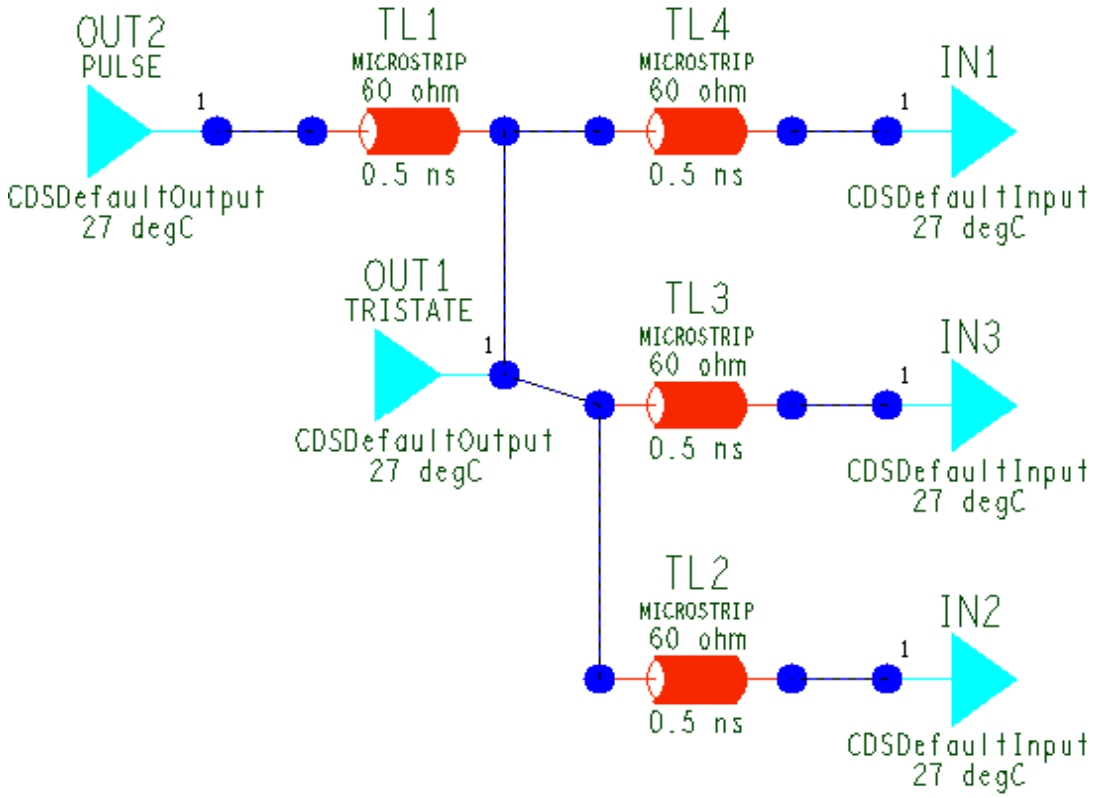
Figure 2-9 Sample Source Load Daisy Chain Schedule



Star Scheduling

In a star schedule, the driver pins are daisy-chained together, and then all of the receiver pins connect to the last driver pin. Figure 2-10 on page 29 is an example of a star schedule.

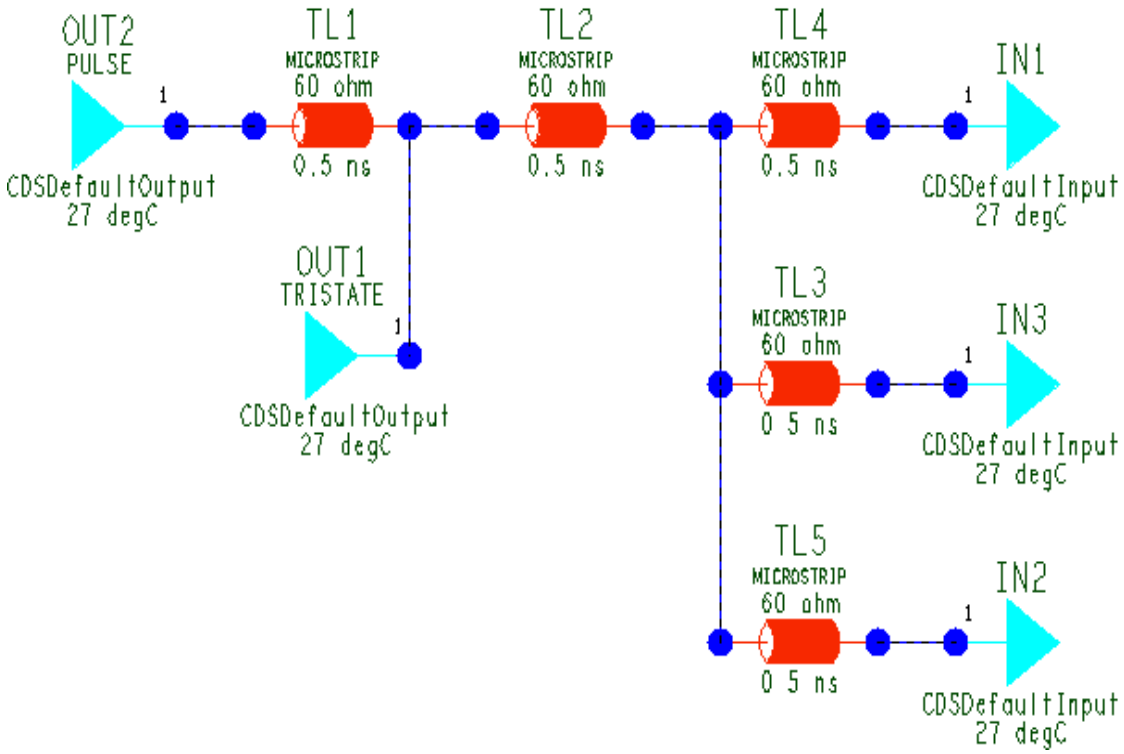
Figure 2-10 Sample Star Schedule



Far End Cluster Scheduling

The far end cluster schedule is similar to a star schedule except that the last driver pin connects to a T-point to which all of the receivers connect. Figure 2-11 on page 30 is an example of a far end cluster schedule.

Figure 2-11 Sample Far End Cluster Schedule

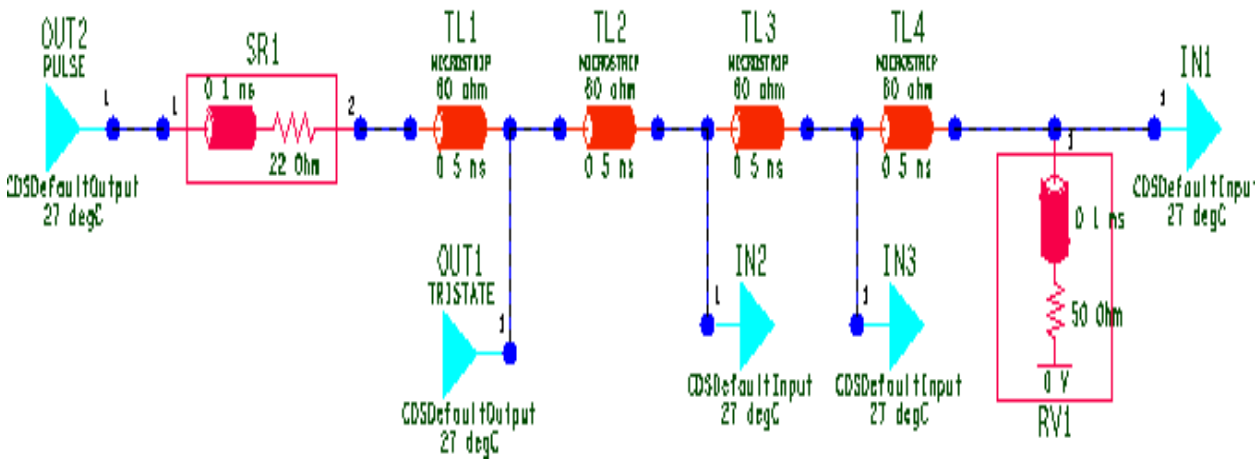


Rules for Generic Topology Schedules

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 parts is when the topology defines a diff pair. In this case both sides (for example, the inverting and non-inverting signals) must contain the same numbers and types of pins.
- SigXplorer deletes any unmatched pins, after a message appears asking for permission. Otherwise, SigXplorer schedules all other types of parts.
- All existing TLines disappear and new TLines added.
- The proximity of the pins to each other determines the scheduling.
- Active bi-directional pins become drivers or receivers, if they are not active.
- Each of the schedules starts sequencing the pins with the primary driver. If the topology only contains one driver IOCell, then it is the primary driver. If there are multiple drivers, SigXplorer selects the active one. If there are multiple active drivers, then SigXplorer chooses one of the active drivers. If there are no active drivers, then SigXplorer chooses one of the non-active drivers. If there are no drivers, but there is a non-active bidirectional pin, then SigXplorer chooses the bidirectional pin.
- If a topology contains no drivers or bi-directional pins, then a warning appears, the schedule type resets to template and no changes occur.
- After you assign a schedule to a topology, use the cleanup command to redraw the topology on the canvas (*Edit – Cleanup*).
- Any terminator, correctly connected to a pin, remains during scheduling. This is shown in Figure 2-12 on page 32.

Figure 2-12 Terminator on a Source Load Daisy Chain Schedule



Extracting a Topology

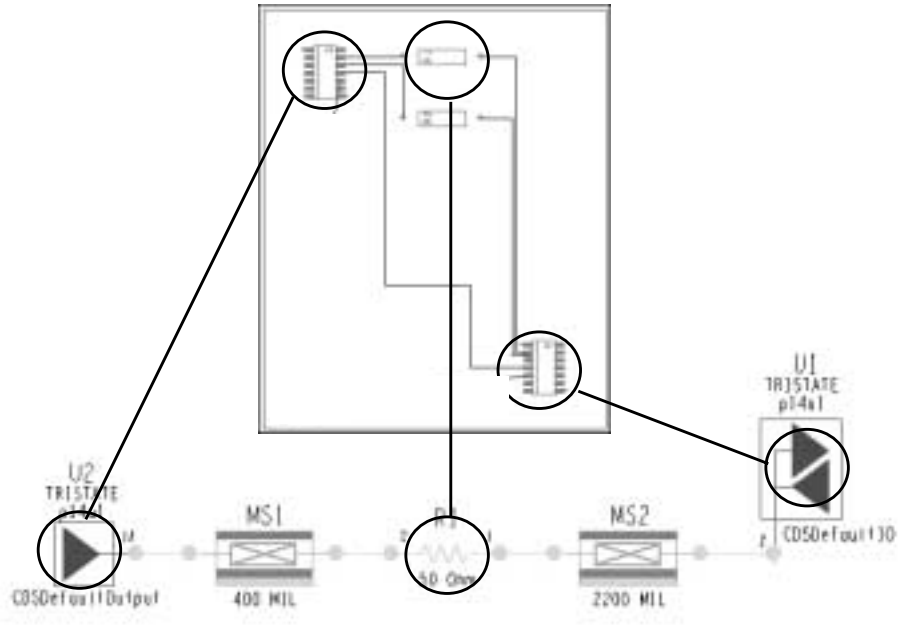
You extract a net topology from the PCB Editor into SigXplorer to see if it meets signal integrity requirements. If not, you can modify the topology until it does meet the requirements.

Using SigXplorer, you develop electrical and physical constraints based on design requirements. These constraints control the final placement and routing of the printed circuit board.

The Database Set Up Advisor (*Tools – Setup Advisor* in the PCB Editor) is a utility used to bridge the physical design representation in the PCB Editor with the equivalent electrical design representation in SigXplorer by guiding you through the steps necessary to ensure a clean net extraction from the PCB Editor database.

Physical view in PCB Editor

Electrical view in SI 210



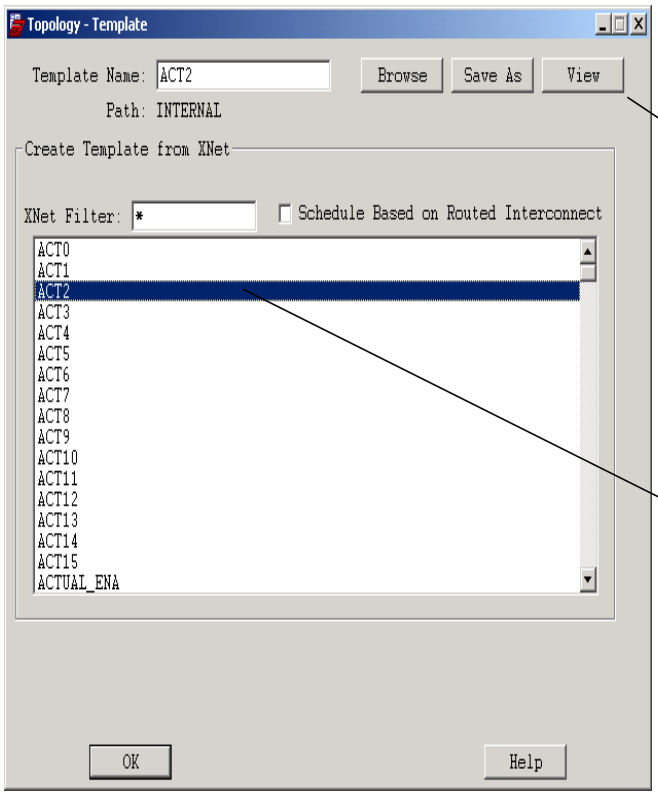
The following sections take you through the database setup requirements. The advisor is organized by the following modules:

In this module...	You will...
Cross-section	Edit the type and characteristics of the varied material layers in the layout.
DC Nets	Identify which nets in the layout are to be connected to a constant DC voltage value that you specify.

In this module...	You will...
Devices	Provide information about the devices in the layout, such as class (IC, IO, Discrete) and pin usage (BI, IN, OUT).
SI Models	Assign electrical models to components in the layout.
SI Audit	Audit specific nets in the layout to verify that they are set up properly for extraction and simulation.

➤ Once you successfully complete the Audit phase of the Database Setup Advisor, choose *Tools – Topology Extract*.

The *Topology Template* dialog box appears.



Click View to launch SI 210 with the selected net

Select net from list or click on the net in the PCB design

Select a net and click *View* to extract the net topology from the PCB Editor database and explore it within SigXplorer. This net is now clean (extractable) and does not require you to repeat the setup process for subsequent extraction with the *Topology Extract* command in the PCB Editor.

3

Preparing for Simulations

Topics in this chapter include:

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 simulate.

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 informaton on simulation and analysis, see the *Allegro PCB SI User Guide*.

Setting Analysis Preferences

Use *Analyze – Preferences* from the SigXplorer menu to set the simulation preferences to modify the simulation results.

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
Simulation Parameters	Determine simulation parameters, including: Fixed Duration Waveform Resolution Cutoff Frequency Buffer Delays Sweep Cases Algorithm Model Generation Simulators and Field Solvers

Simulation Modes	Select simulation modes to perform a single simulation or simulation sweeping, including: Fast, Typical, Slow, Fast/Slow, Slow/Fast Active_Driver, All_Drivers
S-Parameter Modes	Determine: Transient Simulation Method DC Extrapolation Method Impulse Response Causality
Measurement Modes	Select Measurement Modes to specify: Measure Delays At Receiver Selection Custom Simulation 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 your setup, from the external pin node, or from the internal die node, if present. You can set these choices in the `signoise` batch command or by way of the Analysis Preferences dialog box (*Analyze – Preferences*).

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

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

- 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

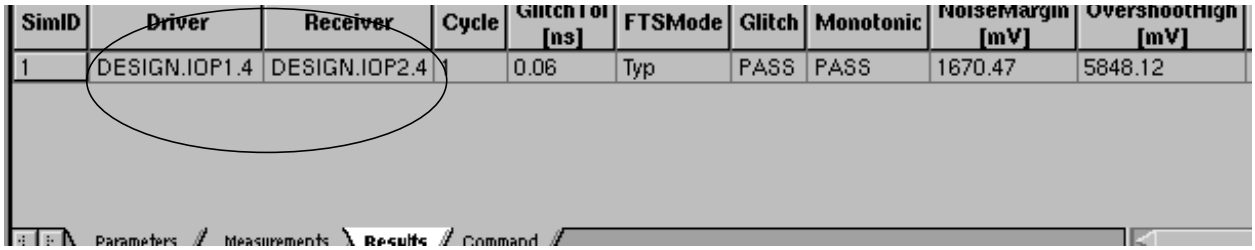
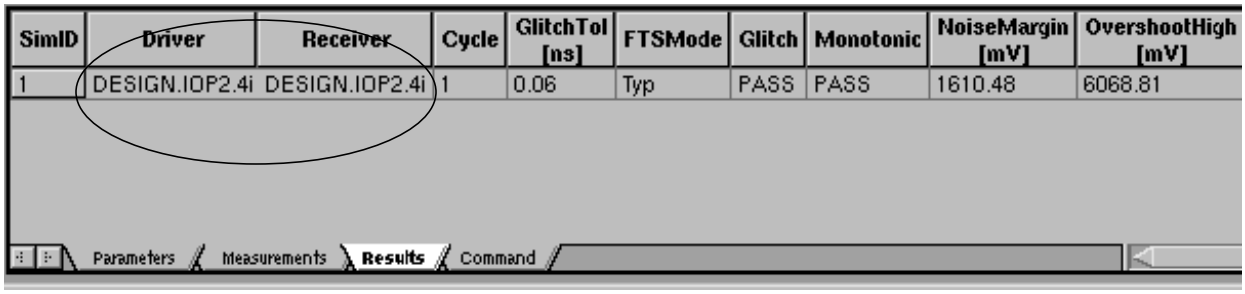


Figure 3-2 Die Measurement Selection Report



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 choose a stimulus type for the driver. You can specify only one IOCell at a time to be the driver. All 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 selection 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. 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 60).

Performing Parametric Sweeps

Simulation sweeping relies on combinations of the following criteria:

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

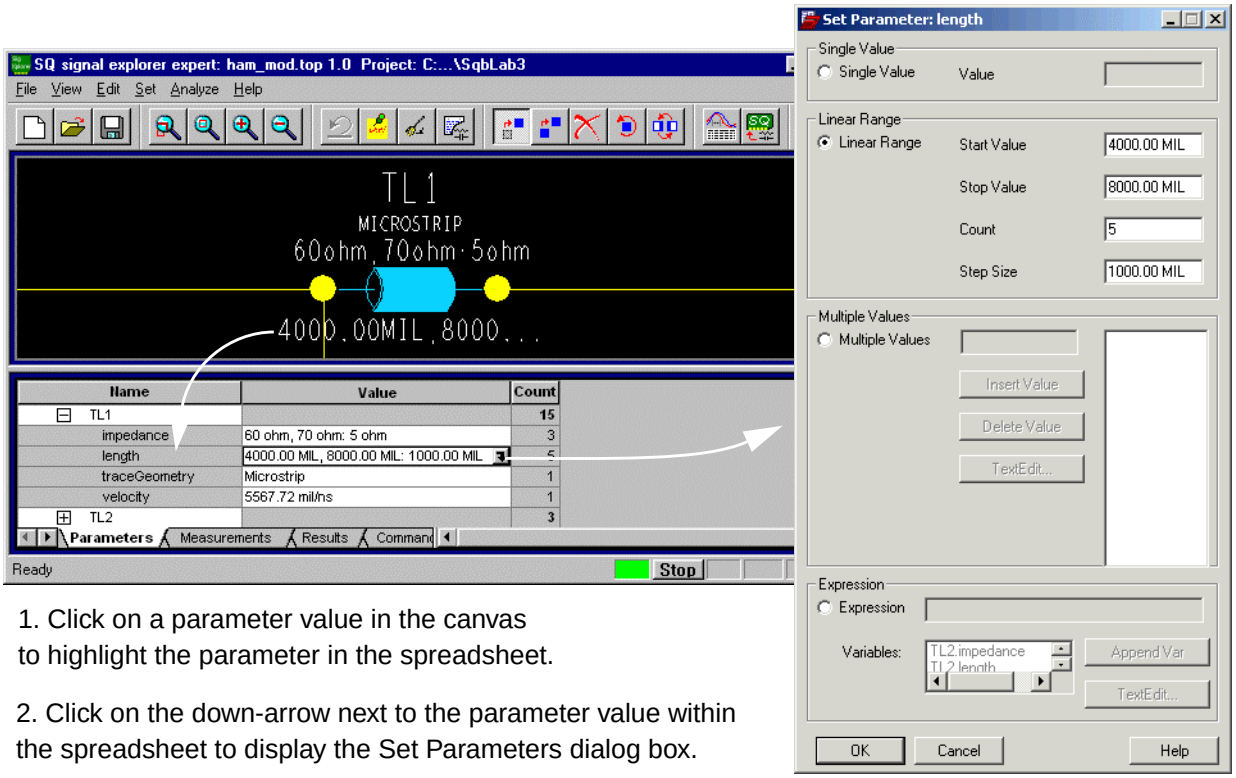
Sweeping by part parameter values involves covering a set or range of values (sweep count points) that you specify for eligible sweep parameters through a set of simulations. SigXplorer calculates the total number of simulations based on the number of sweep count points required for each sweep parameter.

You sweep by driver slew rate by selecting a set of FTS Mode target rates from the Simulation Mode section of the Simulate 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



1. Click on a parameter value in the canvas to highlight the parameter in the spreadsheet.

2. Click on the down-arrow next to the parameter value within the spreadsheet to display the Set Parameters dialog box.

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.

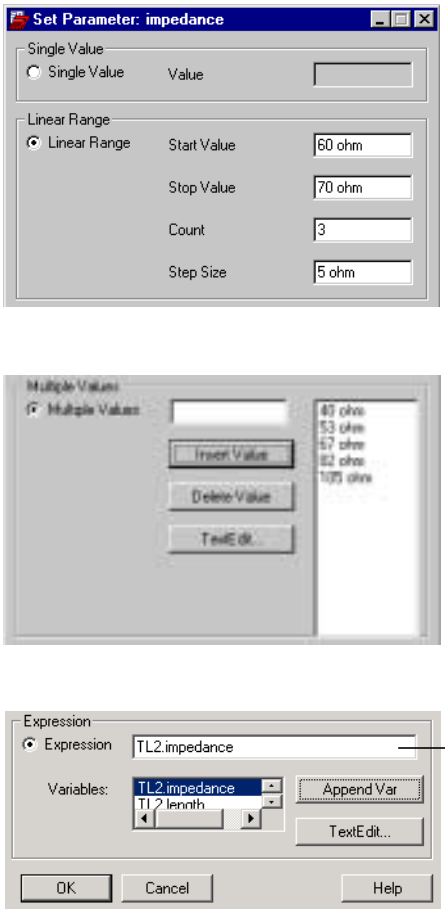
When you use an expression to define a parameter attribute that references a second parameter attribute defined as a range or list, the first parameter tracks the second parameter as it changes during simulation sweeping.

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

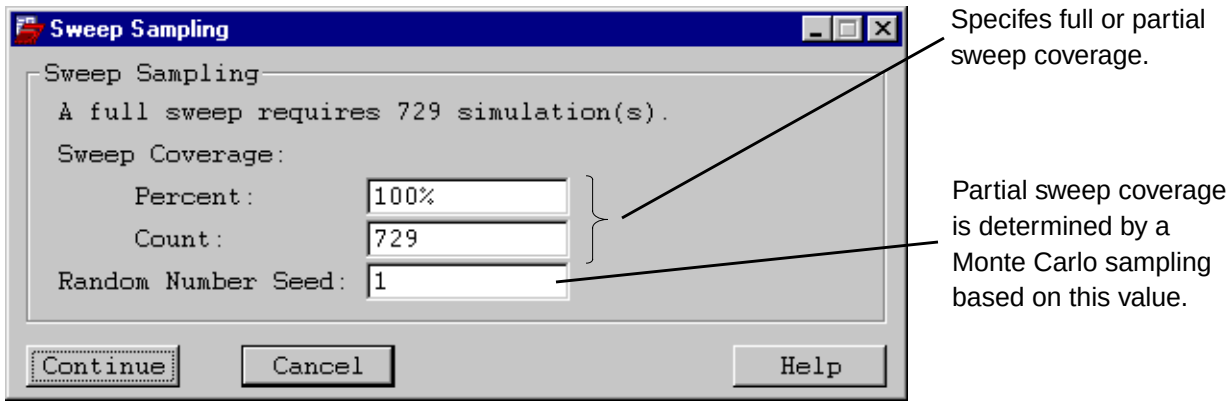
SigXplorer lets you control sweep sampling. After you have set up to perform simulation sweeps, you choose *Analyze — Simulate*. The Sweep Sampling dialog box appears before an active sweep begins.

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.

Figure 3-5 The Sweep Sampling Dialog Box



Sweep Results

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.

The sweep report contains information on topology, swept elements, driver and load names, impedance and delay variables. You choose *File — Export Spreadsheet* to save simulation sweep results in a tab-delimited text file. The contents of this file are suitable for import into an external spreadsheet program.

Waveforms

The parametric sweep function does not produce waveforms directly. However, when viewing the sweep results, you can click to select a row in the spreadsheet which re-runs that single simulation and opens SigWave to display the resulting waveforms.

