cadence®

Topology Workbench Tcl Command Reference

Product Version 23.1
October 2023

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

Contents

1
Using the Topology Workbench Tcl Commands: An Overview
7

Scope of this Document . . . 7
Prerequisites . . . 7
Using the Tcl Commands . . . 8
Running Tcl Commands . . . 8
Related Documents . . . 8
Tutorials . . . 8
Frequently Asked Questions . . . 8
Additional Learning Resources . . . 9
Customer Support . . . 9

2
Topology Workbench Tcl Commands . . . 11

Managing Topology Workbench Environment . . . 12
setEnv . . . 12
setSigrityEDADir . . . 13
testEnv . . . 13
unsetEnv . . . 14
Adding Elements to a Topology . . . 14
addAMI . . . 15
addNewBlocksWidthDiff . . . 15
addRepeaterConnection . . . 16
addToAMM . . . 16
addWhatIf . . . 17
appendTopology . . . 17
newBlockWithSinglePin . . . 18
Enabling Elements in a Topology . . . 18
enableConfigurePortsAutomatically . . . 18

enableDistributedComputing . . . 19
enableUseChannelSimulator . . . 19
Editing Elements in a Topology . . . 19
editAnalysisOption . . . 21
editBlock . . . 34
editDieModel . . . 45
editFrequencyResponse . . . 46
editMCPConnection . . . 47
editOptions . . . 48
editProbePoints . . . 53
editSParamExtraction . . . 55
editTimingBudget . . . 57
editWriteLeveling . . . 58
makeReceiver . . . 59
makeTransmitter . . . 59
renameComponent . . . 60
restoreDefaultPins . . . 60
setupDefaultPinLocation . . . 61
shortToGND . . . 62
shortToPower . . . 62
shortUnConnectedGroundNode2GND . . . 63
Displaying Elements in a Topology . . . 63
displayPin . . . 63
displayUnConnectedGroundNode . . . 64
displayUnConnectedPowerNode . . . 64
displayUnConnectedSignalPins . . . 65
Controlling Simulation and Analysis . . . 65
checkConnectivity . . . 67
deriveGelementCurrent . . . 68
deriveTargetImpedance . . . 69
runJitterTolerance . . . 70
setOption . . . 70
showResult . . . 71
startACAnalysis . . . 72
startBlockSensitivity . . . 72
startBusChannelCharacterization . . . 73

startBusSimulation . . . 73
startChannelCharacterization . . . 74
startChannelSimulation . . . 74
startCircuitChannelCorrelation . . . 75
startConnectivityCheckSimulation . . . 75
startDCAnalysis . . . 76
startImpedanceSimulation . . . 76
startPowerRippleSimulation . . . 77
startSParamExtraction . . . 77
startSimulation . . . 78
startSweepSimulation . . . 78
startTransientAnalysis . . . 79
startVoltusModelExtraction . . . 79
startWriteLevelingSimulation . . . 80
stopSimulation . . . 80
terminateUnconnectedNodes . . . 81
Exporting from Topology Workbench . . . 81
exportBlockSensitivtyResults . . . 82
exportReport . . . 83
exportToEcsetPage . . . 84
extractECSet . . . 85
Managing Projects . . . 85
archiveProject . . . 86
assignSimName . . . 88
checkCompliance . . . 89
cleanup . . . 89
connectToPowerSupply . . . 90
convertToWireBased . . . 90
copyProjectAs . . . 91
defineBlockGroup . . . 91
dumpPropertyWindow . . . 92
editPinArg . . . 93
getLicensesUsed . . . 93
ibisEditor . . . 94
launchAbout . . . 94
launchAMM . . . 95

log . . . 95
purgeUnusedModels . . . 96
queryLicenseUsage . . . 96
saveEcsetPageAsTopFile . . . 97
terminateWithR . . . 97
Generating Via Models . . . 97
auroraViaWizard . . . 98
sigrityViaWizard . . . 98
Managing AMI Models . . . 98
deleteAMI . . . 99
modelEnable . . . 99
Handling Constraint Manager . . . 100
setDesignPrecision . . . 100
setOptionalPins . . . 101
updateCM . . . 101
Toggling in Topology Workbench . . . 101
toggleAutoShort . . . 102
toggleBackChannel . . . 102
togglePauseBeforeSim . . . 103
toggleShowNewMCP . . . 103
toggleTxRxJitterNoise . . . 104