Figure 3-6 The Sweep Report

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 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

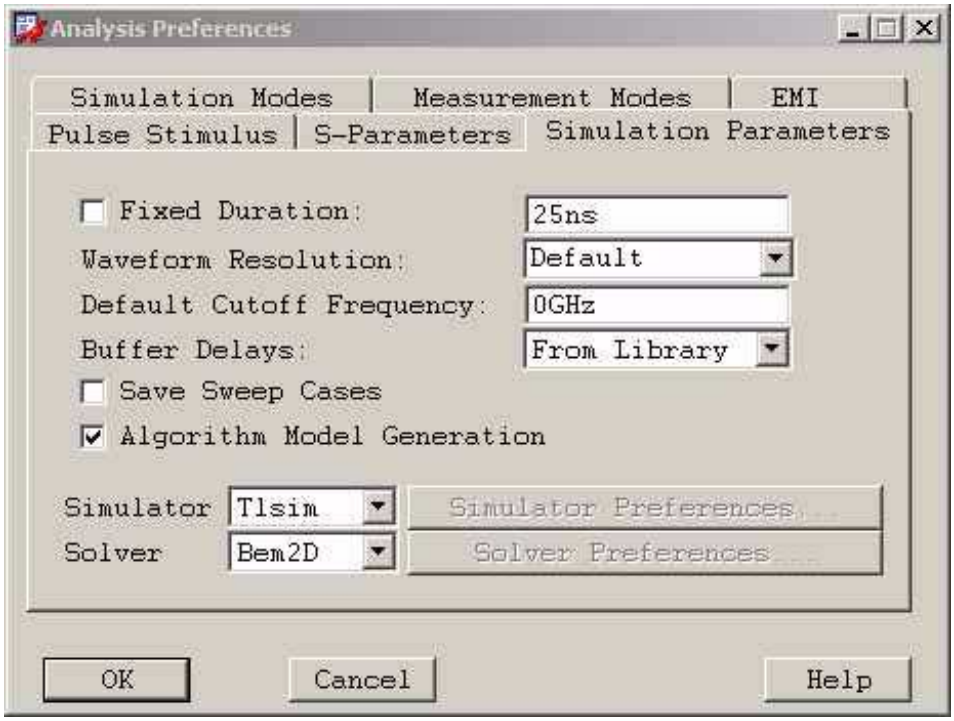
Saved sweep cases comprise the following data:

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

Saving Sweep Cases

When SigXplorer starts, 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 3-7.

Figure 3-7 Analysis Preferences Dialog Box



To save sweep cases

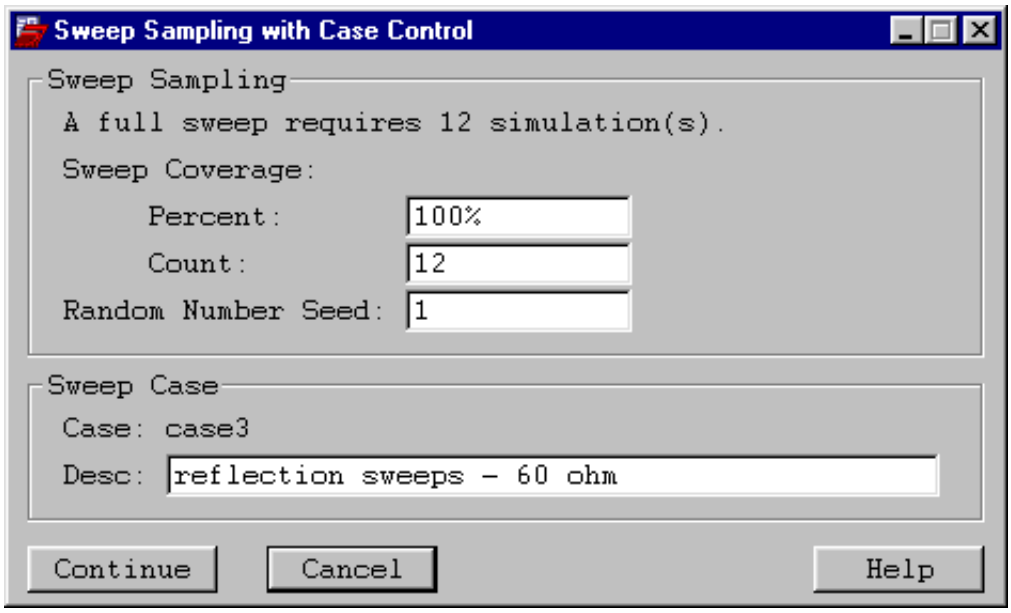
1. Choose *Analyze – Preferences* in SigXplorer.

   The Analysis Preferences dialog box appears.
2. Select the Simulation Parameters tab.
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 3-8.

Figure 3-8 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 chain of events occurs to preserve the current sweep simulation data and environment details.

- Current topology file and SigNoise preferences save to 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 also saves to 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, case3) as the directory for the saved sweep data.

Restoring and Deleting Sweep Cases

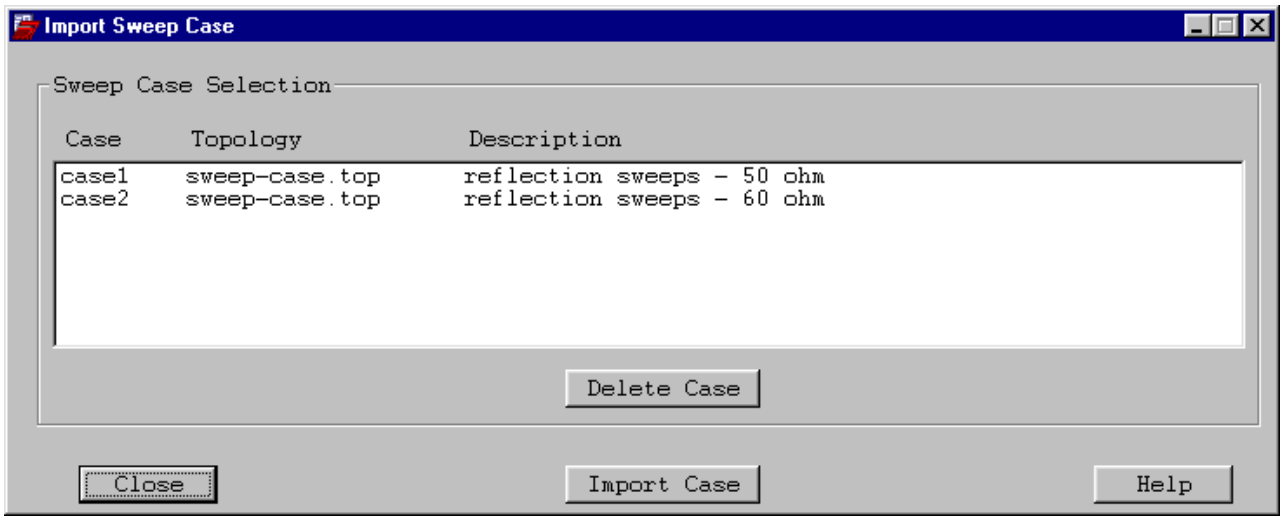
Restoring a sweep case results in the following chain of events:

- Message appears prompting you to save the current topology in SigXplorer.

- Topology file from the selected case loads into SigXplorer.
- Results spreadsheet data from the selected case imports into the Results spreadsheet within SigXplorer.
- SigNoise preferences restore from the selected case.

Saved sweep cases restore using the **Import Sweep Case** dialog box shown in Figure 3-9.

Figure 3-9 Import Sweep Case Dialog Box



To restore a sweep case:

1. Select *File>Import>Sweep Case* - The Import Sweep Case dialog box appears.
2. Click the sweep case you wish to restore from the list.

   The case highlights.
3. Click *Import Case*.

   The case data is imported into *SigXplorer*.

To delete a sweep case:

1. Click the sweep case you wish to delete from the list. The case highlights.
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:

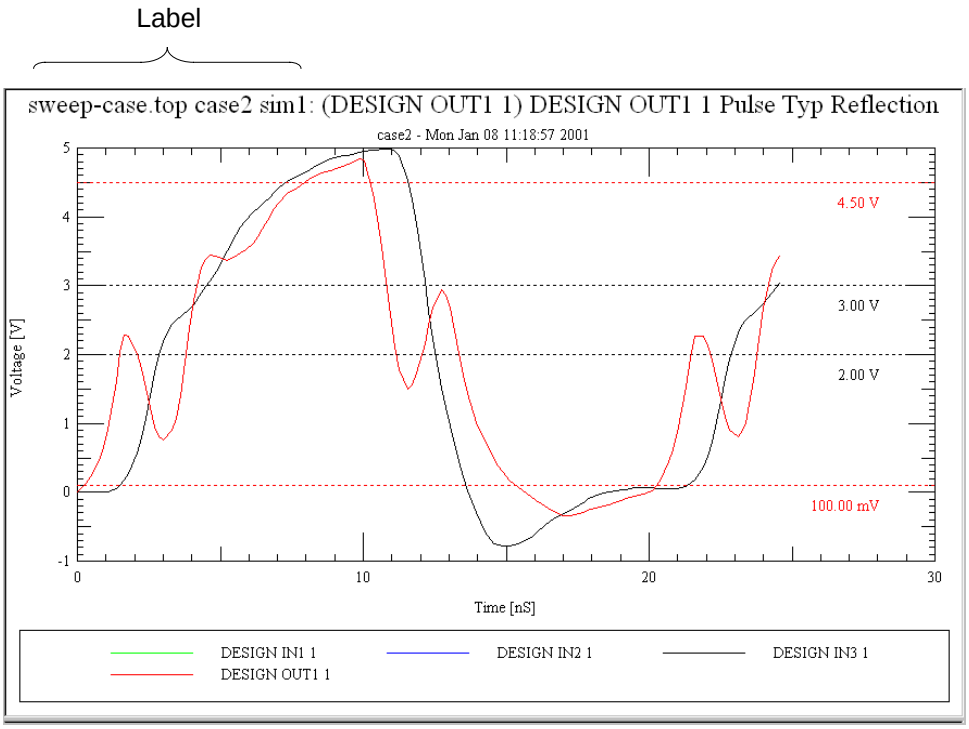
1. Right-click on the desired simulation row within the Results spreadsheet.
2. Click the*View Waveform* button.

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

Waveform Labels

Each waveform has its own label to assist 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 3-10 shows a waveform label.

Figure 3-10 Waveform Labeling



Crossprobing

Crossprobing means to either graphically identify (highlight) or select a design object through the selection of that object via some external source (other than the object's native graphics pane). For example, the selection source may be a list of design objects displayed in a dialog box or possibly the graphics pane of a related application. You enable the selection bidirectionally.

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* displays simultaneously.

2. Click on 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 highlights.

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.

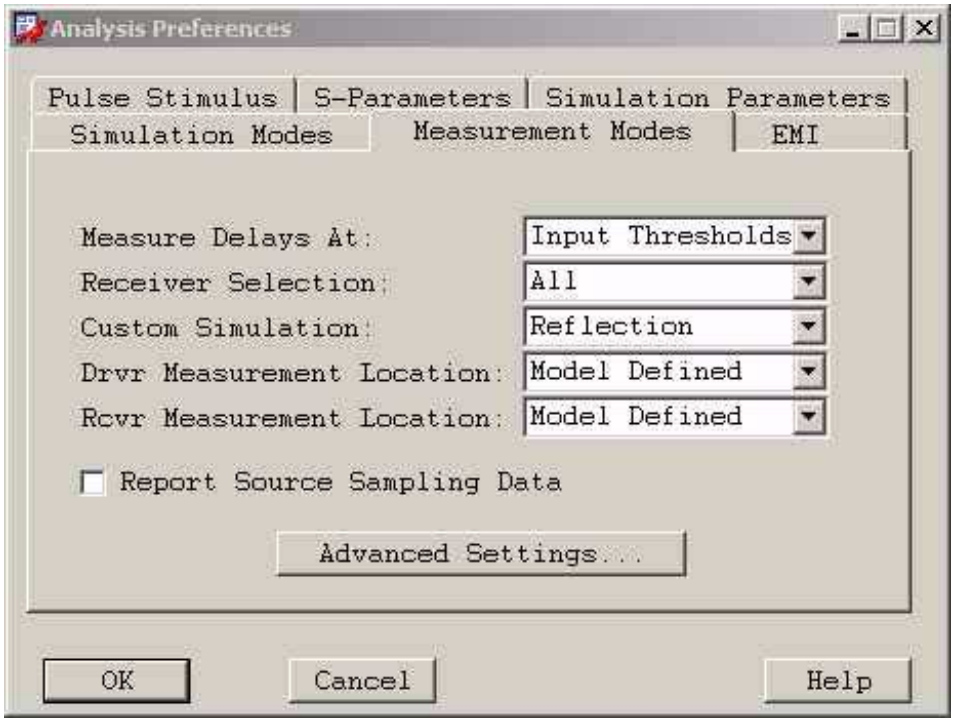
2. Click on a simulation row within the Results spreadsheet of SigXplorer.

   The corresponding waveform object in SigWave highlights.

Advanced Settings

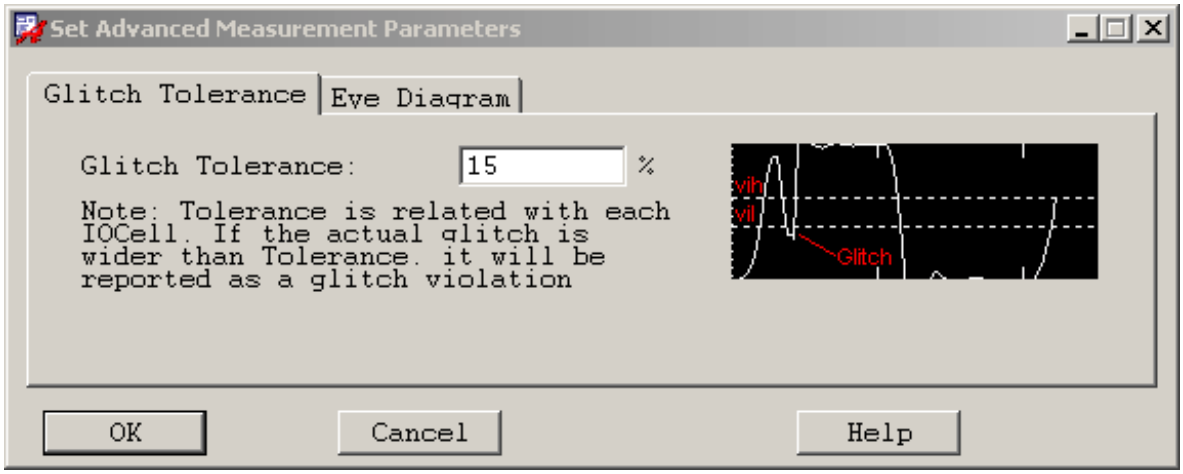
Click the *Advanced Settings* button in the Measurement Modes tab to access the Advanced Measurement Parameters dialog box.

Figure 3-11 Advanced Setting Control



From here, you can set measurement parameters that govern glitch tolerance and measure eye opening and peak-to-peak jitter, both of which can assist you in finding correct cycles in your waveform.

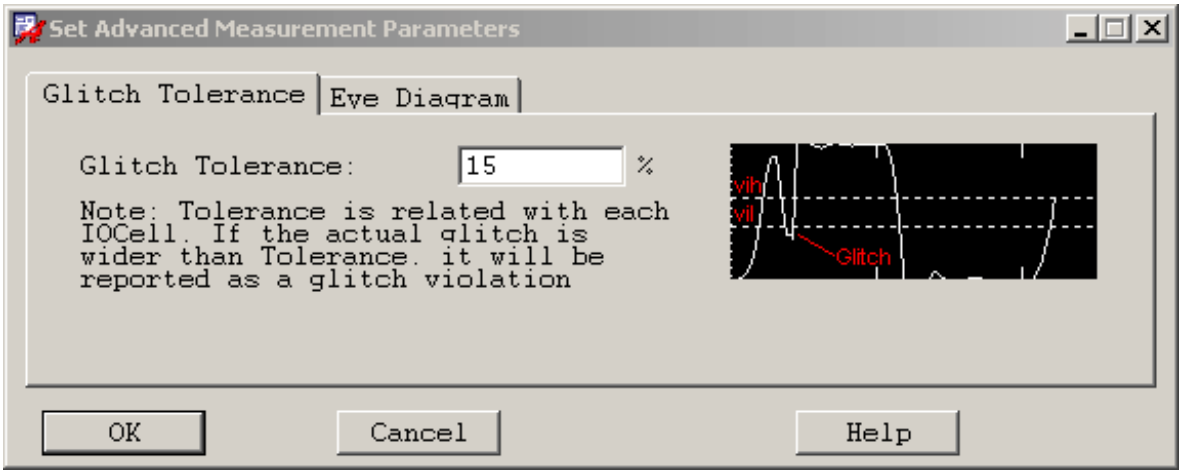
Figure 3-12 Set Advanced Measurement Parameters Dialog Box



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.

Figure 3-13 Glitch Control Tab



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 entered in the Glitch Tolerance field. 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 that tool's user interface:

- *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.

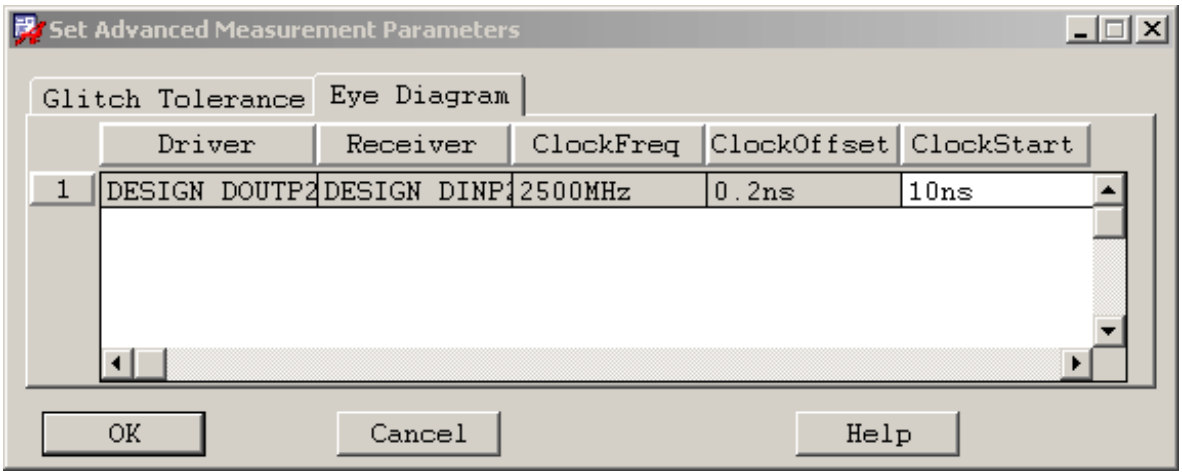
Glitch tolerance values are saved in the topology file and in the `sigxp.run` case management directory. If the tolerance values in these locations differ, the tolerance in the topology file takes precedent.

Access this control from the *Advanced Settings* button in the Measurement Modes tab of the Analysis Preferences dialog box.

Eye Diagram Measurements

To measure the eye diagrams of drivers which have a custom stimulus (that is, a stimulus other than pulse, rise, fall, etc.), eye diagram measurements report the horizontal and vertical eye opening and peak-to-peak jitter within wave forms. When you check the EyeHeight, EyeJitter, and EyeWidth items in the Reflection section of the *Measurements* spreadsheet in the SigXplorer GUI, the measurements are displayed in the *Results* spreadsheet following simulation. The Eye Diagram spreadsheet in the Set Advanced Measurements dialog box displays the current eye diagram parameter settings for the combinations of the topology's drivers and receivers.

Figure 3-14 Eye Diagram Tab



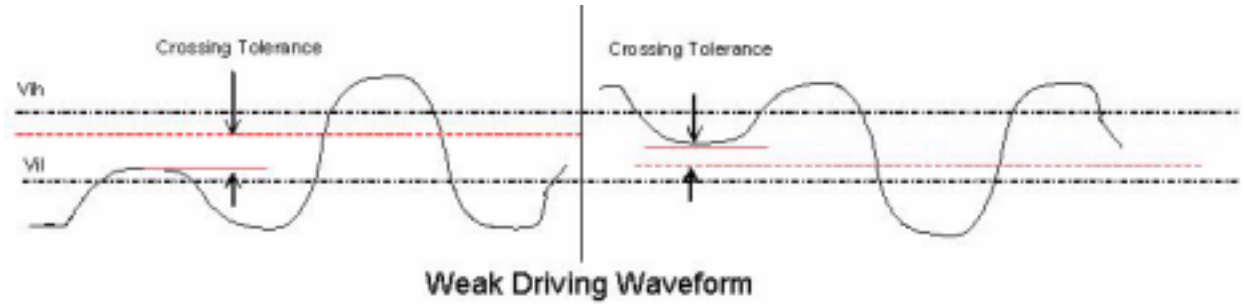
- *Driver* and *Receiver* display the driver/receiver combinations in the topology.
- *ClockFreq* displays the value of the Custom Stimulus state set in the IOCell Stimulus Edit dialog box
- *ClockOffset* displays the value in nanoseconds of 1/2 the clock frequency value.
- *ClockStart* is editable and lets you define the point in time that the eye pattern data should start. The value defaulte is *0ns*.

See the procedure for performing this 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 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 3-15.

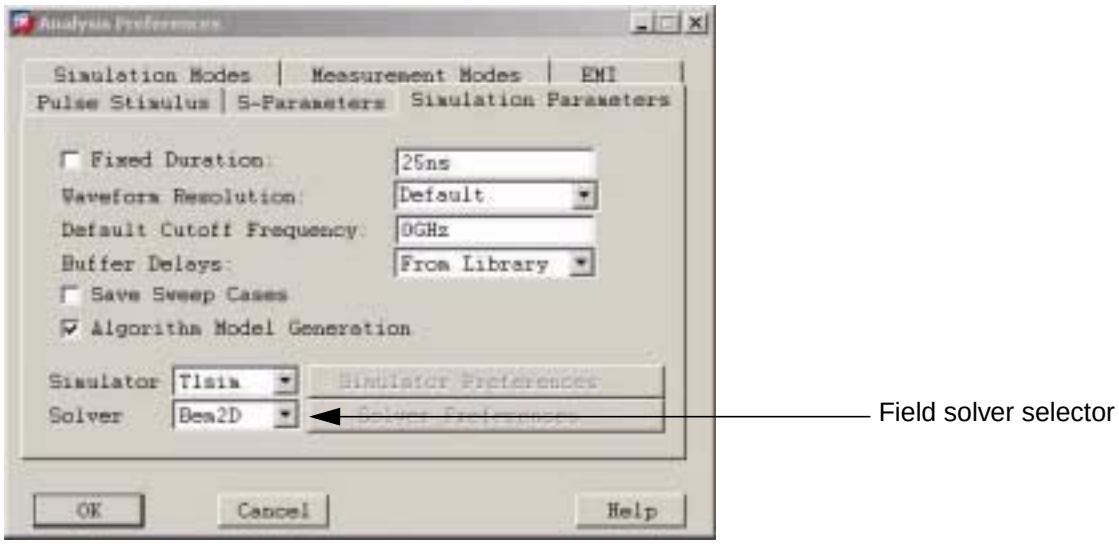
Figure 3-15 Weak Driving Control



Full Wave Field Solvers

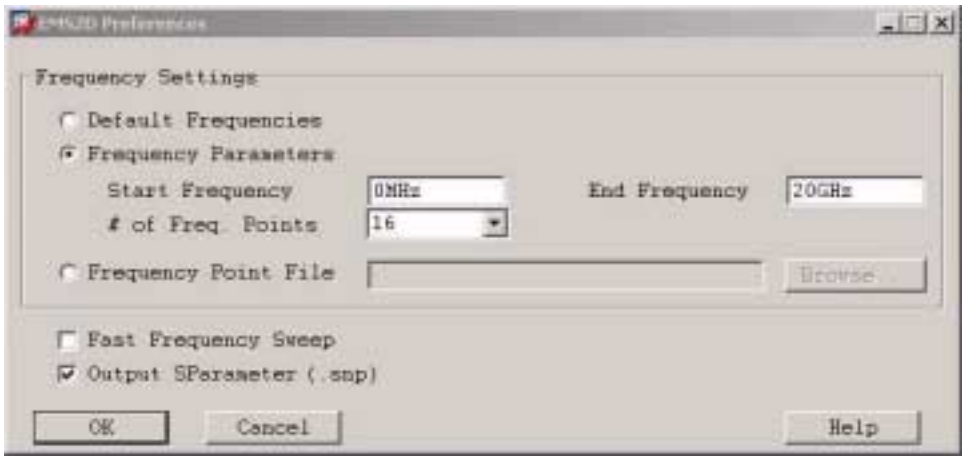
SigXplorer supports two field solvers, Bem2d and Ems2d. New models that you place in SigXplorer, either from PCB SI or by way of the *Add Part* functionality, will attempt to use the solver selected in the Simulation Parameters tab of the Analysis Preferences form (shown in Figure 3-16). Pre-existing models will attempt to use the field solver type initially used to solve the model.

Figure 3-16 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.