1

Using the Topology Workbench Tcl Commands: An Overview

Topology Workbench includes a vast collection of Tcl commands that are used internally when designers perform tasks using its interface menu options. These Tcl commands are also available to anyone who needs to create scripts to modify the default built-in functionality, or wants to accomplish design tasks without clicking and navigating the user interface.

Scope of this Document

This document lists the Tcl commands that are available and describes how to use them. Knowledge of Tcl concepts is mandatory to be able to use and extend the Topology Workbench Tcl commands.

To learn the basics of Tcl commands, as in the language syntax, operands, assignments, substitutions, and so on, refer to https://www.tcl.tk/man/tcl8.6/TclCmd/info.html. Many of the Tcl commands are for Internal use only and details for their behavior and parameters are not disclosed. Do not use these commands directly on your designs. If you require the functionality these commands offer, contact a Cadence representative for information and help. It is recommended you use only those commands that are listed in this document.

Prerequisites

Before you implement Topology Workbench Tcl commands, you should be familiar with the following:

- Basics of schematic design tasks
- Fundamentals of Topology Workbench
- Basics of Tcl scripting

Using the Tcl Commands

The standard Tcl commands are used to extract the details from the built-in Tcl tools and commands within Topology Workbench. Only the relevant commands and features for System Capture are listed and explained. Topology Workbench Tcl commands are based on Tcl version 8.6. There is no Tk support.

Running Tcl Commands

The Command Window in the Topology Workbench window lets you run the supported Tcl commands and view the ones that are run in the background when you perform a corresponding GUI action. To open this panel,

1. Choose the *View – Command Window* menu.
2. Type the command on the Tcl> prompt.

   The commands specific to Topology Workbench use the following general syntax:

   ```
   topxp::<command_name> <arguments/values>
   ```

Related Documents

In addition to the Topology Workbench User Guide, you can refer to the following documents:

Tutorials

- *Topology Workbench: Topology Explorer Tutorial*

  This tutorial captures the step-by-step instructions on exploring the Topology Workbench canvas, creating a topology from scratch, doing pre-layout extraction and post-layout routed interconnect extraction, and updating the ECSet using Constraint Manager.

- *Topology Workbench: Parallel Bus Analysis Tutorial*

  This tutorial covers typical steps you will perform to create, edit, and simulate a parallel bus interface by using a default template from the install hierarchy and by creating a topology from scratch.

Frequently Asked Questions

- *Topology Workbench Frequently Asked Questions*

This document can be read to find answers to a few commonly encountered questions related to Topology Workbench.

Additional Learning Resources

Cadence offers training courses that enable you to understand the applications better. For specific information about the courses available in your region, visit Cadence Training or write to training_enroll@cadence.com.

Note: The links in this section open in a separate web browser window when clicked in Cadence Help.

Customer Support

For assistance with Cadence products:

- Contact Cadence Customer Support

  Cadence is committed to keeping your design teams productive by providing answers to technical questions and to any queries about the latest software updates and training needs. For more information, visit: https://www.cadence.com/support

- Log on to Cadence Online Support

  Customers with a maintenance contract with Cadence can obtain the latest information about various tools at: https://support.cadence.com

2

Topology Workbench Tcl Commands

Topology Workbench supports use of Tcl commands to perform the following tasks:

- Managing Topology Workbench Environment
- Adding Elements to a Topology
- Enabling Elements in a Topology
- Editing Elements in a Topology
- Displaying Elements in a Topology
- Controlling Simulation and Analysis
- Exporting from Topology Workbench
- Managing Projects
- Generating Via Models
- Managing AMI Models
- Handling Constraint Manager
- Toggling in Topology Workbench