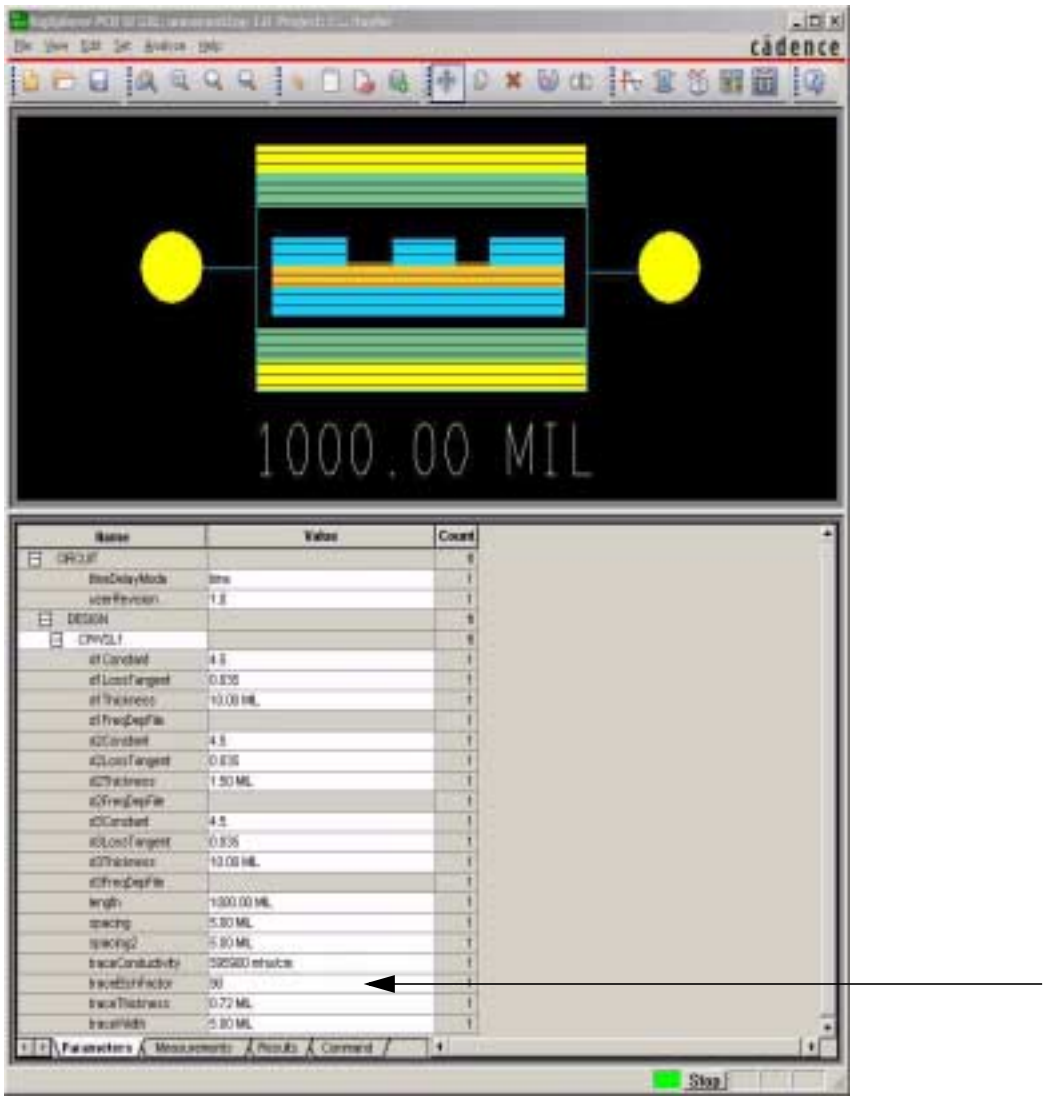
Figure 3-17 EMS2D Preferences Form



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

The field solver type is displayed in the Parameters spredsheet for the model associated with the design element, as shown in Figure 3-18.

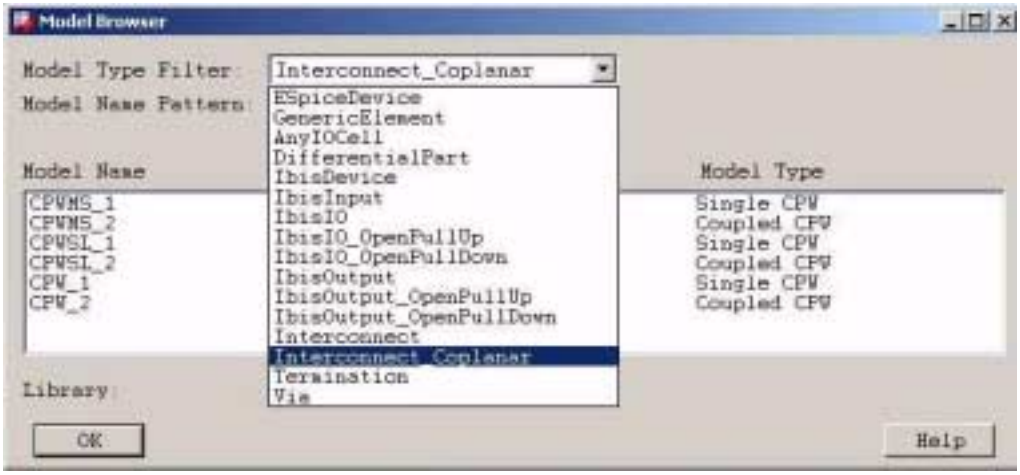
Figure 3-18 Field Solver Entry in Parameters Spreadsheet



Coplanar Waveguide Support

SigXplorer provides full-time support for coplanar waveguide structures in a Help topology, whether extracted from a board layout or added directly to the canvas by way of the Model Browser (*Edit – Add Part*), shown in Figure 3-19.

Figure 3-19 CPW Support in the Model Browser Form

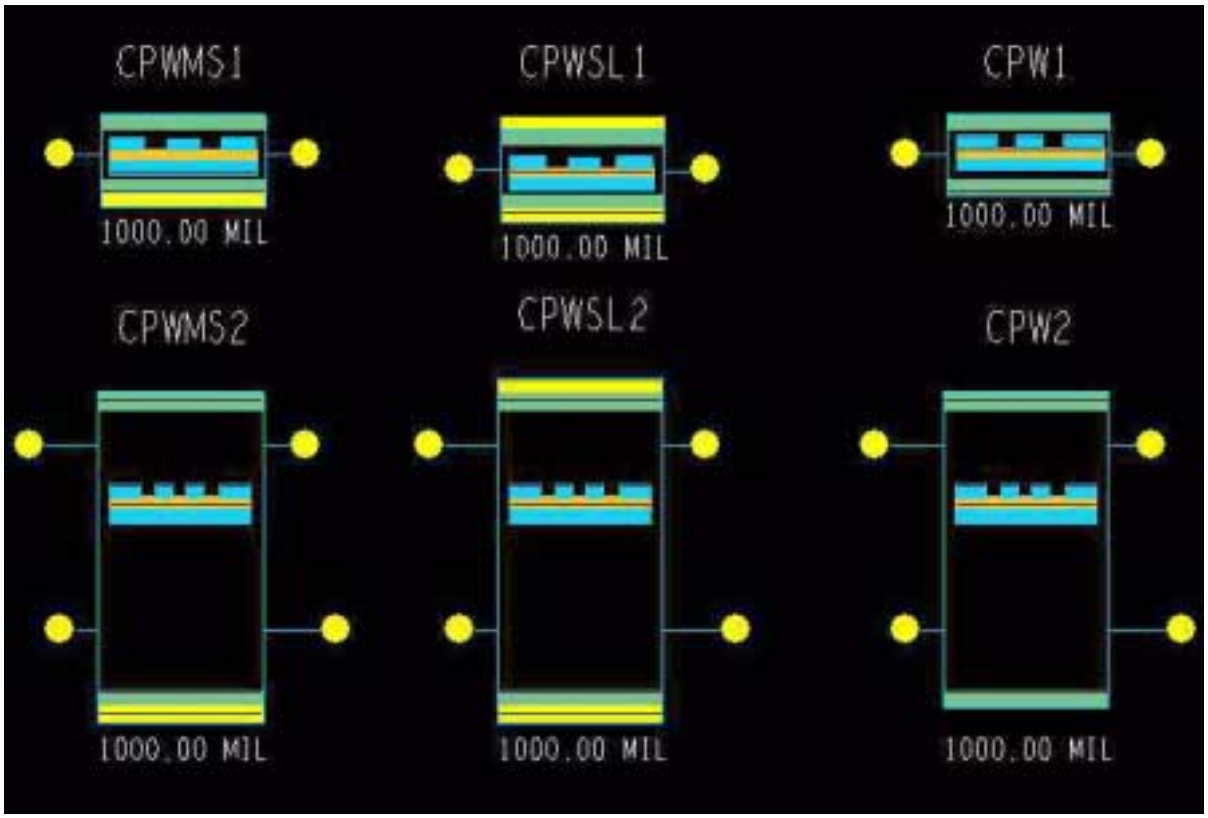


Six CPW structures are supported in SigXplorer. They are:

- 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 3-20.

Figure 3-20 Coplanar Waveguide Symbols



4

Assigning Constraints

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.

Use the Set Topology Constraints dialog box to create and modify topology constraints. See the *SigXplorer Command Reference*.

You can write these modified constraint values back to the design database by selecting *File – Update* inSigXplorer.

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, diff 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.

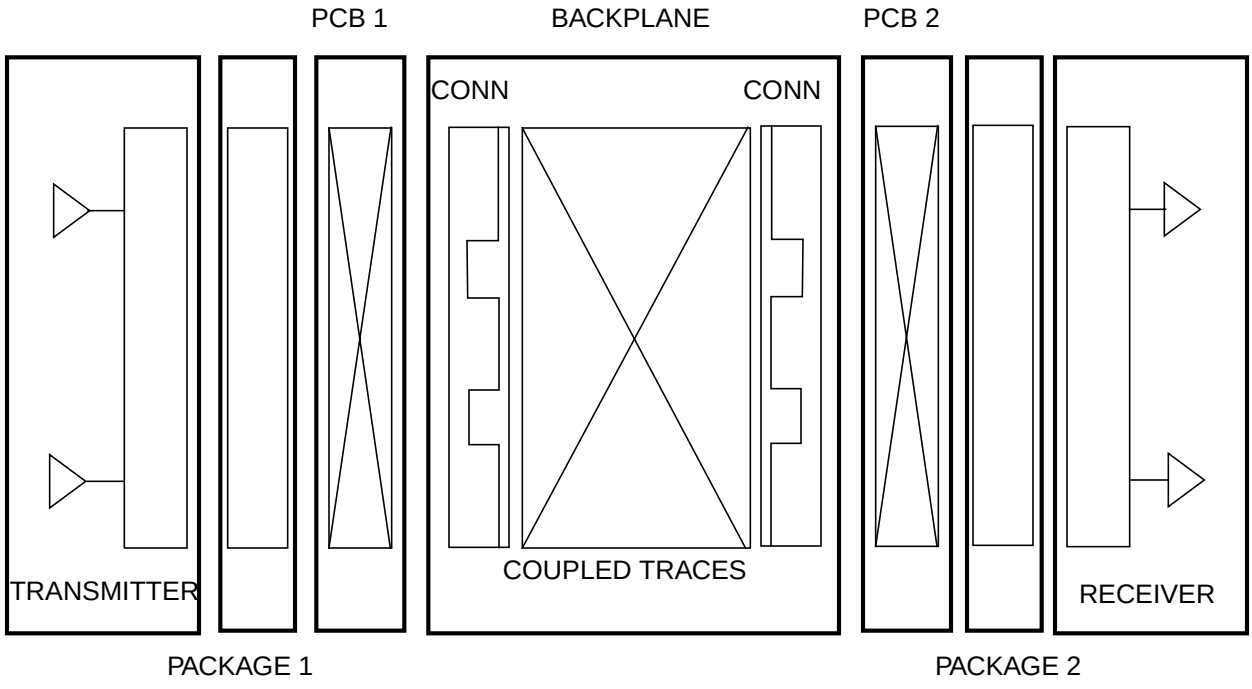
Relative Propagation Delay	Defines connections that are part of a match group. You can specify relative propagation delays between nets and within a net, as well. Assigning the PROPAGATION_DELAY property to one of the connections in a match group restricts all other connections in the group.
Differential Pair	Assigns diff pair rules to diff 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 contraint rules.
Usage	Displays application-specific information on constraint usage for the current topology analysis.
Layer Sets	Defines the layer set data for the current topology and lets you edit entire layer sets and/or the individual layers within a set.

5

Using Channel Analysis

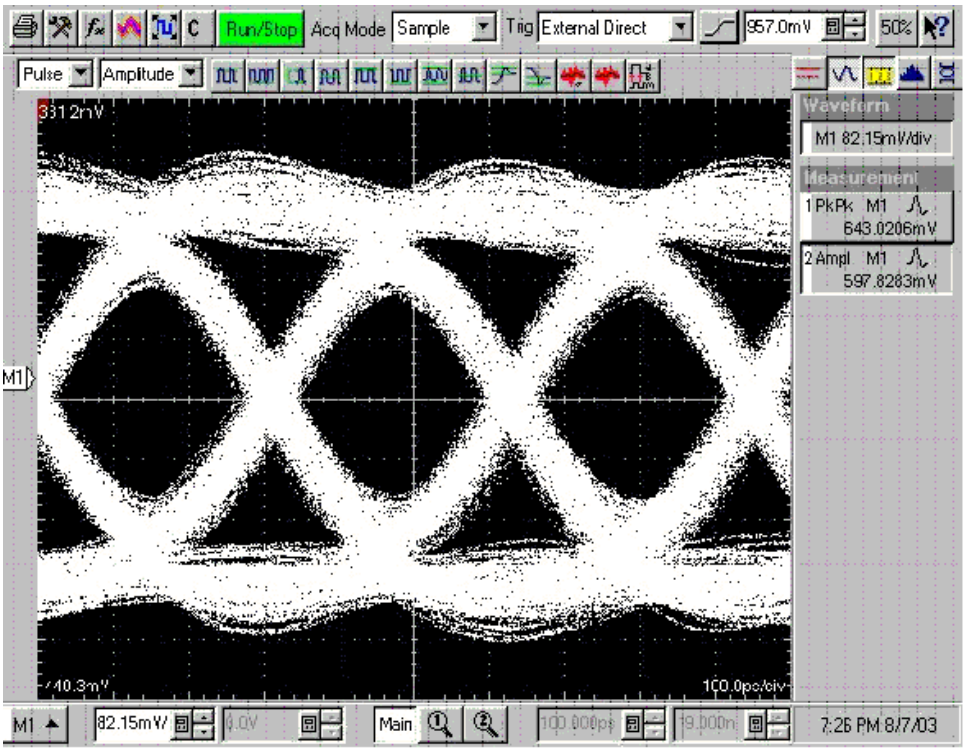
For serial data running at rates over 1 Gigabit per second (Gbps), the chip-to-chip signal path of the differential pair is often referred to as the “channel.” This channel may be made up of multiple printed circuit boards (PCBs), packages, connectors, and backplanes.

Figure 5-1 Archetypical Serial Data Channel



A channel 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.

Figure 5-2 Measured Eye Pattern



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 channel's parasitic influence 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.

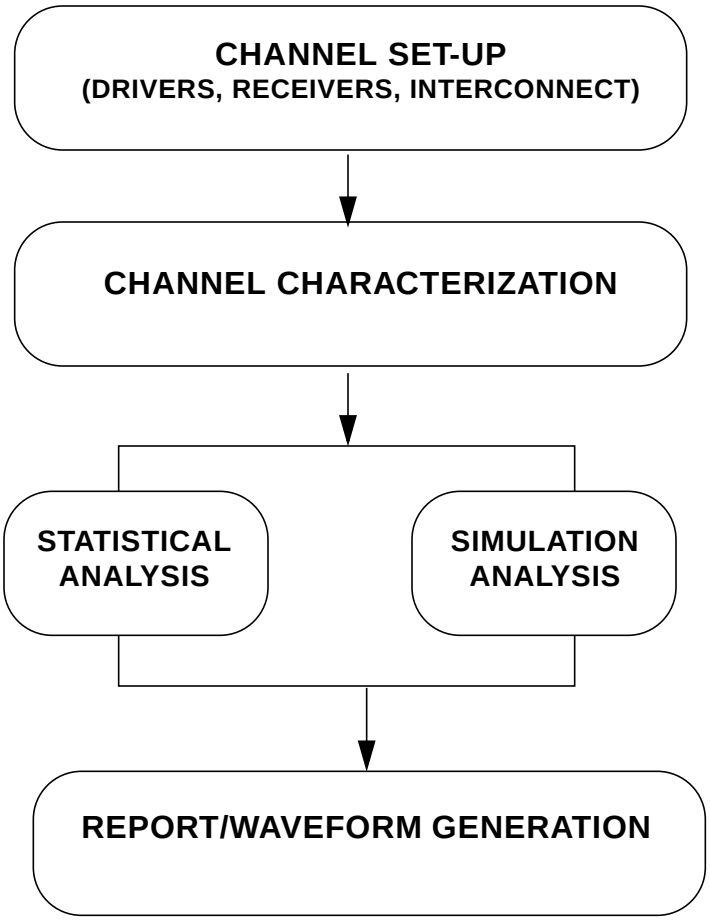
- High capacity simulation

  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 5-3 illustrates the high-level components of the Channel Analysis functionality in SigXplorer.

Figure 5-3 Channel Analysis Work Flow



Your Cadence documentation includes the “Channel Analysis Introductory Tutorial.” Select *Help – Documentation* in the SigXplorer menu bar and click the *Tutorial* tab.

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 CA. You determine this by using the special prefix name `RXin_diff` in a node_param statement in the receiver’s MacroModel. This can be any node in the receiver MacroModel; for example, deep inside the model after an equalization circuit. See the note, below, for an example of this syntax in the `diff_8term.dml` template. If this special prefix is *not* found in the primary receiver’s MacroModel, then the node of interest for CA is automatically determined as follows:

   - If a PackageModel or PinParasitics are used, then 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, then 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. Select *Analyze – Preferences* in SigXp 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. We recommend10GHz 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* pull-down menu

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 87.

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

Using the Algorithmic Modeling Interface

You can integrate vendor-specific modeling algorithms using the Algorithmic Modeling Interface incorporated in the CA work flow. 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 ) )
```

   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 signal_analysis section of the Allegro User Preferences Editor. (Choose *Setup – Preferences* in the menu bar of PCB SI to access the User Preferences Editor.)

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, either from the user interface or in batch mode.

Running Channel Analysis from SigXplorer

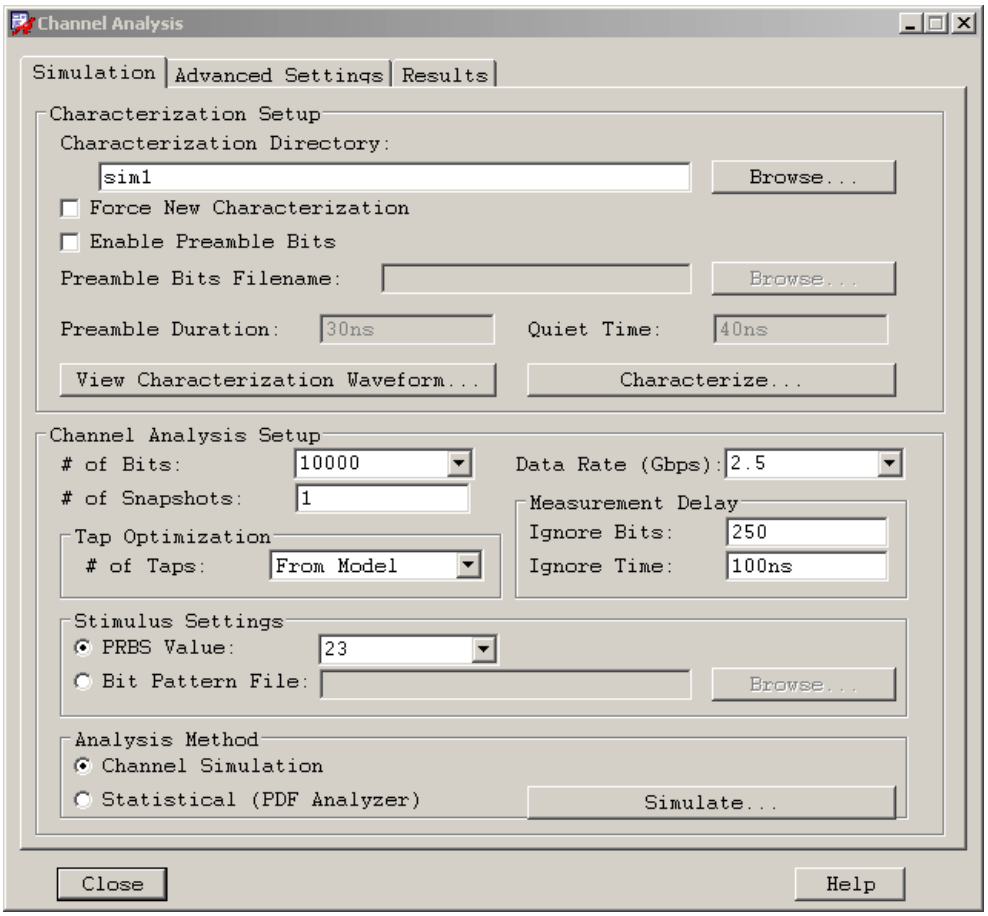
You access Channel Analysis by selecting *Analyze – Channel Analysis* in the SigXplorer menu bar to display CA's graphical user interface (GUI), or by running CA in "batch" mode from your operating system's command prompt. This section describes the controls you set when you run CA in GUI mode. (See Running Channel Analysis from Your System Command Prompt on page 82 for details on how to configure and run CA in batch mode.)

Channel Analysis Characterization and Simulation

The Channel Analysis graphical user interface is composed of three tabs: *Simulation, Advanced Settings,* and *Results*. The *Simulation* tab contains the controls that enable you to characterize and simulate your circuit.

Note: The circuit simulator you use for characterization is dependent on what you select in the *Simulator* pull-down menu in the *Simulation Parameters* tab of the Analysis Preferences dialog box. This does not apply if you select the statistical analysis option.

Figure 5-4 Channel Analysis GUI Simulation Tab



Characterization Setup

Before you can simulate a circuit, you must perform a characterization of it. 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 are responsible for re-characterization when the topology of your circuit is edited; this function is *not* automatically tracked by CA.

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

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. These will 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.

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:

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 analyses based on the number of taps assumed in the driver's pre-emphasis circuit.

# 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, which means that no additional what-if taps are included in the analysis; just whatever is already defined in the driver model itself.

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 simply 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, then 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.

CrossTalk

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 are not present in the topology, the *CrossTalk Mode* setting is ignored. For additional information on crosstalk effects, see Incorporating Crosstalk Effects into Channel Analysis on page 87.

Measurement Delay

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 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 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:

PRBS Value — Specifies the pseudo-random bit sequence (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

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 5-5 Channel Analysis GUI Advanced Settings Tab

Advanced Channel Analysis Setup Section

Random Jitter (%UI) — Specifies the percent unit interval (UI) of jitter you want to post-process in a Gaussian distribution.

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 87.

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 unabled, 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.

Advanced Stimulus Settings Section

Freq Offset — Lets you enter 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.

HI (%) — The value for the 1 bit, as a percentage. The default condition is 50%.

LO (%) — This field is read-only.

Periodic Jitter

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

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.

Channel Analysis Results

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 88.

Figure 5-6 Channel Analysis GUI Results Tab

Display Results from Default Results Directory

When checked, the Results Browser will automatically display the results associated with the governing characterization. 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:

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.

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 of the tab page, 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.

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:

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.

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.

Cancel — Closes the form.

Help — Displays user documentation on Channel Analysis.

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 66 before performing the procedure described below.

1. Using SigXplorer, open a topology (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 68.
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 Channel Analysis Results on page 77.

Running Channel Analysis from Your System Command Prompt

Running Channel Analysis in GUI mode is intended for mainstream usage, and may be the optimum way for you to use this functionality. However, if additional capability is available if you run CA in batch mode at your operating system prompt. Doing so provides a superset of the functionality available through the GUI, and may be advantageous if your requirements dictate enhanced results.

Command line usage of CA falls into two main categories:

- Advanced usage

  Minor extensions to what is available from the GUI, typically handled through edits to the `channel.cfg` file

- Expert usage

  Extensive utilization of CA command line arguments

Note: Before running Channel Analysis (either through the GUI or from a system prompt), you must set up the appropriate topology in SigXplorer, as you would for traditional circuit simulation. See Prerequisites to Running Channel Analysis on page 66 before running CA.

Configuring Channel Analysis for Command Line Usage

CA is invoked is through the CircuitLab infrastructure, utilizing the `cktlab` executable:

```
cktlab <CDS_INST_DIR>\share\pcb\chsim\chsim.lsp
```

Because it is unwieldy to include the entire path to `chsim.lsp`, we recommend you do *not* edit this standard file; rather, the following process is recommended:

1. Copy the file `chsim.lsp` from the path described above to your `channel.run` directory (or other local location), naming it `my_chsim.lsp`.

2. Using a text editor, edit `my_chsim.lsp`.
3. Modify the LIBPATH variable at the top of this file to specify the full path in your installation to `<CDS_INST_DIR>\share\pcb\chsim`. For example, if your <CDS_INST_DIR> is `D:\usr\cds`, then the first few lines of your `my_chsim.lsp` file will look like:

```
(chsim
;
; automate impulse response generation
;
(Options (Precision 12) (LIBPATH “D:\usr\cds\share\pcb\chsim”))
```

This will allow you to execute CircuitLab from the command line:

```
cktlab my_chsim.lsp
```

Note: Meta characters such as $CDS_INST_DIR are not supported in command line usage for CA. You must either explicitly define all paths (unless the required files are found locally) or use the LIBPATH option at the command line, which eliminates the need for you to edit the `chsim.lsp` script. The following example of the LIBPATH method is functionally equivalent to the example above:

```
cktlab -LIBPATH D:\usr\cds\share\pcb\chsim chsim.lsp
```

Note: You must still copy locally the `chsim.lsp` file in order for it to be called by the `cktlab` executable from your command line; however, the LIBPATH method will find other scripts that are, in turn, called by `chsim.lsp`.

You can use either of these command line approaches with customized correlation scripts. Copy the default script `chcorr.lsp` to the `channel.run` directory as `my_chcorr.lsp`. Then edit the file as desired, and invoke it directly by `cktlab`, as shown in Running Channel Analysis with Command Line Arguments on page 84.

Running Channel Analysis with a channel.cfg File

If you are an advanced user who generally runs CA from the GUI, but you occasionally want to stretch beyond it’s limitations, running CA from the command line with a `channel.cfg` file may be a convenient approach for you. The `channel.cfg` file is automatically written into the `sigxp.run\case0\channel.run` directory when you run CA from the GUI. It contains all the settings that are passed to `cktlab` from the control settings in the GUI. When a `channel.cfg` file is found locally, its settings are automatically passed to `cktlab`. Therefore, the `channel.cfg` file is a convenient way of passing multiple arguments to `cktlab`.

As an example, lets assume you want to run a 7-tap what-if simulation with CA, but the GUI only provides a pull-down for up to six taps. You could perform the following procedure:

1. Run CA from the GUI for the six tap case.
2. Open a command window in the `sigxp.run\case0\channel.run` directory.
3. Copy your `my_chsim.lsp` into `channel.run`, modified as described in the previous section, or use the LIBPATH option.
4. Using a text editor, edit the file `channel.cfg` to change the value for the parameter NoofTaps from 6 to 7, then save the file.
5. In the `channel.run` directory, type the command `cktlab my_chsim.lsp`. This will rerun the channel simulation for the 7-tap case, per the updated `channel.cfg` file.

Running Channel Analysis with Command Line Arguments

The `channel.cfg` file is a convenient way to pass command line arguments to CA. To see a listing of the arguments available, simply type:

```
cktlab h my_chsim.lsp
```

This will produce the following output:

```
 cktlab [-h] [-start <start_time>] [-stop <stop_time>] [-startb
<start_time_in_bits>] [-stopb <stop_time_in_bits>] [-bitp
<bitperiod>] [-gbps <data_rate_in_gbps>] [-tr <risetime>] [-_8b10b]
[-poly <poly_val>] [-taps <#taps>] [-md <measure_delay>] [-jit
<jitter>] [-lib <lib_path>] [-fpath <results_dir>] [-xtalkmode
<random|even|odd>] [-seed <seed_number>] my_chsim.lsp
```

Default values for the associated switches are:

`-h`	Displays this help description
`<start_time>`	0
`<stop_time>`	4000n, now superseded by stopb
`<start_time_in_bits>`	Nil
`stop_time_in_bits>`	10000; a valid non-nil specification in bit time will be the primary basis for start and stop actions; for example, in the default case the stop will be based on 10000 bits

<bitperiod>	400p
<data_rate_in_gbps>	Nil; if not nil supersedes <bitperiod>
<risetime>	70p; can be used to explicitly set driver rise time
<poly_val>	Polynomial for PRBS; default is 23
-_8b10b	Enables option for 8b10b coding; default is off
<#taps>	Option to run simulation with an optimized n tap driver
<measure_delay>	Delay before measurements are made; default is 100n
<jitter>	Dimensionless jitter; default is nil
<lib_path>	Path to characterization data; must be taken from channel.cfg file or explicitly specified
<fpath>	Path to write output data; must be taken from channel.cfg file or explicitly specified
<random>	Each driver gets its own unique PRBS
<seed_number>	Random seed number; default is 1

Note: Option names *must* be lower case. Capitalized options are reserved for debugging purposed only.

Some example command line options that you may find useful are:

- Producing raw time domain waveforms instead of just contours (beware of performance issues with SigWave):

  ```
  cktlab -dbg 1 -stopb <stop_bits> my_chsim.lsp
  ```

- Incorporating a custom stimulus file into the Channel Simulation:

  ```
  cktlab -bitsf <full_path_to_mybits.txt> my_chsim.lsp
  ```

- Incorporating a custom correlation script, which runs for a simulation time of 400ns:

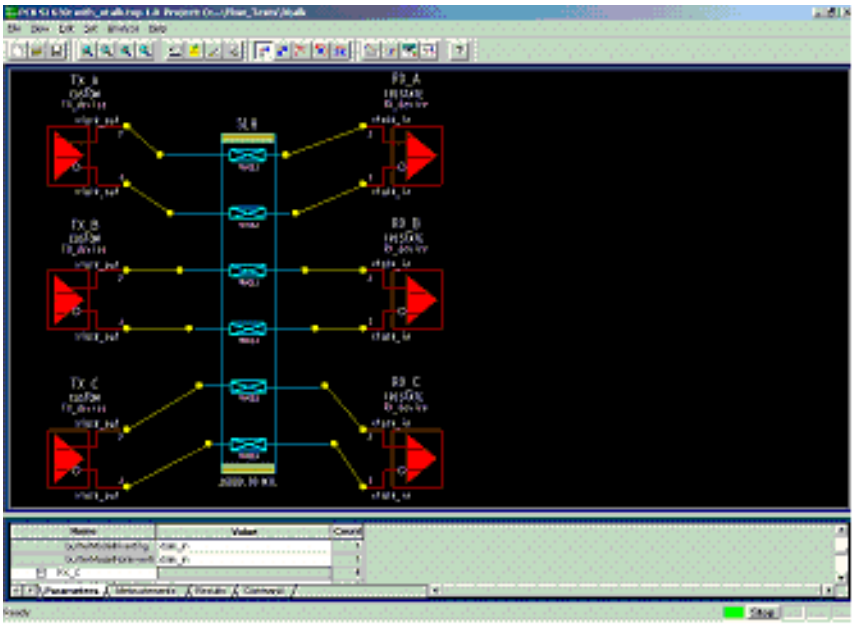
  ```
  cktlab -stop 400n my_chcorr.lsp or
  cktlab -stop 400n <full_path_to_my_chcorr.lsp>
  ```

To learn a step-by-step procedure on how to produce full-time domain waveforms using `cktlab`, see "Producing Full-Time Domain Waveforms" on page 89.

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 5-7.

Figure 5-7 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 66.

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 5-7, 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 71).

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
```

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.

Producing Full-Time Domain Waveforms

This chapter describes how you can produce full-time domain waveforms with Channel Analysis.

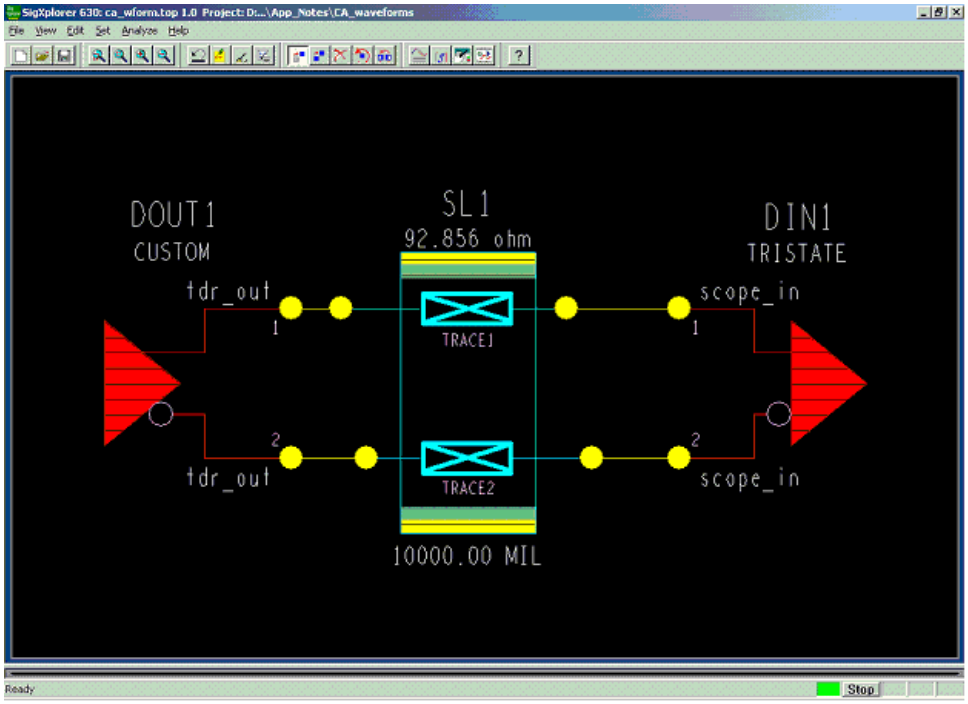
Note: Because CA does not automatically produce full waveforms, you must be able to work from your operating system's command line to effect these results.

For correlation purposes, you may find it useful to produce full-time domain waveforms from CA using small, specific bit streams as stimulus. The procedure for doing so consists of:

1. Running a standard CA simulation from the graphical user interface (GUI)
2. Running *cktlab* to produce full waveforms
3. Adjusting the simulation duration and repeating the procedure as necessary

The topology shown below is the example used throughout this chapter. It is built from elements in the CA tutorial library.

Figure 5-8 Example Topology



Step 1: Run a Standard Channel Analysis Simulation from the GUI

Running a standard CA simulation will set up the directory structure and all the files you will need. From these files, you can make edits and run various commands to produce the desired waveforms.

a. Set your preferences and run an initial circuit simulation with *tlsim* to ensure your topology is set up correctly.

b. Bring up the CA GUI and specify a characterization directory name. The name *my_char* will be used in this example.

c. Click *Characterize*.

d. When the characterization is finished, click *Simulate* with the default settings. Inspect the eye contour in SigWave to verify that characterization and simulation occurred properly.

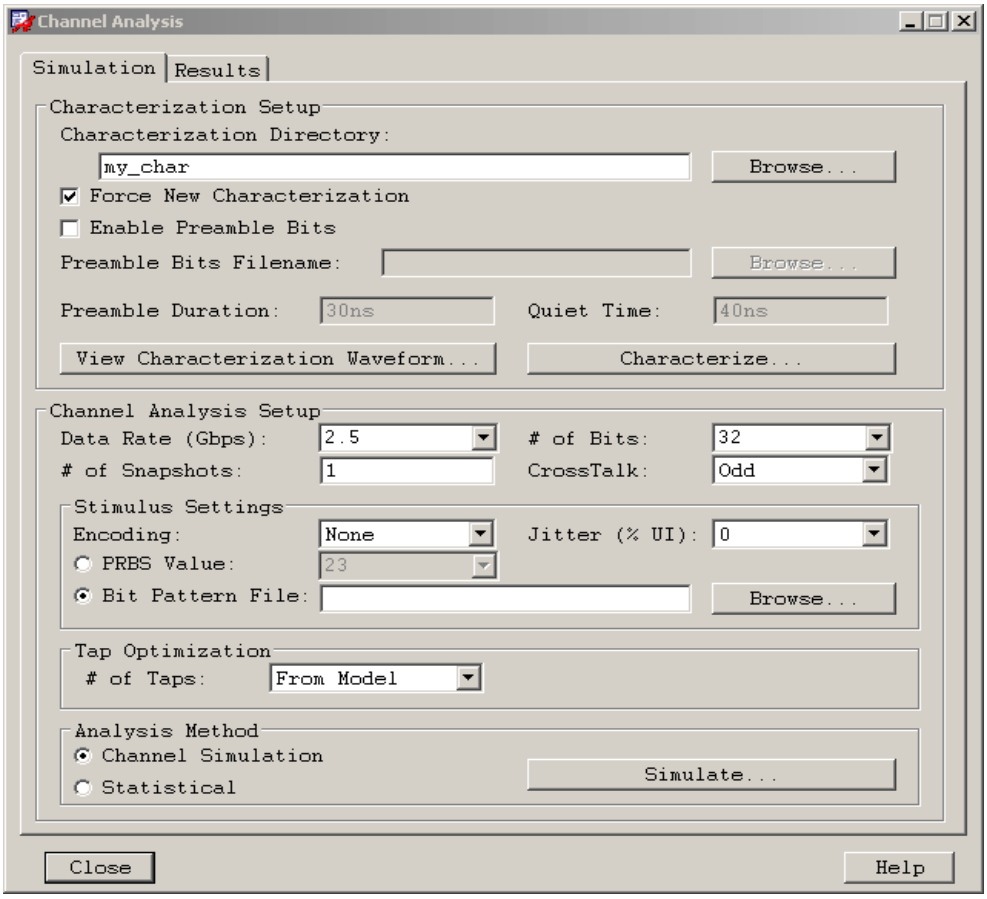
e. In the CA GUI, select *Read Bit Pattern From File* and specify the external file you want to use. In this example the file `test_pattern_32bit.txt` is used, as shown below.

```
(0 1 1 0 0 0 0 0 1 0 1 0 0 0 1 1 1 1 1 0 1 0 1 1 0 0 0 0 0 1 0 1)
```

f. Set *Data Rate* and *# of Bits* you want to use (in this case, 32 bits).

The GUI should be configured as shown in the Figure below.

Figure 5-9 Channel Analysis GUI



g. Click *Simulate*. This will run the simulation again, update the `channel.cfg` file, and produce the other directories and files you will need in `<working_dir>\sigxp.run\case0\channel.run`.

Important

Running such a small bit stream will *not* produce meaningful results in CA reports or eye contours. This procedure simply sets up the necessary input files.

Step 2: Run cktlab to Produce Waveforms

Run *cktlab* from the command line in the `sigxp.run\case0\channel.run` directory to produce waveforms. As a necessary prerequisite to this, you must configure the script `chsim.lsp` to run locally in the `channel.run` directory.

a. In your installation hierarchy, navigate to `<CDS_INST_DIR>\share\pcb\chsim` and copy the script `chsim.lsp` into your `channel.run` directory.

b. Rename the script file `my_chsim.lsp`.

c. In a text editor, edit `my_chsim.lsp` in the following manner:

➤ In the Options section, edit the LIBPATH variable to specify the full path in your installation: <CDS_INST_DIR>\share\pcb\chsim.

For example, if your <CDS_INST_DIR> is D:\usr\cds, then your `my_chsim.lsp` file would read:

```
(Options (Precision 12) (LIBPATH "D:\usr\cds\share\pcb\chsim"))
```

d. To produce the waveforms, type the following in a terminal window in your `channel.run` directory where `my_chsim.lsp` is located:

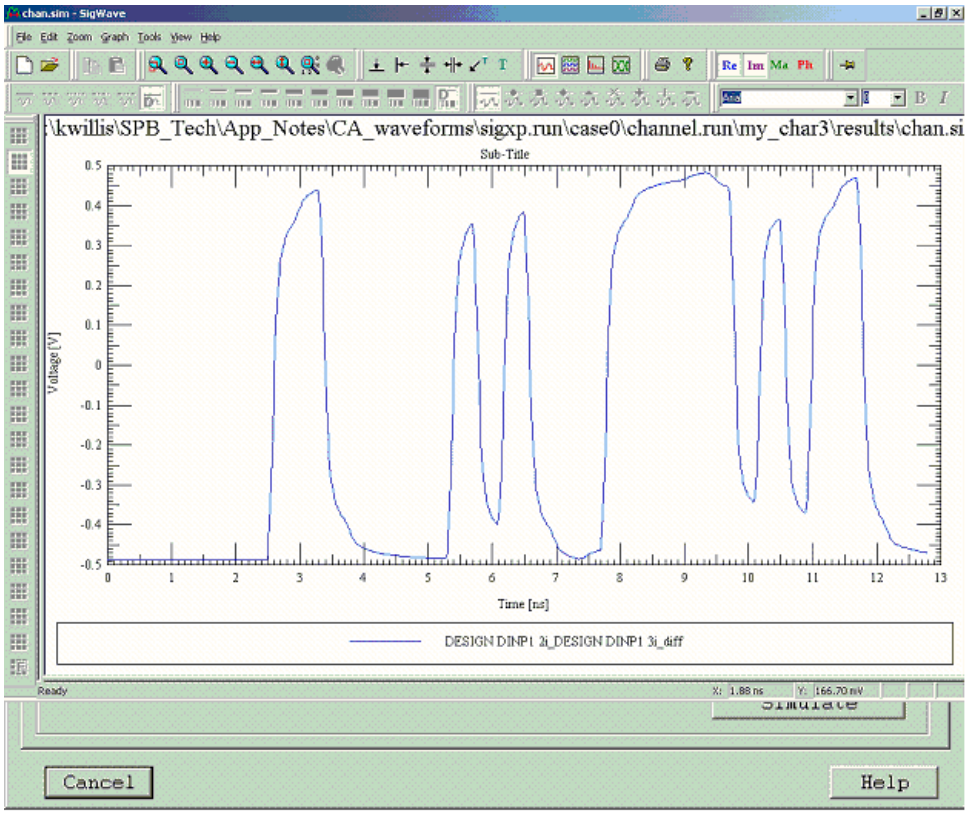
```
cktlab -dbg 1 my_chsim.lsp
```

This produces the waveform files `chan.txt` and `chan.sim` in the Results directory `...channel.run\my_char\results`.

e. Bring up the `chan.sim` file in SigWave to view the results.

An example waveform is shown in the following Figure.

Figure 5-10 First simulation waveform result



Step 3: Adjust the Simulation Duration

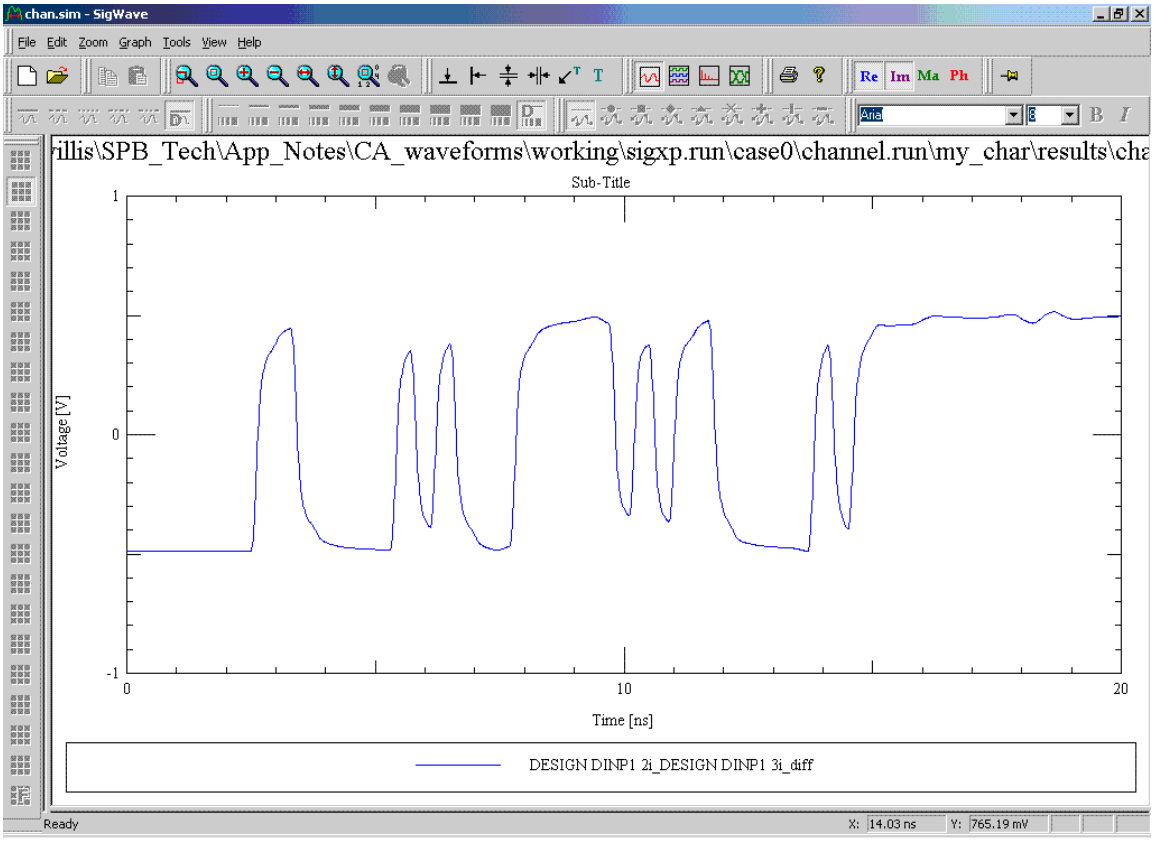
If the `chan.sim` file has run for a sufficient duration, your simulation is most likely complete. However, it is often necessary to adjust the simulation duration and rerun it. In the above Figure, you can see that the entire 32-bit pattern had not completed when the simulation stopped.

To run the simulation for a longer duration, use the `-stopb` switch with the `cktlab` command. In our current example, you could use the following command in the `channel.run` directory:

```
cktlab -dbg -stopb 50 my_chsim.lsp
```

This command reruns the channel simulator and produces a new *chan.sim* file with a longer duration of 50 bits. The result in SigWave produces the waveform shown below.

Figure 5-11 Adjusted simulation waveform result

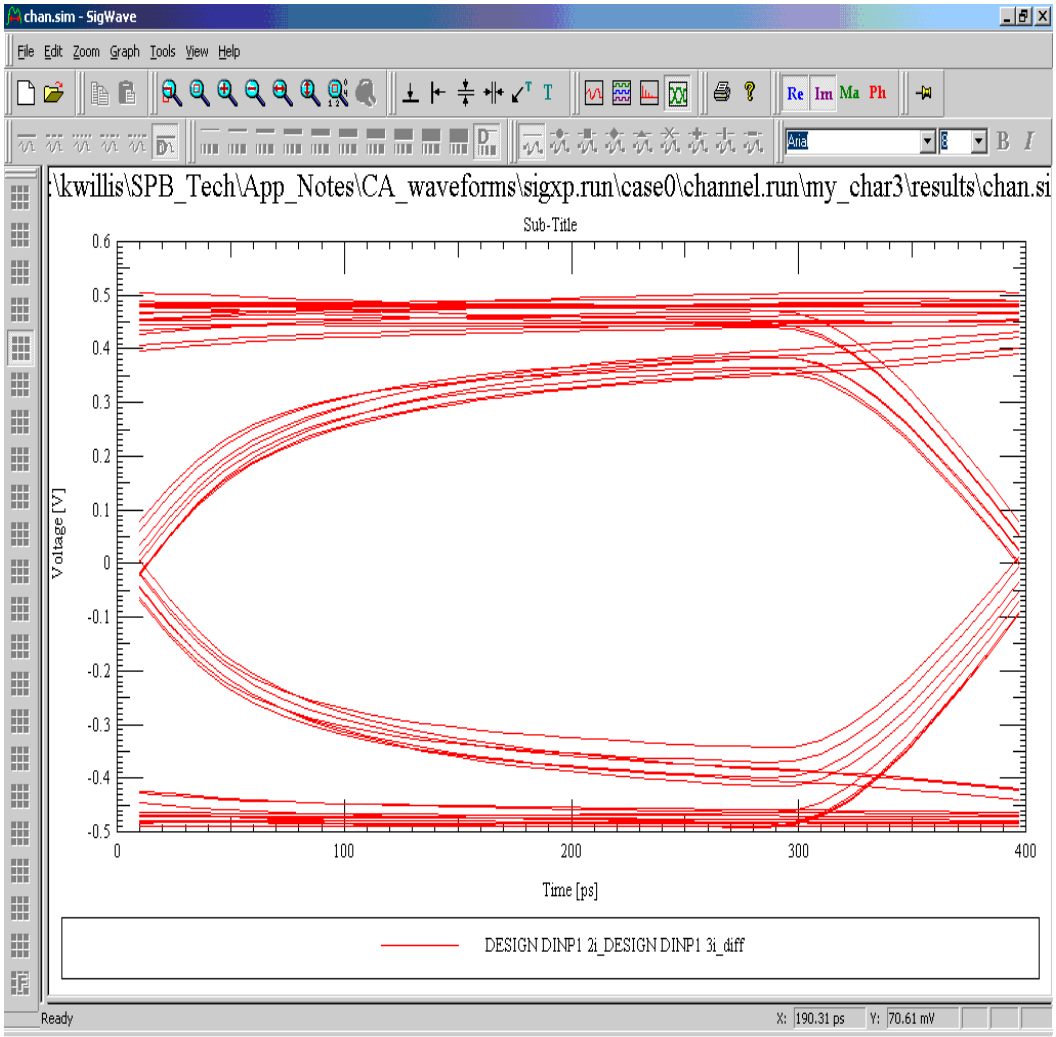


Note: Since the 32-bit pattern did not end with dots (..), the pattern should not repeat. Therefore, when the simulation was run for 50 bits instead of 32, the last bit in the pattern (in this case, 1) was simply held for the remaining bits.

Conclusion

Using this procedure, it is possible for you to produce full time domain waveforms from CA for specific bit streams. You can also view these directly in SigWave as eye patterns, as shown below.