Managing Topology Workbench Environment

You can use the following Tcl commands in the Command Window panel:

- setEnv
- setSigrityEDADir
- testEnv
- unsetEnv

setEnv

Syntax	`topxp::setEnv \`
	`  {<environment>} \`
	`  {<value>}`	

Description	Set runtime environment value as {environment} = {value}	
Arguments	`{<environment>}`	
		Name of the environment.
	`{<value>}`	
		The value that will replace the current environment value.
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

setSigrityEDADir

Syntax	`topxp::setSigrityEDADir {<sigrity_installation_path>}`	
Description	Sets the path to the Sigrity installation directory during a running session to allow access to the latest Sigrity engines and functionality. You should alternatively set the `SIGRITY_EDA_DIR` environment variable before starting Topology Workbench. The approach of setting this environment variable is recommended because the path set using the `setSigrityEDADir` Tcl command is retained only in the current Topology Workbench session.	
Arguments	`{<sigrity_installation_path>}`	
		Specifies the path to the Sigrity installation.
GUI Equivalent	None.	

Related Topic

Prerequisites for Running Topology Workbench and Other SystemSI Products

Using the Topology Workbench Tcl Commands: An Overview

testEnv

Syntax	`topxp::testEnv \` `  {environment}`	
Description	Displays the runtime environment value.	
Arguments	`{<environment>}`	
		Specifies the name of the environment.
GUI Equivalent	None.	

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

unsetEnv

Syntax	`topxp::unsetEnv \` `  {environment}`	
Description	Unset runtime environment.	
Arguments	`{<environment>}`	
		Specifies the name of the environment.
GUI Equivalent	None.	

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

Adding Elements to a Topology

You can use the following Tcl commands in the Command Window panel:

- addAMI
- addNewBlocksWidthDiff
- addRepeaterConnection
- addToAMM
- addWhatIf
- appendTopology
- newBlockWithSinglePin

addAMI

Syntax	`topxp::addAMI \`
	`  {<block_name>} \`
	`  {<executable_file>} \`
	`  {<parameter_file>}`	

Description	Adds an AMI model to a block with the specified executable file and parameter file.	
Arguments	`<block_name>`	
		Specifies the block name.
	`<executable_file>`	
		Specifies the path to the executable file.
	`<parameter_file>`	
		Specifies the path to the parameter file.
GUI Equivalent	None	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

addNewBlocksWidthDiff

Syntax	`topxp::addNewBlocksWidthDiff`
Description	When toggled, adds new blocks with differential signals.
Arguments	None
GUI Equivalent	The *Diff Signals* toggle button in the *Settings* option of the floating toolbar.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

addRepeaterConnection

Syntax	`topxp::addRepeaterConnection`
Description	Adds a repeater connection between the Tx and Rx blocks.
Arguments	None
GUI Equivalent	Context Menu - Add Repeater Connection

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

addToAMM

Syntax	`topxp::addToAMM`
Description	Adds or updates models of the selected block to AMM. Currently Spice model and IBIS model are supported.
Arguments	None
GUI Equivalent	Context Menu - Add To AMM

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

addWhatIf

Syntax	`topxp::addWhatIf \` `  <block name>:<pin name>`	
Description	Adds a what-if decap.	
Arguments	`<block_name>`	
		Specifies the block name.
	`<pin name>`	
		Specifies the name of the pin.
GUI Equivalent	Context Menu - "Add What-If Decaps"	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

appendTopology

Syntax	`topxp::appendTopology`
Description	Appends an existing topology (`.topx` or legacy `.top`, `.ssix`) to current topology.
Arguments	None.
GUI Equivalent	In the Topology Workbench window, the *Topology – Append...* menu.