Figure 5-12 Simulation eye pattern



6

Advanced Techniques: Custom Measurements

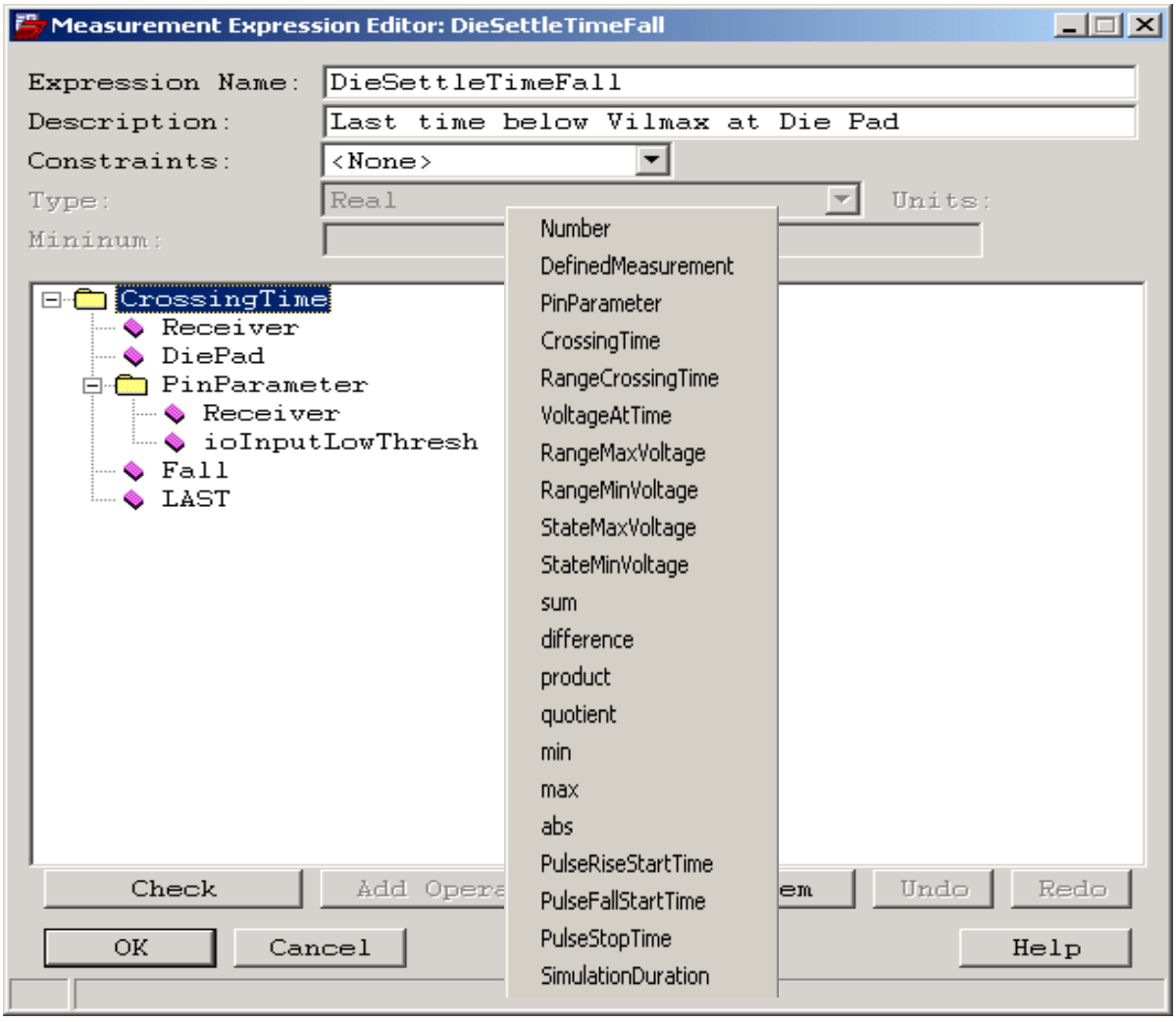
Topics in this chapter include:

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 (as shown in Figure 6-1) 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. For more information about the editor, see the *SigXplorer Command Reference*.

Figure 6-1 Measurement Expression Editor



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

Measurement Expressions

Each object in a measurement expression is a:

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 useable unit suffixes.
Reference to another measurement expression	Reference standard and user-defined measurements by their measurement expression names. When you select from the treeview 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 high Vihmin minus BufferDealyFall (Reflection measurement)
SwitchDelayRise	Time	First time to switch low below Vilmax minus BufferDealyFall (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.

7

Advanced Techniques: Multi-terminal 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. (*Analyze – Libraries – Browse Models – TextEdit*).

The following is an example of the text describing the pin order of the ESpice Device model Figure 7-1.

```
("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 7-1 Example of pin order on an ESpice Device

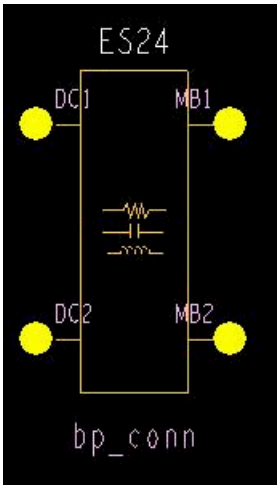
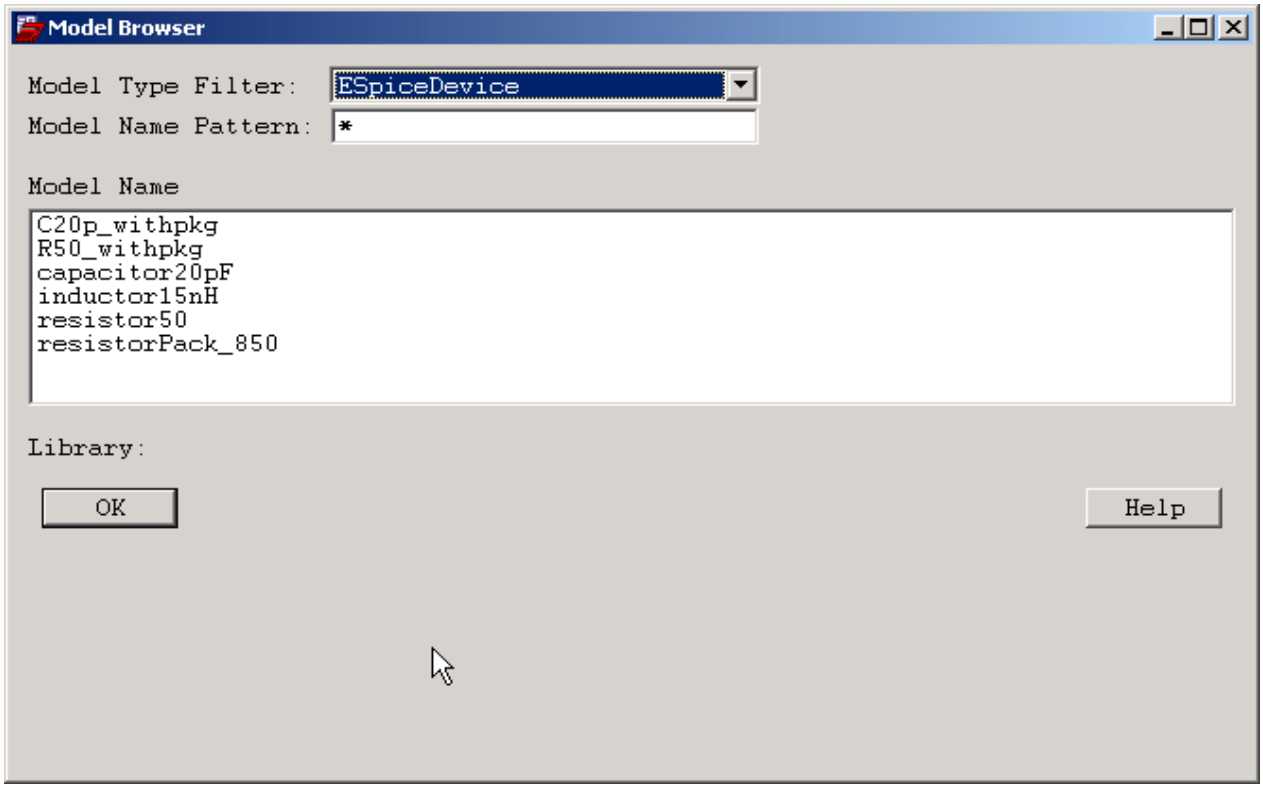


Figure 7-2 shows the selection of a user-defined multi-terminal ESpice from the Model Browser.

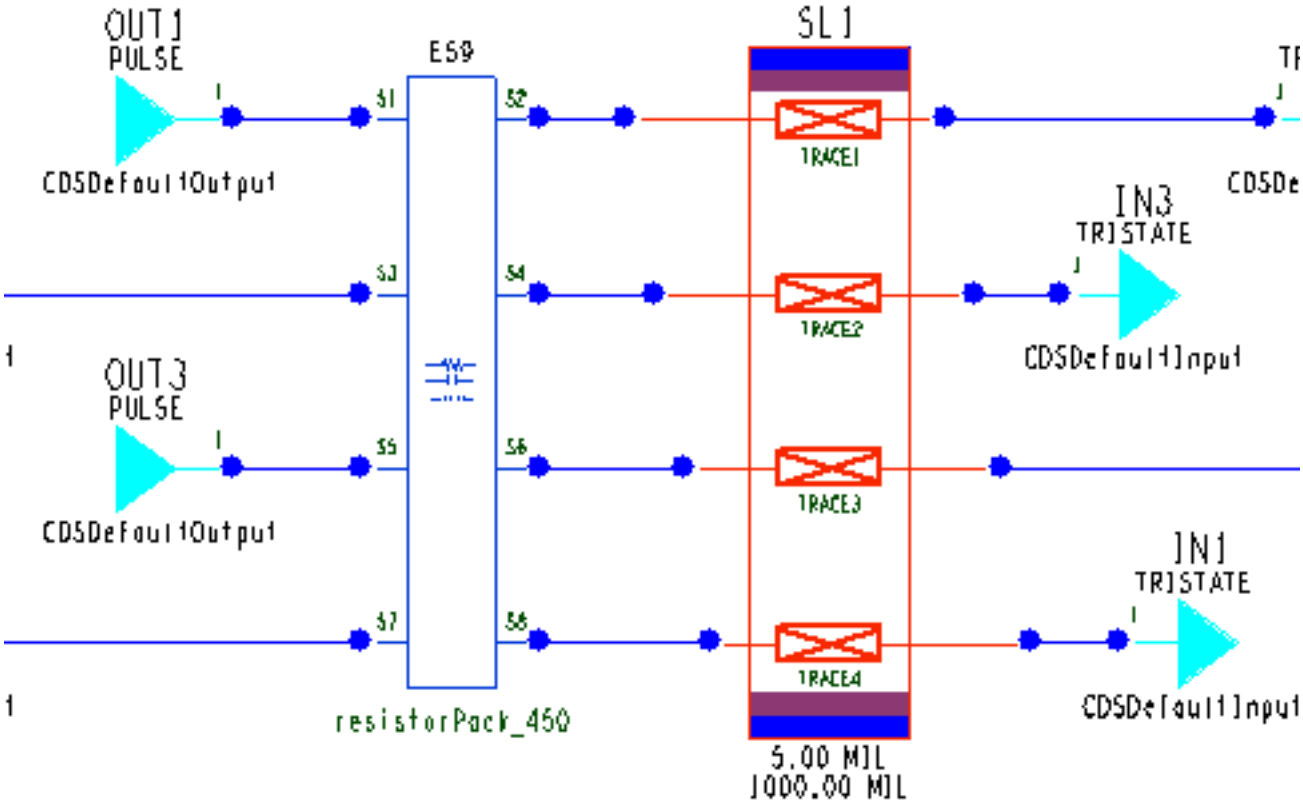
Figure 7-2 Selection of a Multi-terminal ESpice Device in the Model Browser



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 7-3 shows a user-defined multi-terminal ESpiceDevice and its use within a sample topology.

Figure 7-3 Sample Multi-terminal ESpice Topology



8

Advanced Techniques: Modifying Stimulus for an IOCell

Topics in this chapter include:

Introduction

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

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

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

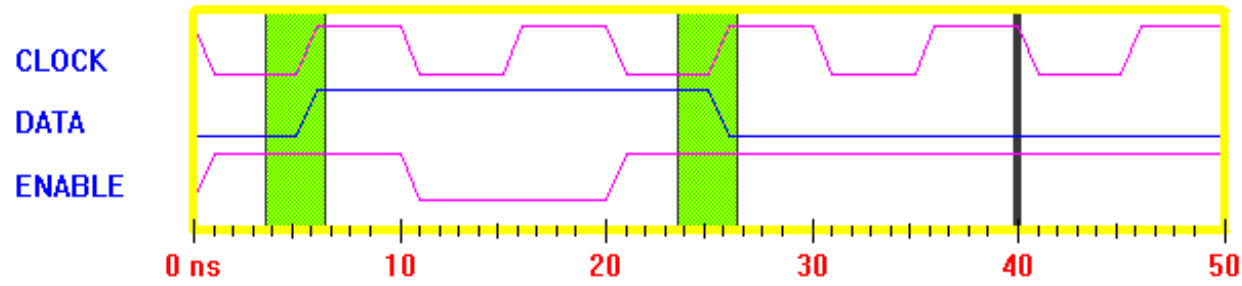
You save and recall stimulus vectors for all IOCell drivers in the topology with the *Set – Vectors* command.

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 here 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 8-1 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.

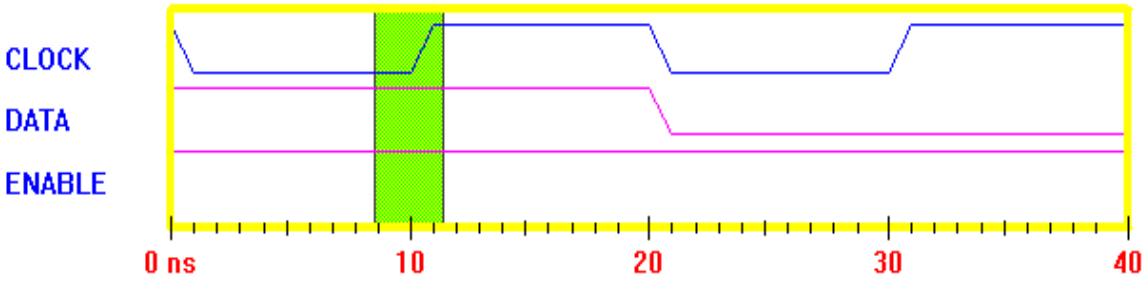
Defining Measurement Information for Custom and Tristate Stimuli

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 8-2 illustrates sample *Setup* and *Hold* margins.

Figure 8-2 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.

9

Advanced Techniques: 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.

The advantages of using S-parameters are 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:

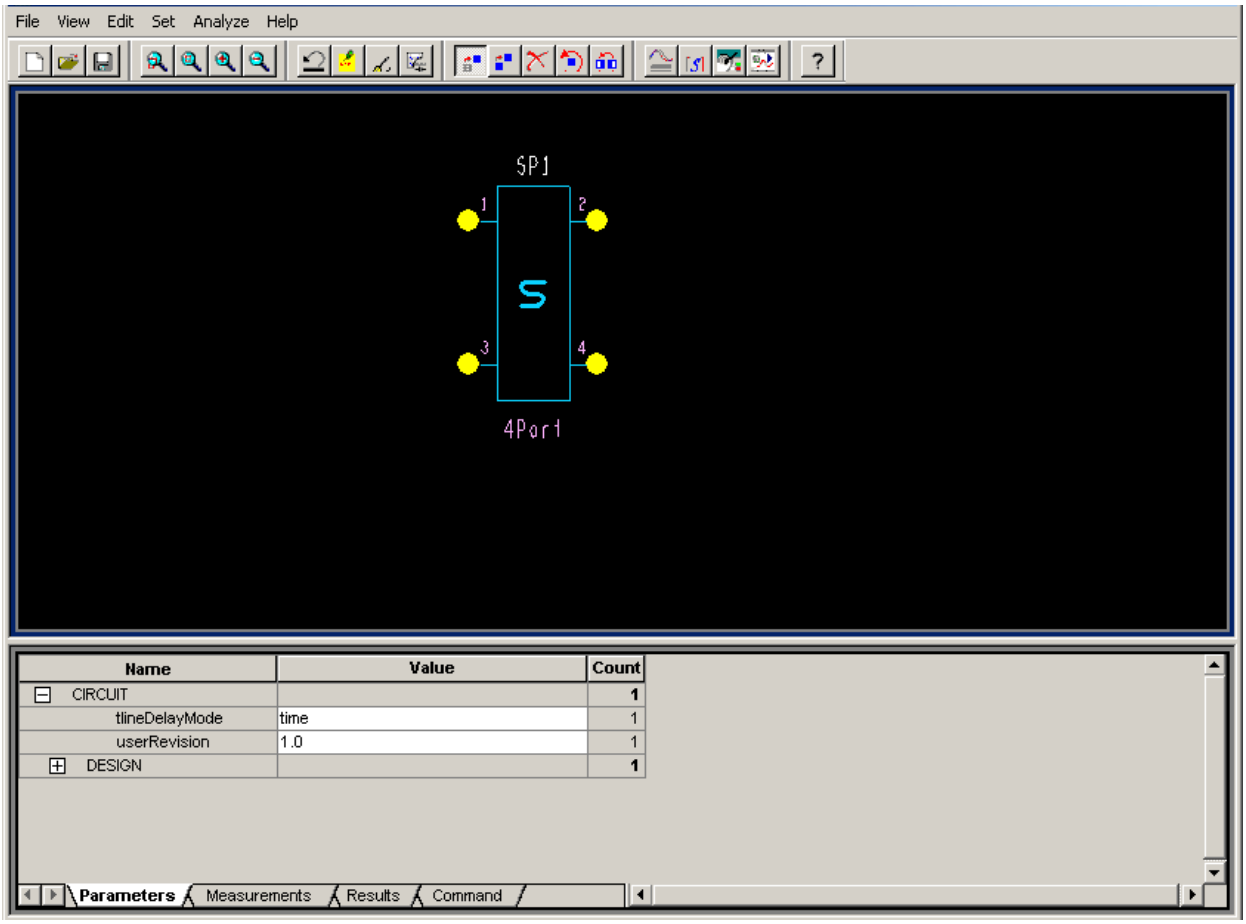
- Evaluate channel loss in the frequency domain to determine if the topology meets the loss budget.

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 125

- Create an ESpice black box model of the S-parameter data for use in Time Domain Analysis, as seen in Figure 9-2.

Figure 9-1 S-Parameter Model with 4 Ports (Maximum of 12 Ports Allowed)



Defining Ports

You can define ports for all diodes, I/OCells, 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.

In the S-Parameter Generation dialog box, use *Add* to generate ports automatically or the Port Editing dialog box to manually set ports one by one.

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.

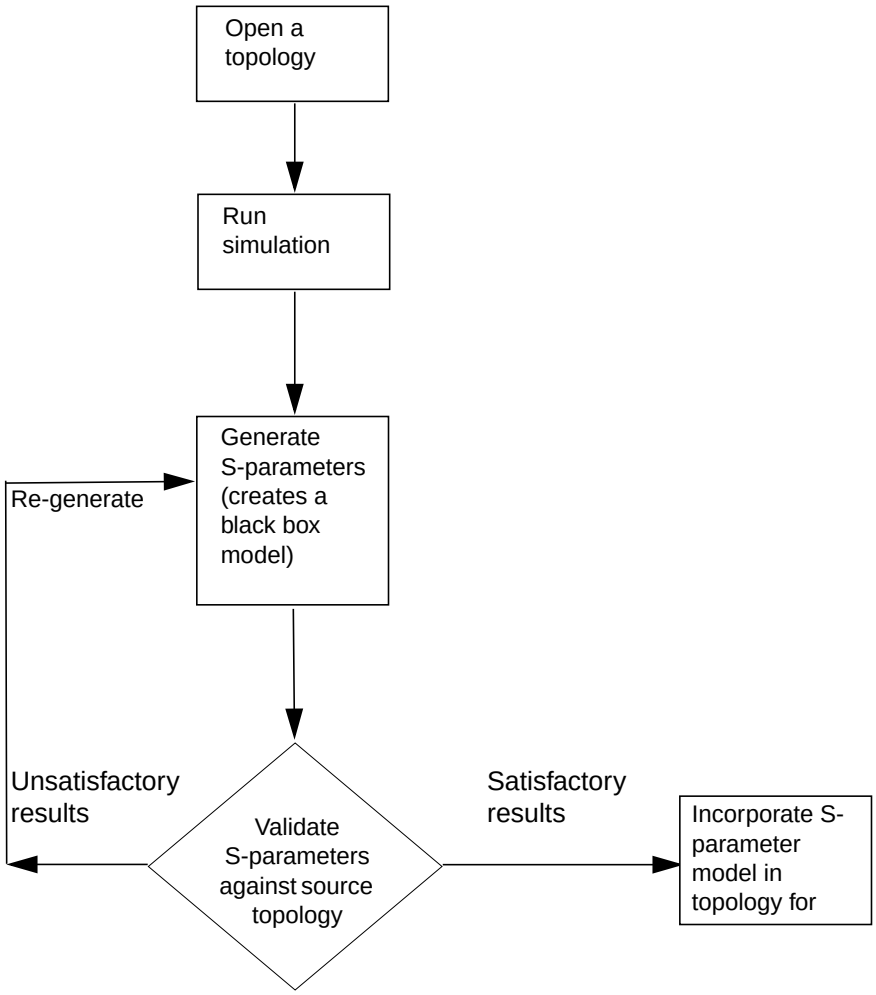
Time Domain Analysis

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 for:

- Incorporating measured data for known or fixed elements in the signal path for simulations. For example, a legacy backplane.
- Providing 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 9-2 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`.

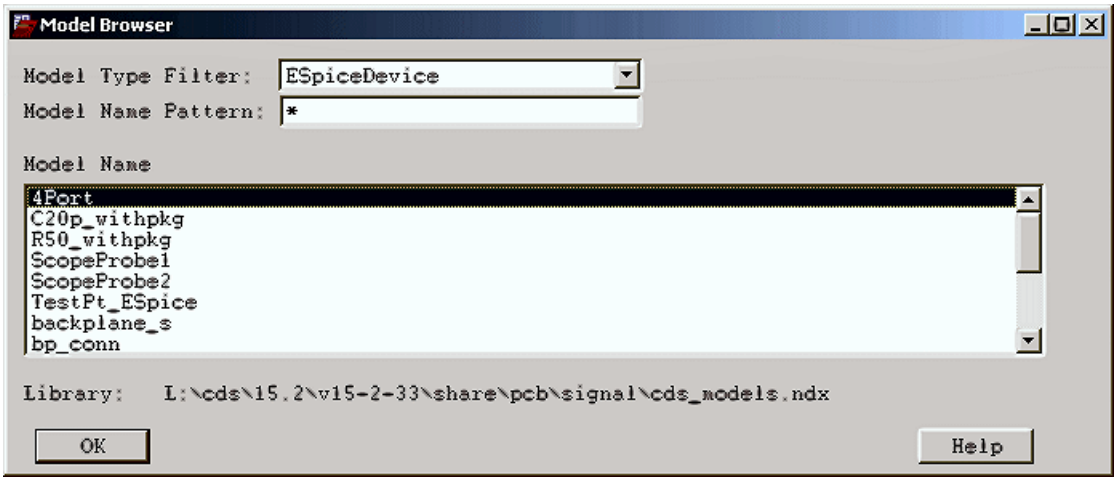
S Parameter Settings Example

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 127
- 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, as shown in Figure 9-3.

Figure 9-3 ESpice Device Model Containing 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 9-1 on page 119.

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

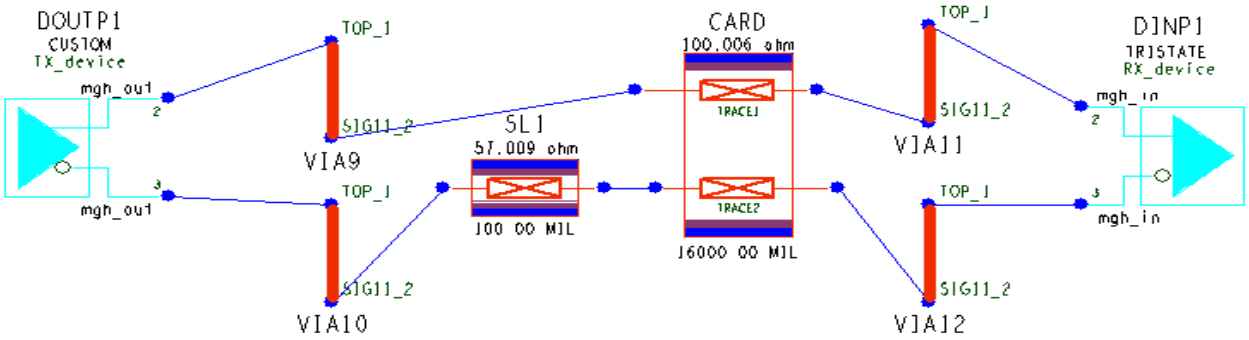
Viewing Frequency Response Using S-Parameters

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 9-4.

   The backplane and receiving card appear in detail.

Figure 9-4 Source Topology



2. Choose *Analyze – Preferences – Simulation Parameters*.
3. Set the *Waveform Resolution* to `10 ps` and the *Cutoff Frequency* to `10GHz` (*Analyze – Preferences – Simulation Parameters*).

Important

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

4. Choose *Analyze – [S] Generation*.

   The S-Parameter Generation dialog box appears.

5. Click *Add Set Port for each IOCell*.

   The ports automatically appear in the topology with names.

6. Enter the following values:

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

7. Click *Generate*.

   The S-Parameter Generation Log appears.

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

   In this example, it is S[DOUTP2][DINP2].

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

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

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

11. 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.

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

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

14. Click *Analyze – [S] Generation*.

    The S-Parameter Generation dialog box appears.

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

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

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

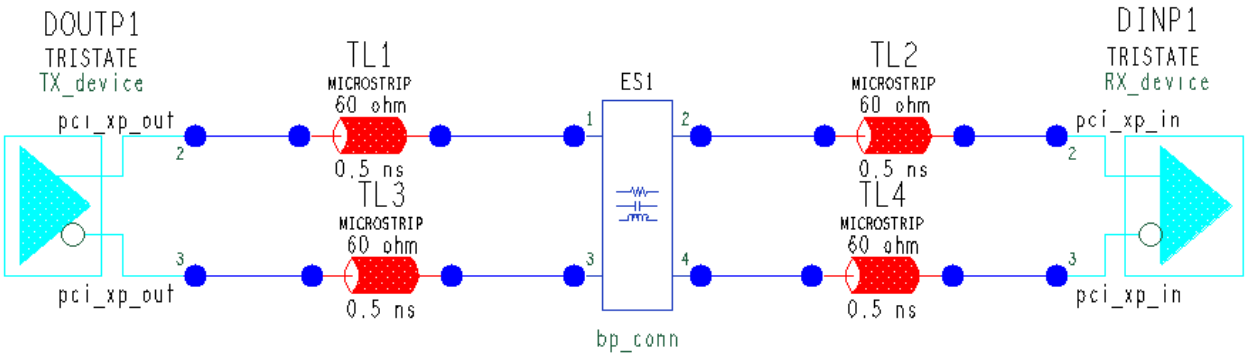
    The trace length now meets the loss budget.

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 9-5.

Figure 9-5 Topology without Generated S-Parameters



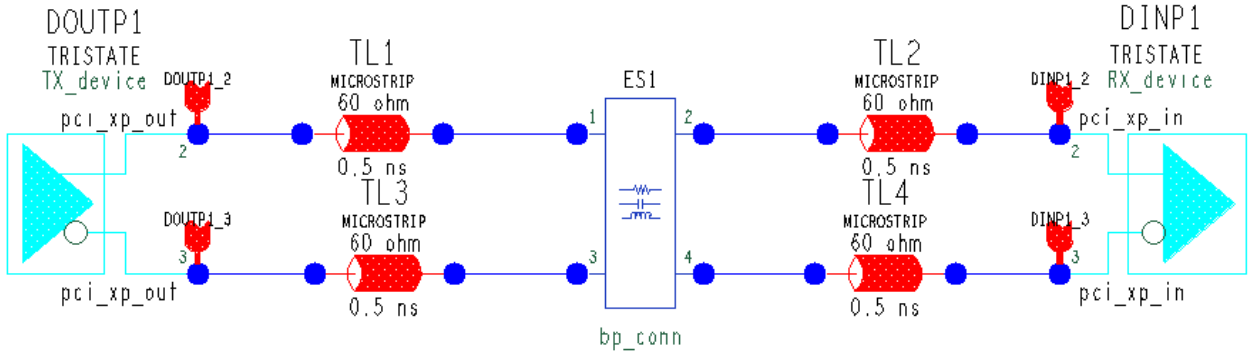
2. Choose *Analyze – [S] Generation*.

   The S-Parameter Generation dialog box appears.

3. Check *Set Port For Each IOCell.*

   The added ports appear in the canvas, as seen in Figure 9-6.

Figure 9-6 Topology with Ports on IOCells

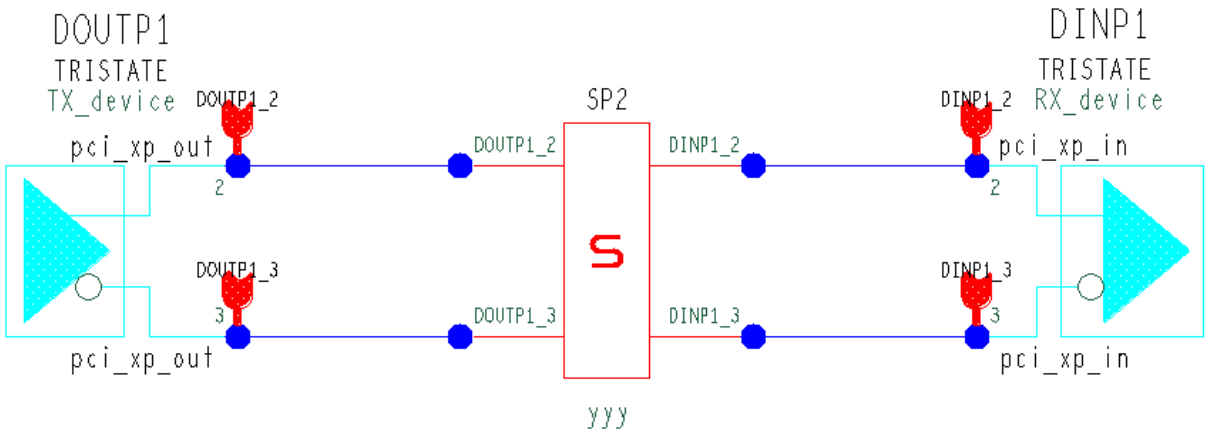


4. Check *Substitute with the Generated S-Parameter.*

5. Click *Generate*.

   The original topology updates with the generated S-parameter black box, as seen in Figure 9-7.

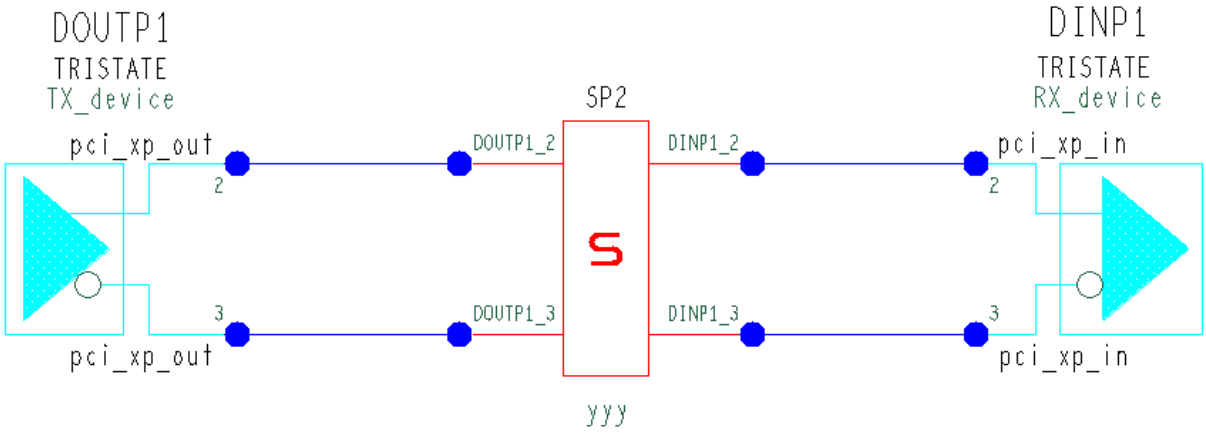
Figure 9-7 Generated S-Parameter Black Box



6. Click *Close*.

The dialog box closes and all ports disappear from the topology. See Figure 9-8. The new S-parameter model appears in the working DML library as a Touchstone file.

Figure 9-8 Updated Topology



10

Advanced Techniques: Via Modeling

Topics in this chapter include . . .

Introduction

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, and scattering parameters (S parameters); via model types include single vias and coupled vias (signal, signal-and-ground, and signal-and-power). You can easily create via models in SigXplorer, add them as parts to a topology, perform what-if simulations, and analyze the results using SigWave before committing to a PCB layout.

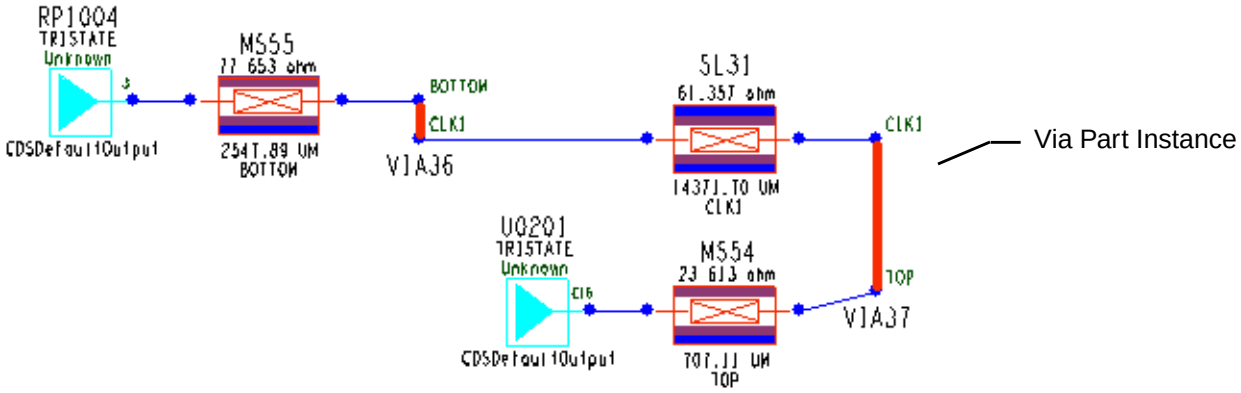
Important

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 10-1 on page 131. 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 10-1 Extracted Topology with Vias

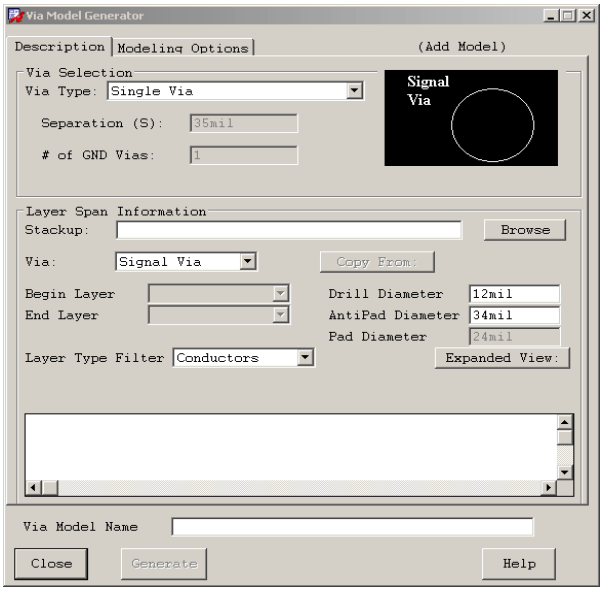


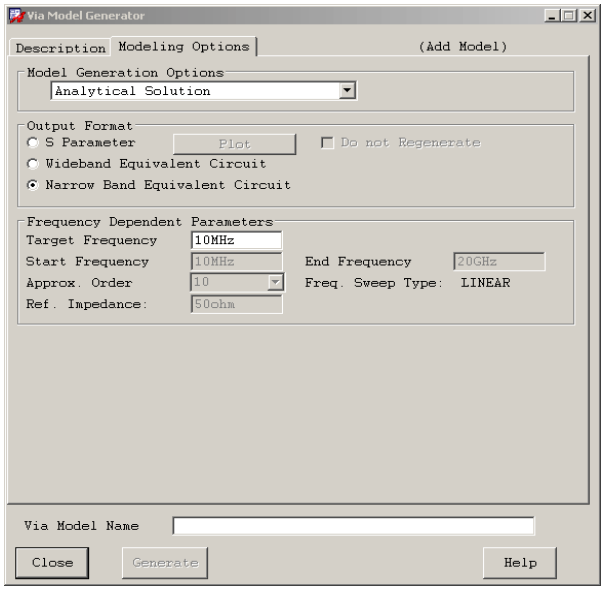
Via Model Generation

You can modify an existing via model or create one from scratch using the Via Model Generator dialog box shown in Figure 10-2.

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 10-2 Tabbed Via Model Generator Dialog Box





To access the Via Model Generator dialog box

1. Choose *Analyze – Libraries*.

   The Signal Analysis Library Browser appears.

2. In the *Interconnect Library Files* list box, select an interconnect library for the model by double-clicking on its entry.

   The Model Browser dialog box appears showing a list of existing models in the selected interconnect library.

3. Click *Add Model* and choose *Via* from the drop-down menu.

   The Via Model Generator dialog box appears.

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 10-3 Single Signal Via Symbol

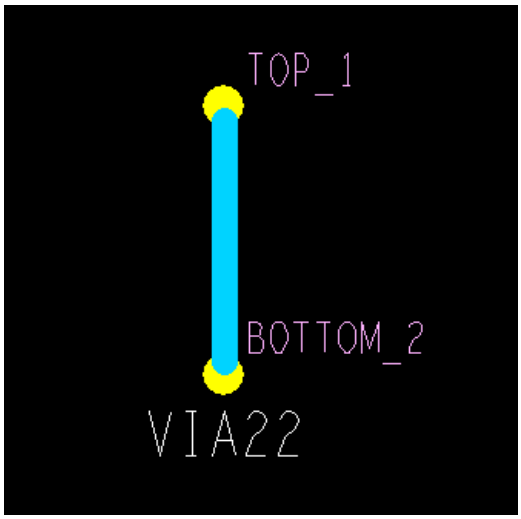
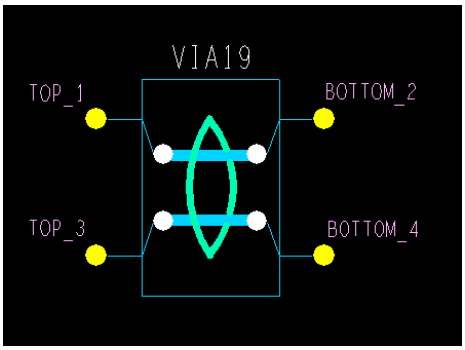
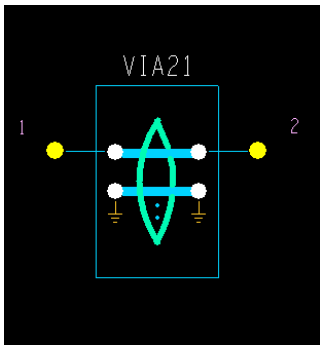


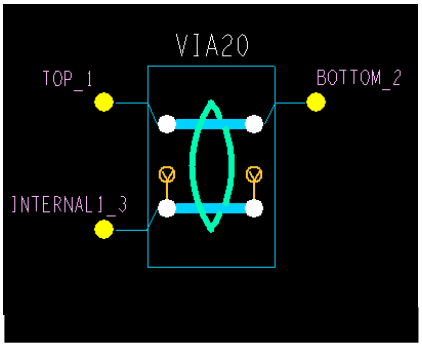
Figure 10-4 Coupled Via Symbols



Signal-and-signal Via



Ground-and-signal Via



Power-and-signal Via

A

Designing Common Clock Buses

Topics in this chapter include

Introduction

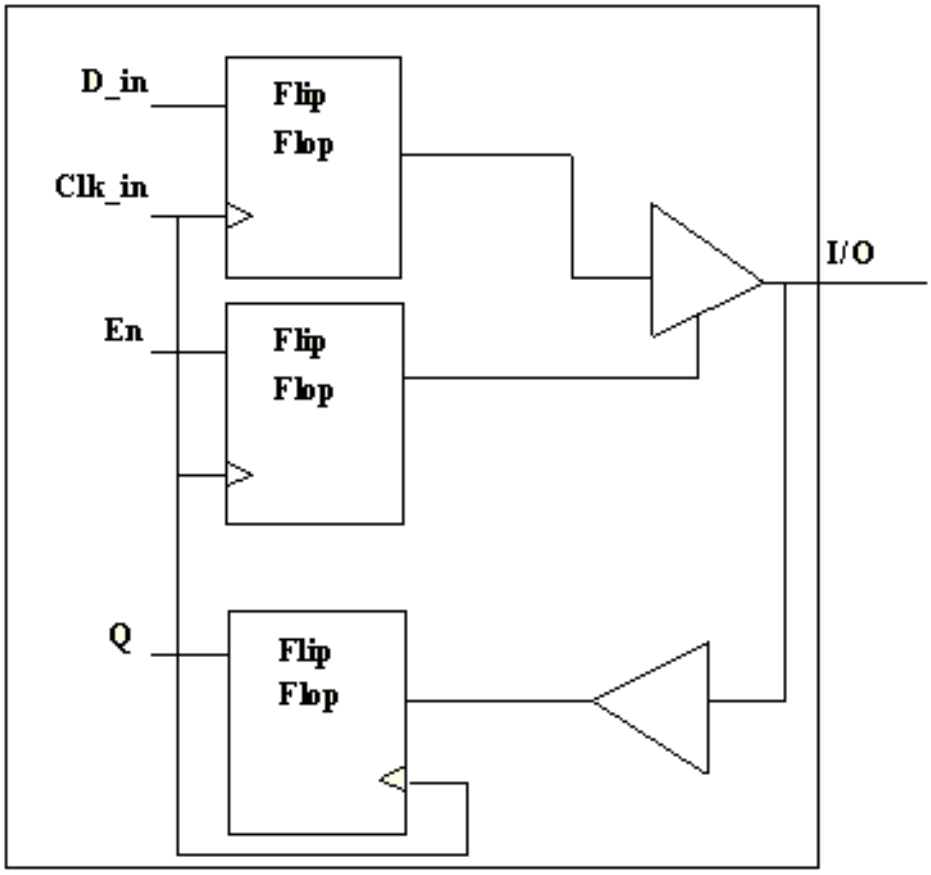
SigXplorer helps you to prototype and design high-speed bus topologies. You can create target bus topologies with multiple drops where each drop, in turn, takes and releases control of the bus based on stimuli you define for each IOCell on the bus. This allows you to see true dynamic effects over several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

The following 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 A-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.

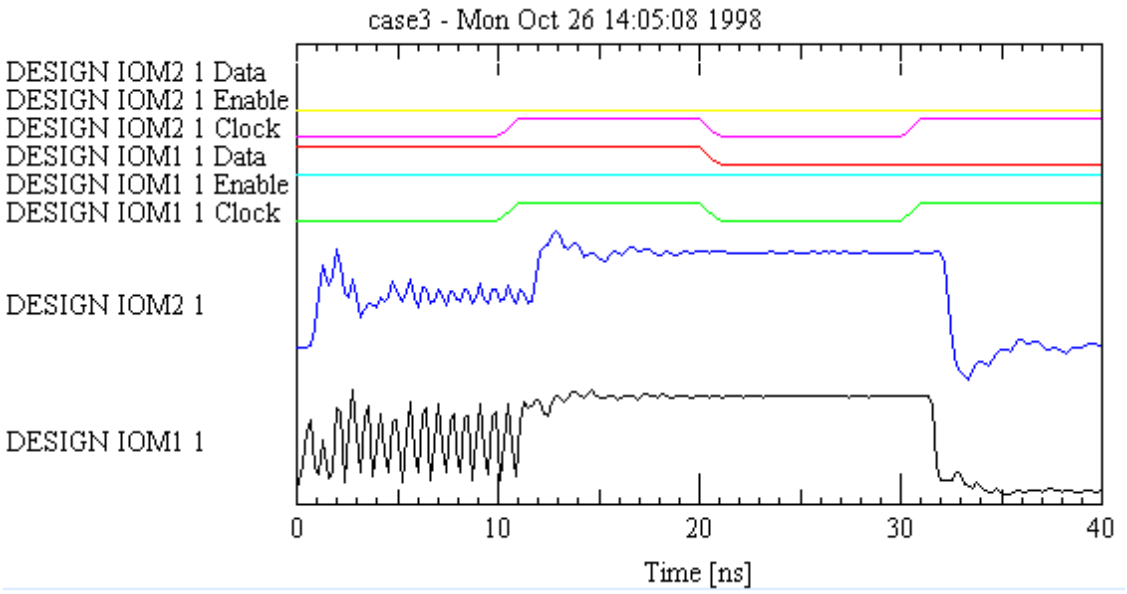
Figure A-1 Internal View of IOCell Clocked MacroModel



Timing Diagram Display in SigWave

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

Figure A-2 Timing Diagram Display in SigWave



Adding a Clocked IOCell MacroModel

1. Choose *Edit – Add Part*.

   The Model Browser appears.

2. From the pull-down menu, 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

- 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.

2. In the IOCell Stimulus Editor, make the appropriate edits to the clock, data, and enable signals.

3. Click *Apply* to apply the settings and retain the IOCell Stimulus Editor

   - or -

   Click *OK* to apply the settings and dismiss the IOCell Stimulus Editor.

To Modify the RefDes Associated with the IbisIO

1. In the canvas, click the RefDes, or part name, associated with IbisIO symbol. For example, 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. See About Modifying the Stimulus for an IOCell for more information.

3. Click Apply to apply the settings and retain the IOCell Stimulus Editor.

   - or-

   Click OK to apply the settings and dismiss the IOCell Stimulus Editor.

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.

B

Working with Coupled Traces

Topics in this chapter include

Introduction

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 Part Model 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.

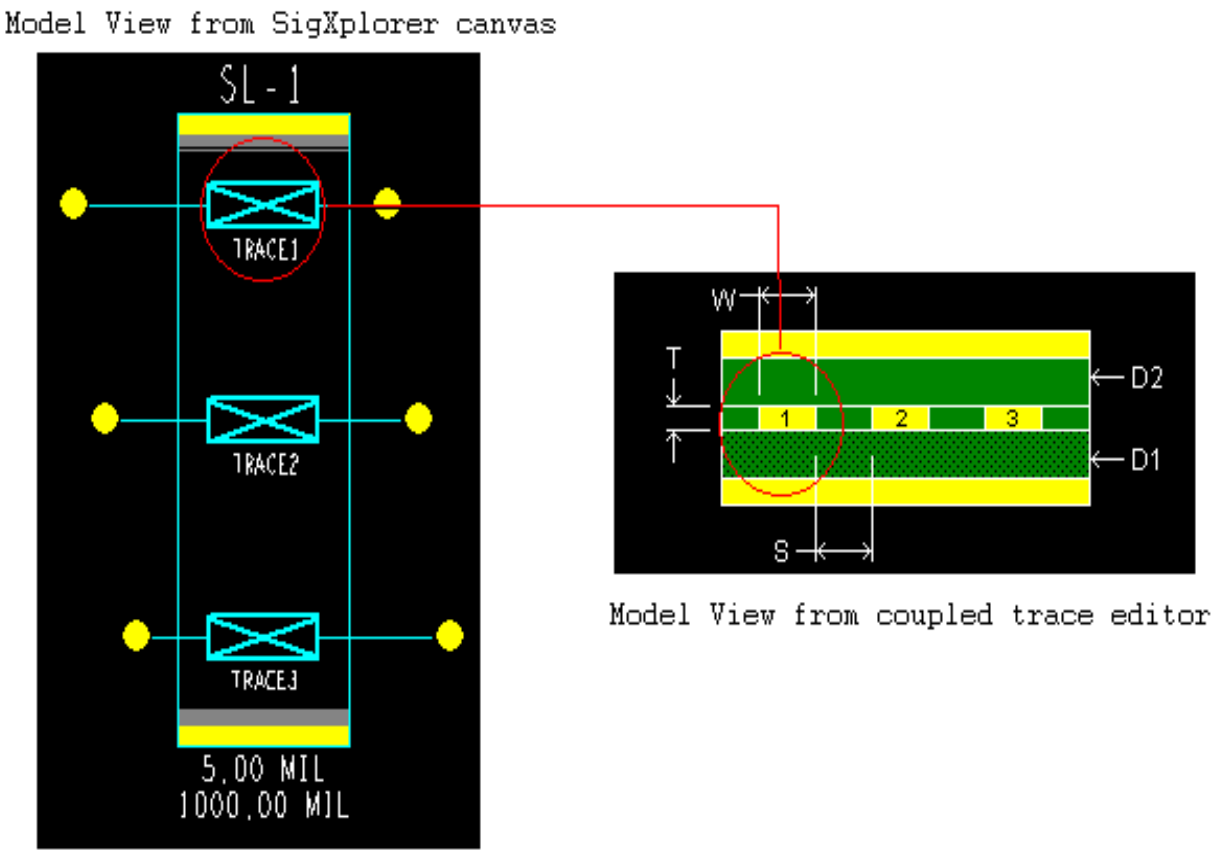
You define stimuli for victim and aggressor net IOCells using the IOCell Stimulus Editor.

Coupled Trace Symbols

Coupled trace symbols appear in the canvas as individual traces enclosed with 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 B-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).

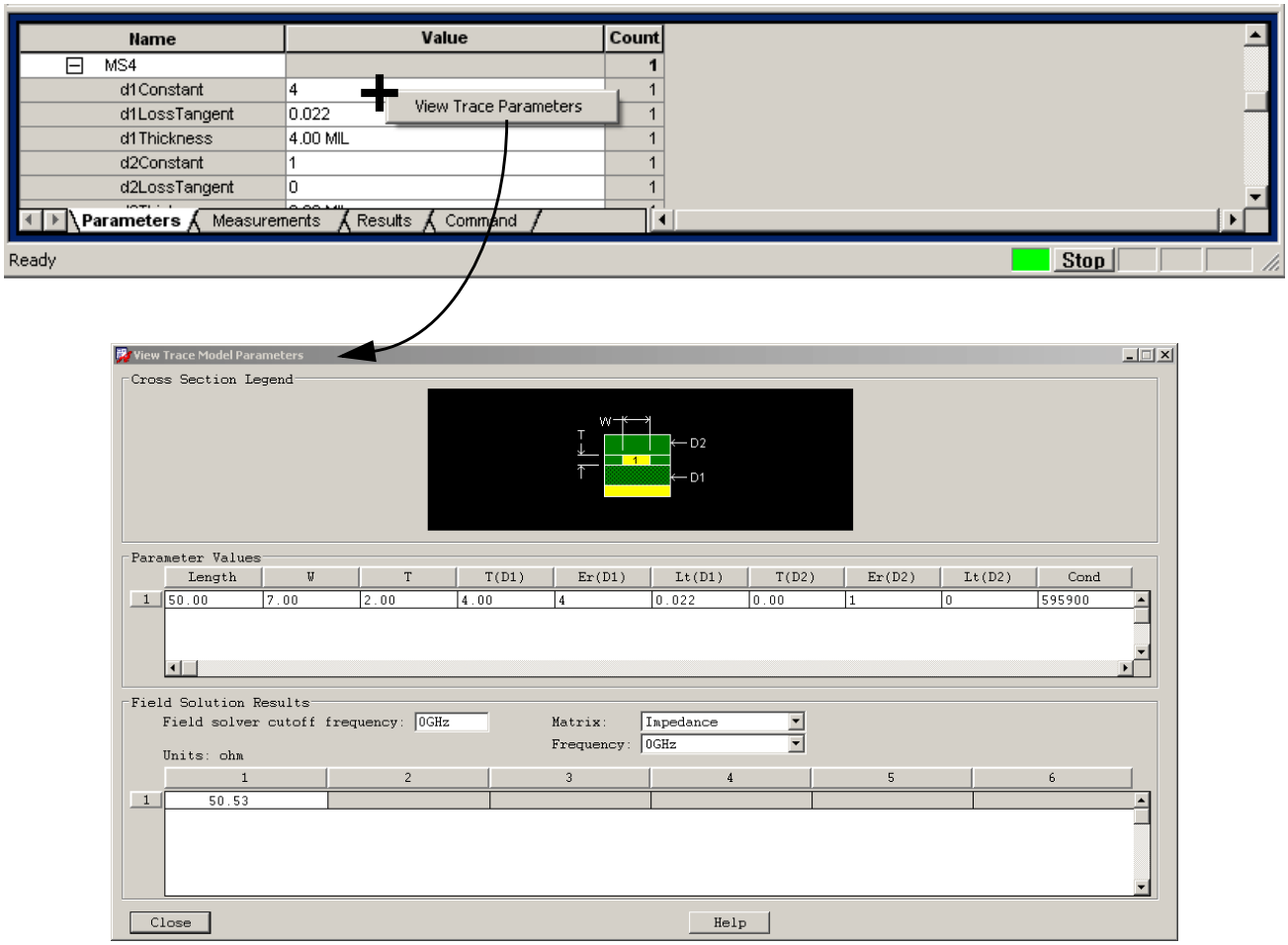
Figure B-1 Viewing Trace Model Parameters



You invoke the View Trace Model Parameters dialog box by positioning your cursor over a trace name or one of its values in the Parameters tab of the SigXplorer spreadsheet. Right-click to display the *View Trace Parameters* button, then click the button (as shown in Figure B-2). 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.

Figure B-2 Invoking the Trace Model Parameters Dialog Box



Exploring Topologies Containing Coupled Trace Models

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

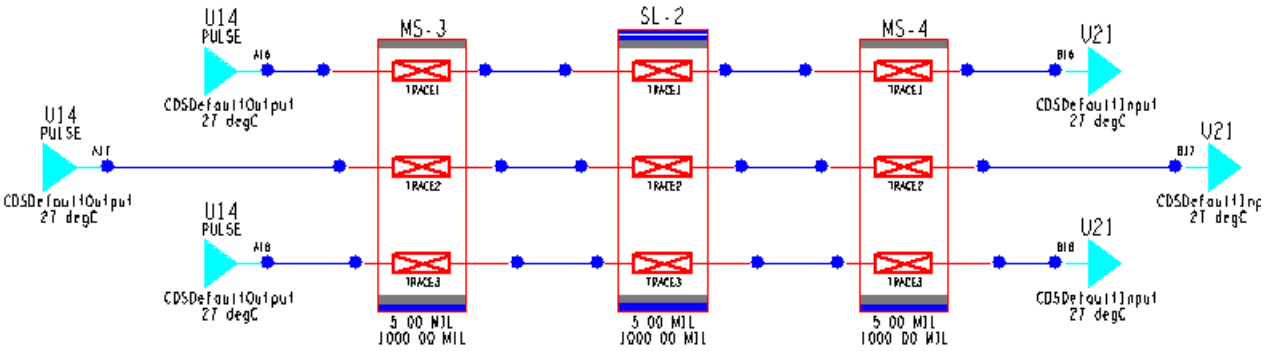
For post-part placement, you can extract a net from a routed board into the canvas, and then replace the interconnect with the appropriate coupled trace models. You can then experiment in the canvas with trace length, spacing and layer stackup parameters to derive a topology that satisfies the target 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 B-3) 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 B-3 Sample Coupled Trace Circuit



Adding a Coupled Trace Model

To add a coupled trace model

1. Choose *Edit – Add Part*.

   The Model browser appears.

2. From the pull-down menu, choose Interconnect.
3. Select the desired coupled trace model and drag it to the canvas for placement.
4. Click *OK* to dismiss the Model 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 the 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. Enter the new RefDes and click *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 (icon fig) 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, enter the new values.

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

   b) To enter multiple values for simulation sweeping:

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

- Use *Multiple Values* to enter a list of values by specifying a list of discrete number values.
- Use *Expression* to enter an expression by specifying 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 Editor 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 necessary, 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 (*Set – Constraints*) in your target topology and subsequently apply the topology (*File – Update*) in Allegro SI to

each bit of a bus. You can also apply the topology to the Allegro SI or Allegro PCB Editor design using the Allegro SI *Toplogy – Template* command.

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.

C

Working with Differential Pairs

Topics in this chapter include

Differential Signaling

Differential Signaling uses a pair of nets (instead of a single net) with opposite current and voltage swings to convey data. The advantage of the differential approach is that noise is coupled onto the two nets as common mode noise (the noise appears on both nets equally) and is therefore rejected by the receivers that look at the difference between the two signals. The differential signals also radiate less noise than single-ended signals due to the canceling of electro-magnetic fields. For more information, see *Best Practices: Working with Differential Pairs*.

SigXplorer helps in determining the optimum differential impedance and differential propagation delays in exploration, pre-placement, and post-route modes.

In addition to powerful exploration and analysis features, SigXplorer provides the following functionality for developing and analyzing electrical diff pair topologies:

- Differential buffer models

  See Differential buffer pin and coupled transmission line parameter on page 159.

- Coupled transmission lines

  See Differential buffer pin and coupled transmission line parameter on page 159

- 1024-bit custom stimulus for inverting and non-inverting drivers

  See Custom Stimulus on page 160.

- Constrained custom measurements at either the package-pin or die-pad

  See Constrained Custom Measurements on page 168.

- Differential pin parameters

  See Differential pin parameters on page 170

- Differential buffer delays

  See Differential Buffer Delays Test Fixture on page 172.

- Differential constraints

  See Capturing Diff Pair Constraints on page 164.

Types of Diff Pairs

You specify Diff Pairs in the following ways:

- Model-defined Diff Pair (also known as electrical diff pairs)

  You specify model-defined diff pairs in a device model. In the device model, you designate inverting and non-inverting signals of the diff pair. The device's signal model lets you uniquely characterize the diff pair by specifying pin parasitics, launch delays, logic thresholds, and buffer delays. We recommend model-defined diff pairs for the high-speed flow.

- User-defined Diff Pair

  You specify user-defined diff pairs in Constraint Manager. This affords you more flexibility in renaming diff pair objects and changing diff pair membership, but you forgo the accuracy of model-defined diff pairs. See the Constraint Manager User Guide for more information on user-defined diff pairs.

This chapter focuses on model-defined diff pairs.

To create a model-defined diff pair

1. In SigXplorer (PCB Editor, PCB Editor - APD, or Allegro SI), choose *Analyze – Libraries*.

   The Signal Analysis Library Browser appears.

2. Select a device library.

3. Click *Browse Models*.

   The Model Browser appears.

4. Click the *IbisDeviceModel*.

5. Click *Edit*.

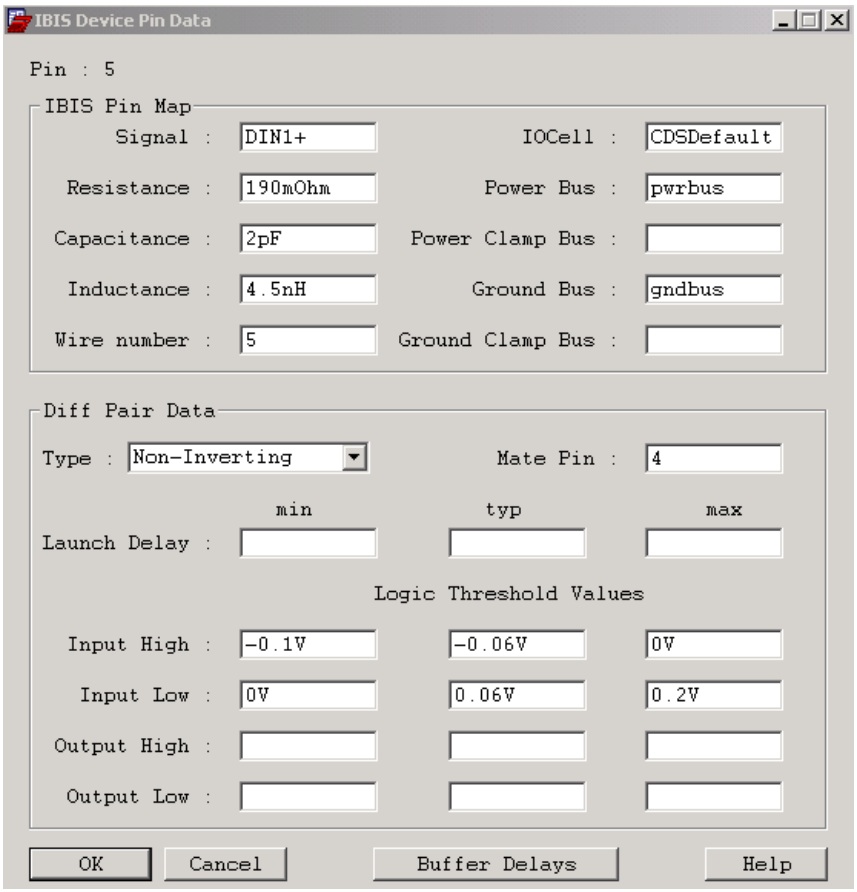
   The IBIS Device Model Editor appears.

6. Select a pin.

   Note: Your component manufacturer should specify which pins to route differentially.

   The IBIS Device Pin Data dialog box appears for the selected pin.

Figure C-1 IBIS Device Pin Data dialog box



7. From the Differential Pair Data Type drop-down menu, choose *Inverting* or *Non-inverting*.
8. Specify a mate pin.

   This pin will inherit a differential data type opposite than the type chosen for the seed pin (Step 7).
9. Specify launch delays and logic thresholds.
10. Click *OK*.

High-speed Electrical Diff Pair Flow

You work with diff pair nets in the high-speed PCB flow as outlined in the Exploration Flow (below) in the Verification Flow on page 162.

Exploration Flow

In SigXplorer . . .

To add parts

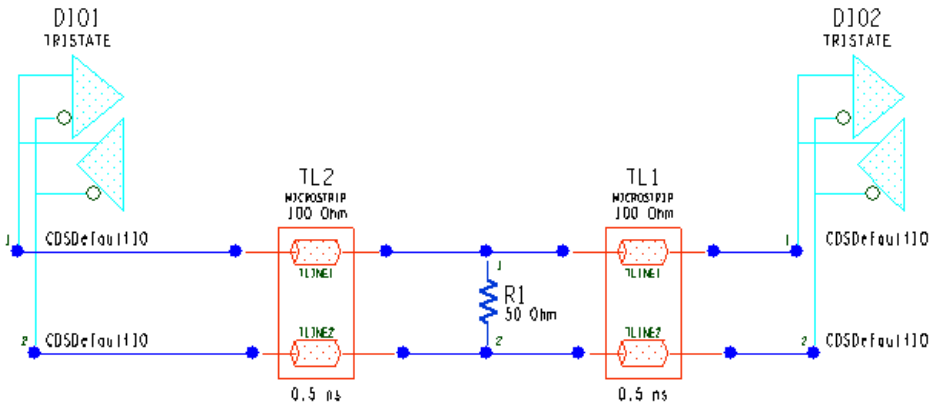
1. In SigXplorer, choose *Edit – Add Part*.

   The Model Browser appears.

2. Choose *DifferentialPart* from the Model Type filter drop-down menu.
3. Drag the part (DiffIO, DiffInput, or DiffOutput) to the canvas and click to place it.
4. Click again to place another instance, then right-click and choose *End Add* from the pop-up menu.
5. From the Model Browser choose *ESpiceDevice* from the Model Type filter drop-down menu.
6. Drag a resistor to the canvas, then right-click and choose *End Add* from the pop-up menu.
7. From the Model Browser choose *Interconnect* from the Model Type filter drop-down menu.
8. Drag *TlineCoupled* to the canvas, then right-click and choose *End Add* from the pop-up menu.
9. Click *OK* to dismiss the Model Browser.
10. Wire the circuit.

    Your topology should resemble Figure C-2 on page 158.

Figure C-2 Coupled diff pair topology



To replace the default CDSDefaultIO model

1. In the SigXplorer canvas, select the *CDSDefaultIO* label on the diff pair element.

   SigXplorer highlights the element in the Parameters tab below the canvas. You may have to resize the canvas pane.

2. Click in the *Value* cell in the *bufferModelNonInverting* field.

   A down-pointing arrow appears.

3. Click the down-pointing arrow.

   The Buffer Parameters dialog box appears.

   You may have to add the model to one of the libraries.

4. Click > to move your model to the *selected models* list on the right.

5. Repeat Steps 2 - 4 for the bufferModelInverting field.

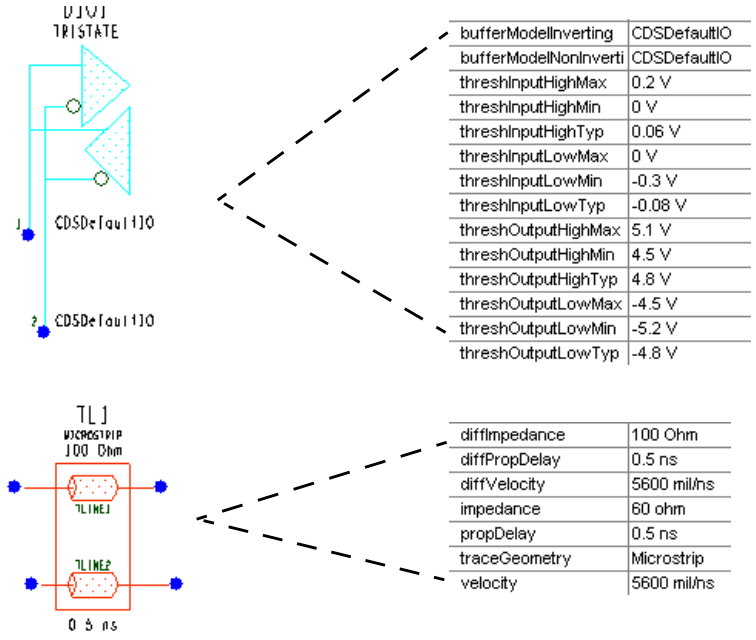
6. Click *OK* to accept the changes and dismiss the dialog box.

To edit diff pair parameters

1. On the SigXplorer canvas, select a diff pair element or a transmission line element.
2. Click the parameters tab.

   You may have to resize the canvas pane.
3. Experiment by changing parameters for the buffer model and the transmission line.

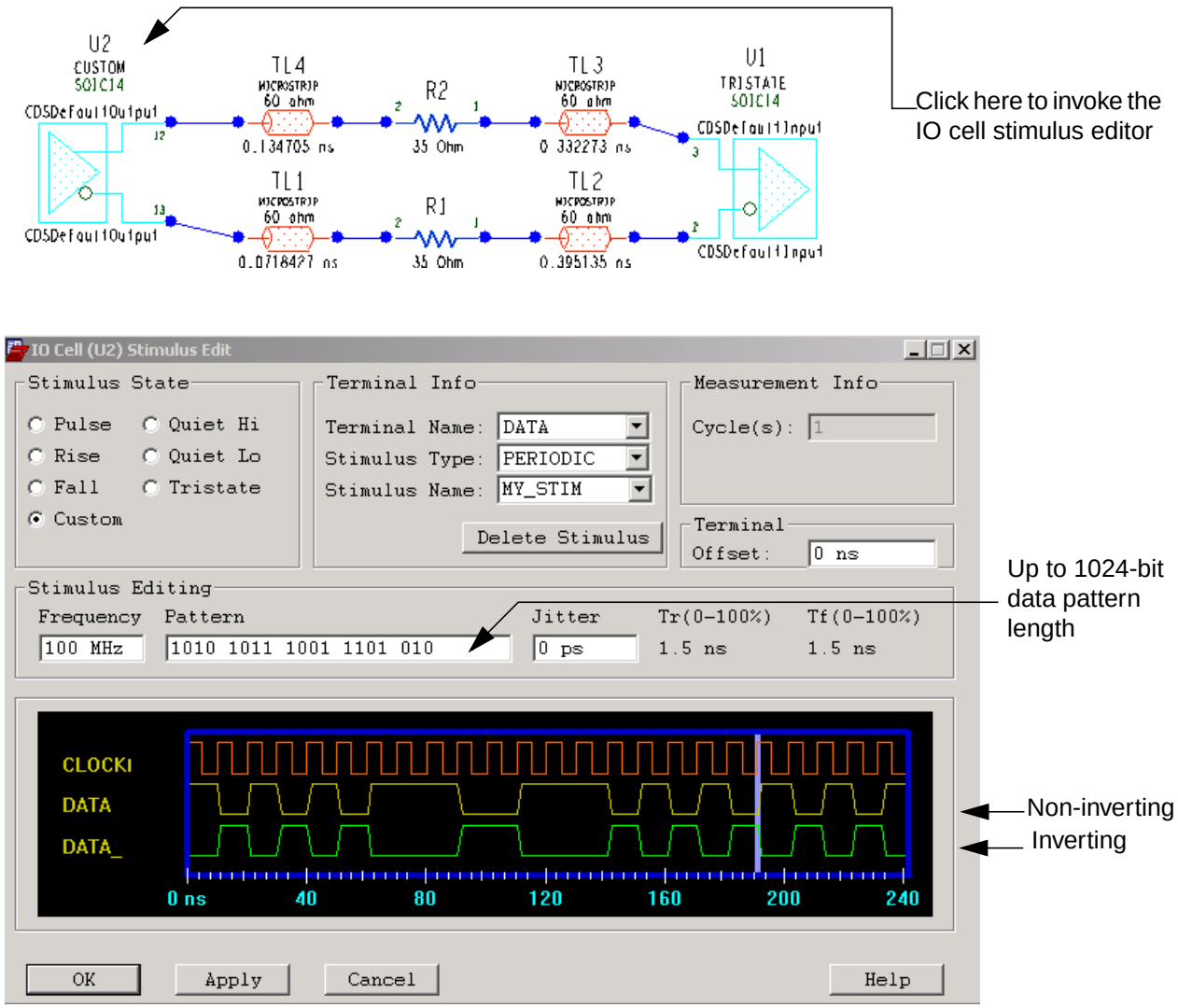
Figure C-3 Differential buffer pin and coupled transmission line parameter
r



4. Optionally, to specify differential pin parameters, see Constrained Custom Measurements on page 168.
5. Optionally, to specify differential buffer delays, see Differential Buffer Delays Test Fixture on page 172.
6. Specify a default or custom stimulus for the differential driver.

   You can specify a data pattern of up to 1024-bits. SigXplorer applies opposing stimulus patterns for the inverting and non-inverting differential driver pins.

Figure C-4 Custom Stimulus



7. Choose *Analyze – Simulate* to sweep the topology parameters.
8. Find the cornercases and re-edit and re-sweep the topology parameters if necessary.

To implement constraint requirements

1. Choose *Set – Constraints*.
2. The Set Topology Constraints dialog box appears.
3. Click the Diff Pair tab.

4. Enter your target constraints resulting from the topology exploration. (See Capturing Diff Pair Constraints on page 164.)
5. In the Set Constraints dialog box, click another tab and enter other constraints as needed.
6. Choose *File – Save As* to save the constraints to a topology file.

In Constraint Manager . . .

1. From the layout tool, choose *Setup – Electrical Constraint Spreadsheet*.

   Constraint Manager launches with a view of the board database.
2. Choose *File – Import – Electrical CSet*.

   The Import Electrical CSets dialog box appears.
3. Click on the diff pair topology you saved in SigXplorer.
4. Click *Open*.

   Constraint Manager imports the topology as an ECSet.
5. In the objects column, select a candidate diff pair net (inverting or non-inverting).
6. Choose *Objects – Electrical ECSet References*.

   The Electrical ECSet References dialog box appears.
7. Choose the ECSet (named after the imported topology file) from the drop-down menu.
8. Click *OK*.

   Constraint Manager applies the constraint set to both members of the diff pair in the layout tool.

Tip

Before you can analyze a diff pair circuit with custom stimulus and custom measurements, in Constraint Manager you must enable custom stimulus (choose *Analyze – Settings*), and you have to enable diff pair design rule checking and custom measurements (choose *Analyze – Analysis Modes*).

Verification Flow

The verification flow can be either pre-placement (ratsnest) or post-route (etch). It is assumed that diff pair objects exist in the design and Constraint Manager is launched from the layout tool. SigXplorer can then be used for verification.

In Constraint Manager . . .

1. In the Objects column, select a diff pair object.
2. Right-click and choose SigXplorer from the pop-up menu.

   SigXplorer launches with both members of the selected diff pair object appearing on the canvas.

   Note: SigXplorer observes the preferences that you specify in the layout tool before performing the extraction. Click the *InterconnectModels* tab in the Analysis Preferences dialog box (Choose *Analyze – SI/EMISim – Preferences* in the layout tool to set these preferences).

   - If you enable *Do Topology Simplification*, SigXplorer extracts the diff pair net using an accumulative mechanism to simplify its topology while important information such as length and delay is kept intact. This simplification produces simulation results nearly identical to the original topology while avoiding overly pessimistic impedance mismatch from bendovers.
   - If you enable *Differential Extraction Mode*, SigXplorer extracts both members of the diff pair object with full coupling effects.

   Additionally, you can specify whether to extract ideal transmission lines or routed interconnect (choose *Tools – Options* in Constraint Manager). You cannot apply changes back to a net in the design if your topology includes routed interconnect.

In SigXplorer . . .

1. As required, perform the following procedures to verify and make changes to the topology:
   - To edit diff pair parameters on page 159
   - To add diff pair constraints on page 164

- To specify a constrained custom measurement on page 168
- To specify differential pin parameters on page 169
- To specify differential buffer delays on page 172

2. Choose *File – Update Topology*.

Constraint Manager prompts you to accept the changes to the topology and it refreshes the ECSet with the updated parameters. In turn, Constraint Manager updates diff pairs in the layout based on previous established associations with ECSets.

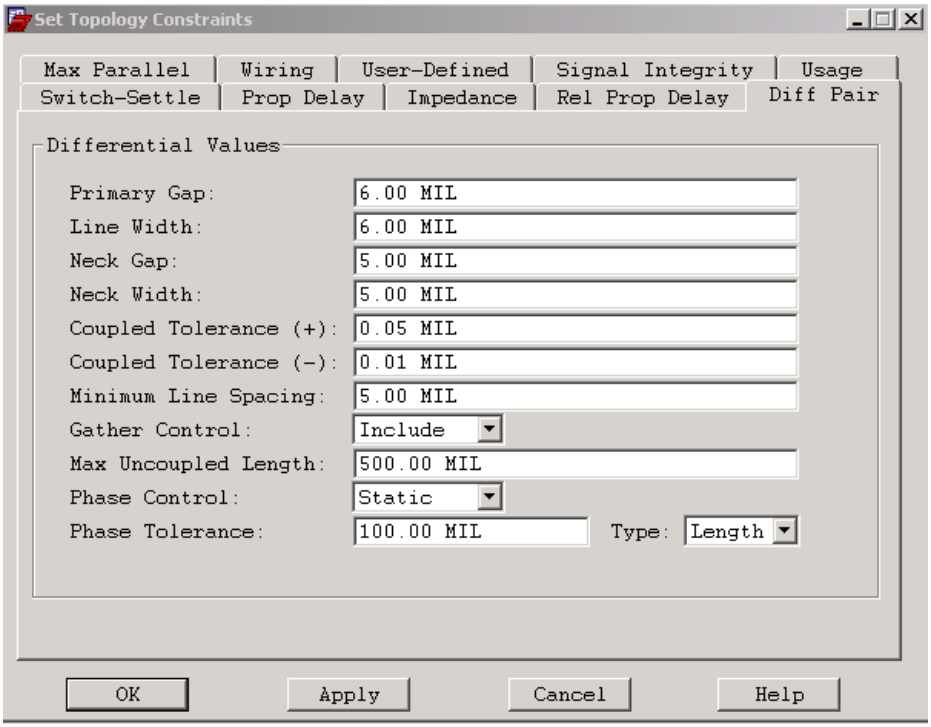
Capturing Diff Pair Constraints

Once you develop a diff pair circuit and sweep transmission line and buffer parameters, you need to capture your constraint requirements in a topology file.

To add diff pair constraints

1. In SigXplorer, choose *Set – Constraints*.

   The Set Topology Constraints dialog box appears.



2. Click the *Diff Pairs* tab.
3. Complete the fields as described in Table C-1 on page 165.

Table C-1 Capturing Diff Pair Constraints

Field	Value
■ Primary Gap	Enter a value for the ideal edge-to-edge spacing between the pair that should be maintained for the entire length of the pair. Values you set for Neck Gap override Primary Gap values in areas that need smaller gaps to get through dense components.
■ Line Width	Enter a value for the minimum width of each member of the diff pair.
■ Neck Gap	Enter a value for the edge-to-edge spacing between a pair as it goes through tight areas full of component pins and vias. Neck Gap overrides any value in the Primary Gap when the differential pair's spacing collapses to or below the value of the *Neck Width*. ■ Ensure that the neck gap does not go below any *Minimum line spacing* value you have set. ■ You do not need to define a neck gap if you set *(-) Tolerance* with a value that accounts for the needed neck gap.
■ Neck Width	Enter a value for the width of each line in a diff pair as it goes through confined areas among densely placed components.
■ Coupled Tolerance (+)	*Enter a (+) Tolerance* value to define a band around the primary gap in which the lines of a pair can go beyond the primary gap value. **Note:** The lines are considered coupled when they are within the band specified by the *(+) Tolerance* and outside the band specified by the *(-) Tolerance*.
■ Coupled Tolerance (-)	*Enter a (-) Tolerance* value to define a band around the primary gap in which the lines of a pair can go closer than the the primary gap value. **Note:** The lines are considered coupled when they are within the band specified by the *(+) Tolerance* and outside the band specified by the *(-) Tolerance*.

- Minimum Line Spacing — Enter a value to constrain the distance between any two segments from each Xnet member of the diff pair. The value you enter must be less than or equal to the *separation,* minus the negative tolerance. The value must also be greater than or equal to the *neck gap* value.
- Gather Control — Indicates whether the line segments that diverge and converge, as the pair of nets go from driver to receiver, should be included (*include*) or excluded (*ignore*) from the uncoupled length.
- Max uncoupled length — Enter the maximum allowable uncoupled length.
- Phase Control — Choose *Static* from the pull-down menu to enable static phase control. This option checks the differential pair for mismatched length tolerance only on the overall lengths. To circumvent this setting, set the field to <*clear*> or null (blank).
- Phase Tolerance — Enter a tolerance value (in *time* or *length)* to specify a separation to which phase matching is maintained.

Tip

Electrical diff pairs consist of driver/receiver pairs. As such, you should specify the following component class and pinuse information for parts to ensure accurate simulation results.

For this part . . .	specify this component class . . .	and this pinuse type . . .
Capacitor	DISCRETE	UNSPEC
Resistor	DISCRETE	UNSPEC
Inductor	DISCRETE	UNSPEC
Connector	IO	UNSPEC
Chip	IC	BI, OUT, IN, POWER, GROUND, UNSPEC

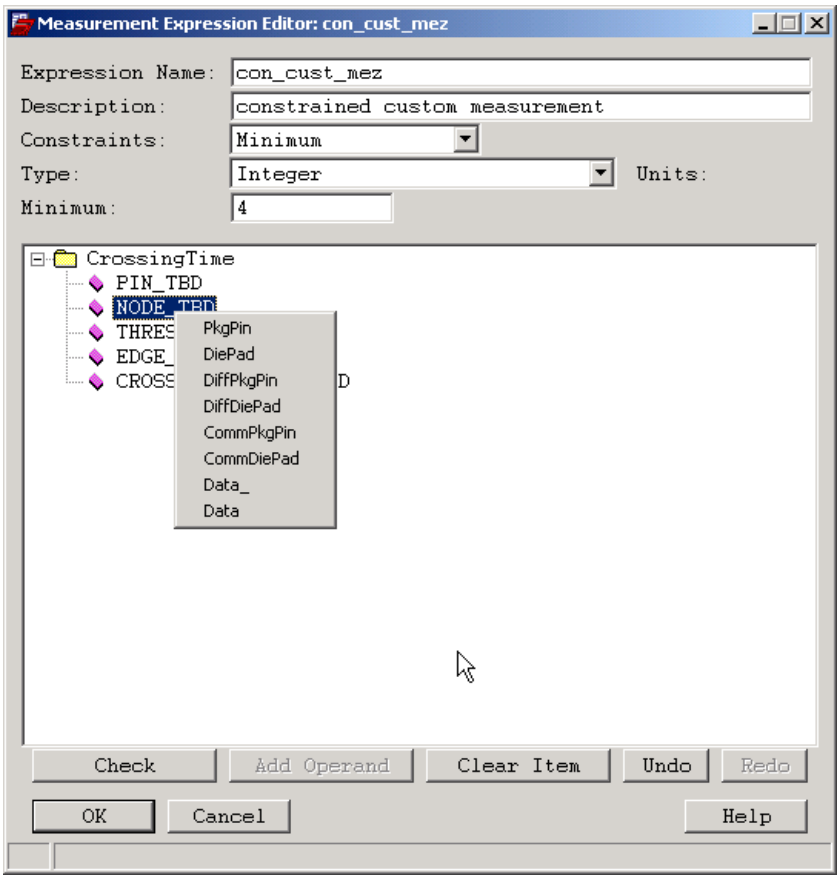
Constrained Custom Measurements

Custom measurements functionality has been extended to capture differential pin parameters such as thresholds and buffer delays. This form of measurement is called a constrained custom measurement. Furthermore, you can constrain at either the package pin or at the die pad.

To specify a constrained custom measurement

1. Click the Measurements tab.
2. Expand the *Custom* column (tree).
3. Right-click and choose *Add* from the pop-up menu.

   The Measurement Expression Editor appears.



Differential package (DiffPkgPin) and die pad (DiffDiePad) options

4. Enter an expression name.

5. Enter a description.
6. Choose from the following constraint (or measurement) types from the Constraint drop-down menu.
   - ❑ Minimum
   - ❑ Maximum
   - ❑ Min:Max
   - ❑ Target:Tolerance

     Note: Tolerance is absolute, not a percentage.

Important

When you choose *None*, SigXplorer creates a custom measurement; when you choose an option other than None, SigXplorer creates a *constrained* custom measurement. A constrained custom measurement is, in effect, a constraint that can be validated through simulation.

7. Choose a unit from the *Type* drop-down menu.

   SigXplorer adds the appropriate unit type label to the measurement.

   Note: You can change the default unit for some types (choose *Set – Defaults*, and click the *Units* tab).
8. Specify a value or a range or values for the constraint specified in Step 6.
9. Select the VALUE_TBD label and build your custom measurement expression.
10. Click *Check* to verify the proper syntax.
11. Click *OK*.

To specify differential pin parameters

1. Complete Steps 1 - 8 of To specify a constrained custom measurement on page 168.
2. Develop an expression that captures a differential requirement, such as CrossingTime.

3. On a node that supports pin parameters, right-click and choose *PinParameter* from the pop-up menu.
4. On the PIN_TBD node, right-click and choose a pin from the list.
5. On the PARAMETER_TBD node, right-click and choose from the following differential pin parameters:

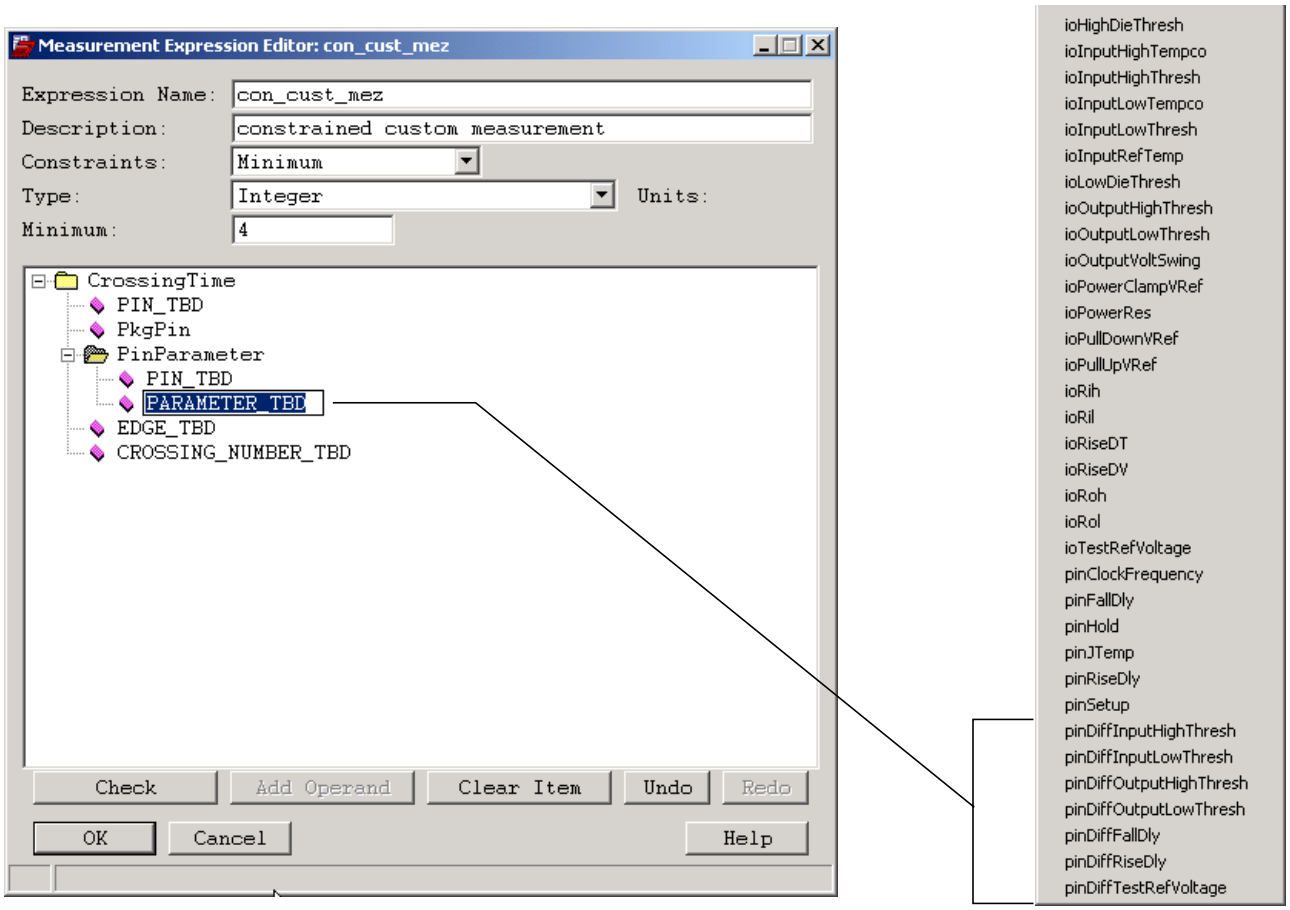


Table C-2 Differential pin parameters

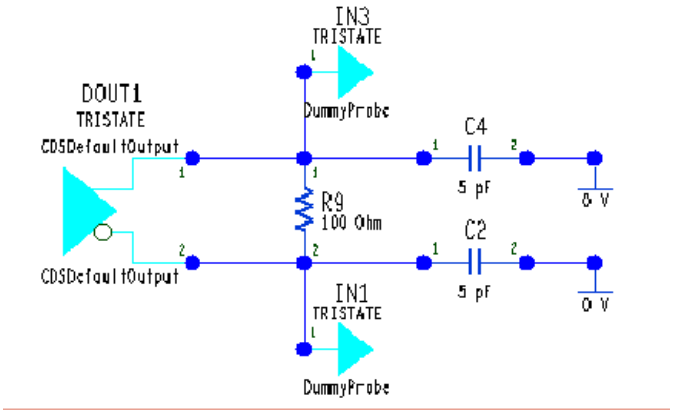
Use this pin parameter . . .	To obtain this . . .
`pinDiffInputHighThresh`	Voltage-based differential high-state threshold
`pinDiffInputLowThresh`	Voltage-based differential low-state threshold
`pinDiffOutputHighThresh`	The manufacturer's voltage-based minimum differential high-state output at full load

Use this pin parameter . . .	To obtain this . . .
`pinDiffOutputLowThresh`	The manufacturer's voltage-based minimum differential low-state output at full load
`pinDiffFallDly`	Time-based differential buffer delay at fall edge
`pinDiffRiseDly`	Time-based differential buffer delay at rise edge
`pinDiffTestRefVoltage`	Voltage-based reference threshold for differential buffer delay

Differential Buffer Delays Test Fixture

When differential time measurement is related to buffer delay, SigXplorer uses the buffer delay value of the non-inverting and inverting drivers. In theory, this may be inaccurate because differential buffer delay could be defined either on differential signal or single-end pin. Even when the measurement is taken on the single-end pin, the test fixture may differ from the simple (non-differential) IBIS test fixture.

For example, you typically place a resistance terminator between the differential pin pair for differential impedance matching. Since IBIS does not specify the differential test fixture, you must define your own in SigXplorer.



We recommend that you complete the *Differential Buffer Delays* parameters in the buffer delay dialog box. You can edit these parameters only when a pin-pair is defined in the *Ibis Device Pin Data* dialog box; otherwise, these fields are grayed out. You can edit the *Diff Rise Delay* and *Diff Fall Delay* fields. You must reference a two-node Espice model for the test fixture (See To specify differential buffer delays on page 172).

The “Measure Delays” checks the setting of the test fixture, creates signoise simulations to obtain the buffer delay values for all Fast/Typical/Slow modes, and fills the values in the corresponding fields (see the “Buffer Delays dialog box” on page 175). SigXplorer captures all this information in a topology file.

To specify differential buffer delays

1. In SigXplorer, choose *Analyze – Library*.

   The Signal Analysis Library Browser appears.

2. Choose a device library.

3. Click *Browse Models*.

   The Model Browser appears.

4. Choose a device.

5. Click *Edit*.

   The IBIS Device Model Editor appears.

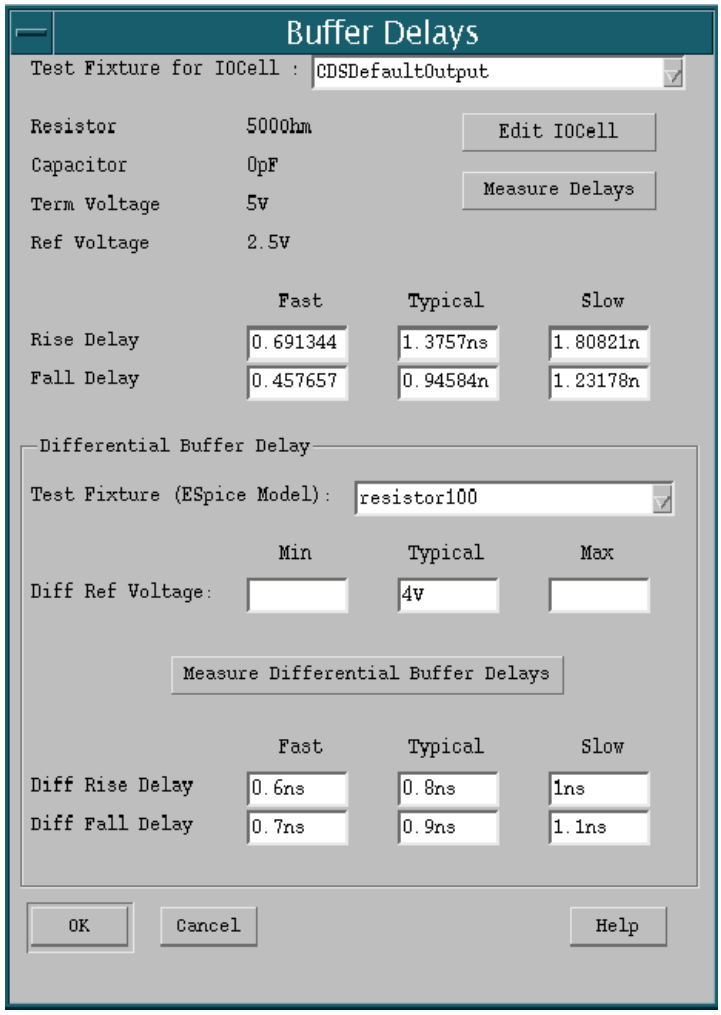
6. Double-click on a differential pin.

   The IBIS Device Pin Data dialog box appears.

7. Click *Buffer Delays*.

   The Buffer Delays dialog box appears.

Figure C-5 Buffer Delays dialog box



8. In the Test Fixture (ESpice model) field, choose a discrete from the drop-down menu.

   The Measure Differential Buffer Delays button becomes active.

9. Click the *Measure Differential Buffer Delays*.

   SigXplorer calculates Fast/Typical/Slow values for *DiffRiseDelay* and *DiffFallDelay*.

   Note: If *Diff Ref Voltage* is not specified, SigXplorer assumes a reference voltage value equal to zero.

D

Working with Source Synchronous Custom Measurements

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. 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, consider it 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.

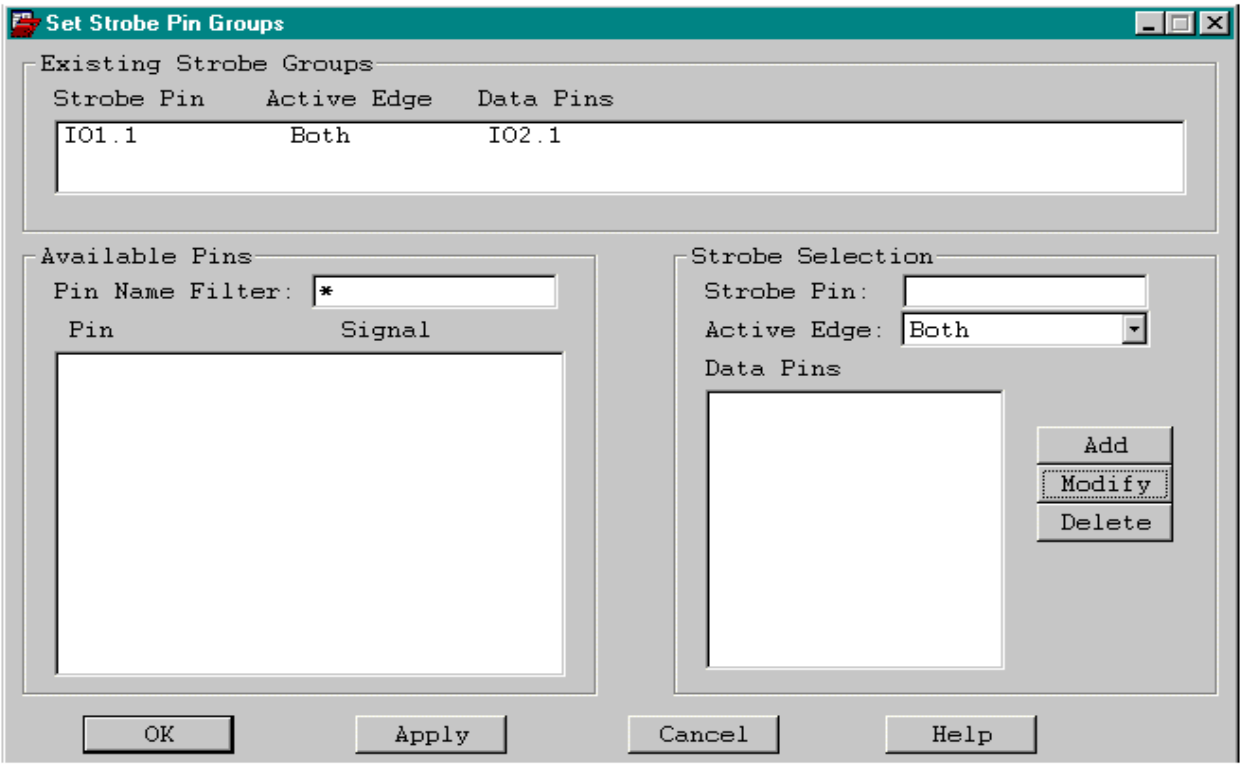
The hold margin measures the difference between the 23ns marker (at the strobe pin's rising edge) and the data pin's next transition through 2.5V, which happens at about 27.5ns. This produces a hold margin of 27.5 - 23 = 4.5ns.

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

Use *Set – Strobe Pins* to 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 D-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.

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.

Available Pins and Signals

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.

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.

Identifying Active Edges for the Strobe Pin

In the *Active Edge* field select *Rising*, *Falling*, or *Both* to specify the active edges on the strobe signal when data pin measurements trigger. For most source synchronous designs, data measures on both the rising and falling edges.

When you finish editing, use *Add* to create the strobe pin group and display it in the *Existing Strobe Groups* list box.

Modifying and Deleting Existing Strobe Pin Groups

To modify an existing strobe pin group, 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. Make your modifications to the data fields in the *Strobe Selection* area. Then use *Modify* to update the strobe pin group and redisplay it in the *Existing Strobe Groups* list box.

To remove an existing strobe pin group, 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.

Use *Delete* to remove the strobe pin group from the *Existing Strobe Groups* list box. The strobe and data pins redisplay in the *Available Pins* list box.

Source Synchronous Topology Files

When you are through making changes in the Set Strobe Pin Groups dialog box, use *OK* or *Apply* to add the strobe pin groups displayed in the *Existing Strobe Groups* list box to the topology. *OK* and *Apply* also mark the topology as a source synchronous topology and change the symbols associated with the Strobe and Data pins in the canvas to indicate that they are strobe or data pins.

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.

Strobe and Data Pins in the Canvas

Figure D-2 shows a canvas symbol for an IOCell marked as a strobe pin.

Figure D-2 IOCell Marked as a Strobe Pin

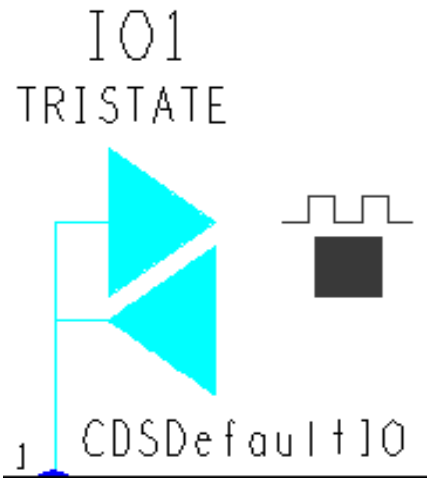
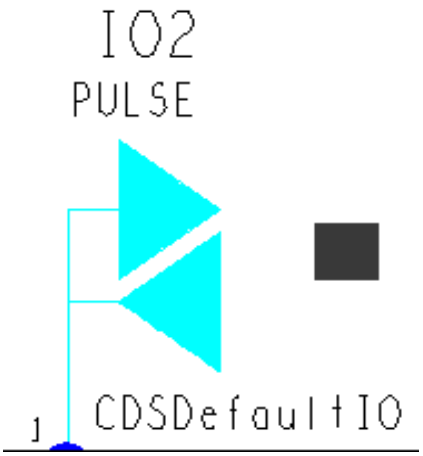


Figure D-3 shows a symbol for an IOCell marked as a data pin.

Figure D-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.

Adding a Strobe Pin Group

To create a new strobe pin group

1. Choose *Set – Strobe Pins*.

   The Set Strobe Pin Groups dialog box appears.

2. If necessary, modify the contents of the *Pin Name Filter* field to restrict the pins listed in the *Available Pins* list box.

3. In the *Available Pins* list box, select the strobe pin. 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.

   The selected pin moves from the *Available Pins* list box, to the *Strobe Pin* field in the *Strobe Selection* area.

4. In the *Active Edge* field, pull-down the menu of active edge choices and select either *Rising* edge, *Falling* edge or *Both* rising and falling edges.

5. 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.

6. Click *Add* to add the new strobe pin group list of existing strobe pin groups.

   The strobe pin group appears in the *Existing Strobe Groups* area and the *Strobe Selection* area is cleared.

7. Click *OK* or *Apply* to add the strobe pin group to the topology.

Editing a Strobe Pin Group

To edit a strobe pin group

1. Choose *Set – Strobe Pins*.

   The Set Strobe Pin Groups dialog box displays.

2. If necessary, modify the contents of the *Pin Name Filter* field to restrict the pins listed in the *Available Pins* list box.

3. In the *Existing Strobe Groups* list box, select the strobe pin group to modify.

   The strobe pin group is highlighted in the *Existing Strobe Groups* list box, and the data is copied to the fields in the *Strobe Selection* area.

4. In the *Active Edge* field, pull-down the menu of active edge choices and select either *Rising* edge, *Falling* edge or *Both* rising and falling edges.

5. In the *Available Pins* list box, select a data pin.

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

6. In the *Data Pins* list box, select the data pin to remove.

   The selected data pin moves from the *Data Pins* list box, to the *Available Pins* list box. Its designation as a data pin is removed.

7. In the *Strobe Selection* area, click *Modify* to make the modifications to the strobe pin group.

The modifications to the strobe pin group are displayed in the Existing Strobe Groups area and the Strobe Selection area is cleared.

8. Click *OK* or *Apply* to add the modified strobe pin group to the topology.

Deleting a Strobe Pin Group

To delete a strobe pin group

1. Choose *Set – Strobe Pins*.

   The Set Strobe Pin Groups dialog box displays.

2. If necessary, modify the contents of the *Pin Name Filter* field to restrict the pins listed in the *Available Pins* list box.

3. In the *Existing Strobe Groups* list box, select the strobe pin group to delete.

   The strobe pin group highlights in the *Existing Strobe Groups* list box, and the data copies to the fields in the *Strobe Selection* area.

4. In the *Strobe Selection* area, click *Delete* to delete the strobe pin group.

   The strobe pin group deletes from the *Existing Strobe Groups* area, the fields in the *Strobe Selection* area clear, and all pins display in the *Available Pins* list box. The Data Pins and Strobe Pin designations disappear from the pins.

5. Click *OK* or *Apply* to remove the strobe pin group from the topology.