Related Topics

Appending to a Topology

Using the Topology Workbench Tcl Commands: An Overview

newBlockWithSinglePin

Syntax	`topxp::newBlockWithSinglePin`
Description	Toggles adding new blocks with block-based connectivity.
Arguments	None.
GUI Equivalent	The *Block-Based* toggle button in the *Settings* option of the floating toolbar.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

Enabling Elements in a Topology

You can use the following Tcl commands in the Command Window panel:

- enableConfigurePortsAutomatically
- enableDistributedComputing
- enableUseChannelSimulator

enableConfigurePortsAutomatically

Syntax	`topxp::enableConfigurePortsAutomatically`
Description	Toggles *Configure Ports Automatically*.
Arguments	None.
GUI Equivalent	In the *Workflow* pane of Custom Compliance Kit, the *Configure Ports Automatically* check box in the *Simulation Setup* schema.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

enableDistributedComputing

Syntax	`topxp::enableDistributedComputing`
Description	Toggles *Enable Distributed Computing*.
Arguments	None.
GUI Equivalent	In the *Workflow* pane of supported workflows, the *Enable Distributed Computing* check box in the *Distributed Computing Setup* schema.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

enableUseChannelSimulator

Syntax	`topxp::enableUseChannelSimulator`
Description	Toggles the *Use Channel Simulator* check box.
Arguments	None.
GUI Equivalent	In the *Workflow* pane of Parallel Bus Analysis, the *Use Channel Simulator* check box in the *Simulation Setup* schema.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

Editing Elements in a Topology

- editAnalysisOption
- editDieModel
- editFrequencyResponse
- editProbePoints
- editSParamExtraction
- editTimingBudget
- editWriteLeveling

- makeReceiver
- makeTransmitter
- renameComponent
- restoreDefaultPins
- setupDefaultPinLocation
- shortToGND
- shortToPower
- shortUnConnectedGroundNode2GND

editAnalysisOption

Syntax

```
editAnalysisOption \
     -autotimestep <Boolean> \
     -timestep <step> \
     -timestop <stop> \
     -busType <bus_type> \
     -datarate <data_rate> \
     -datarate [list {Block <block>} {BusGroup <busgroup>} {Signal
     <signal>} {<data_rate>}] \
     -baudrate [list {Block <block>} {BusGroup <busgroup>} {Signal
     <signal>} {<baud_rate>}]\
     -numberofbits [list {Block <block>} {BusGroup <busgroup>}
     {Signal <signal>} {<bit_num>}]\
     -minnumberofbits <bit_num> \
     -corner {<corner_type> <Boolean>} \
     -writedirection <Boolean> \
     -readdirection <Boolean> \
     -memoryiomodelshared <Boolean> \
     -wloenabled <Boolean> \
     -clockingtype <clocking_type> \
     -memoryrank  [list {Rank <mem_name> … <mem_name> <IsActive>}
     {Rank <mem_name> ... <mem_name> <IsActive>}] \
     -ignoretime [list {Time <time>} {Unit <unit>}] \
     -channelminnumberofbits <bit_num> \
     -bitsamplingrate <bit_sampling_rate> \
     -checkberfloor <Boolean> \
     -lberfloor <lber_floor> \
     -channelnumberofbitsfordisplay <bit_num> [{Option
     <option>}]\
     -timescale <Boolean> \
     -voltagescale <Boolean> \
     -boththtimeandvoltage <Boolean> \
     -lber {<Log_ber>}
     -eyeDistributionMethod <method> \
     -stimuluspattern  [list {Block <block>} {BusGroup <busgroup>}
     {StimulusType <stimulus_type>} {Seed <seed>} {Poly <poly>}
     {DataFile <datafile>} {BitPattern <bit_pattern>} {DCD <dcd>}
     {PulseType <pulse_type>} {Init <init>} {SwitchTimes
     <switch_time>}  {DataCodingEnabled <bool>} {DataCoding
     <data_coding>} {RiseFallTimeEnabled <bool>} {RiseTime
     <rise_time>} {FallTime <fall_time>}] \
     -standbyiomodel [list {Block <block>} {Bus <bus>} {BusGroup
     <busgroup>} {Signal <signal>} {<iomodel>}] \
```

	`-transmitjitternoise [list {Block <block>} {BusGroup <busgroup>} {JitterSinusoidalEnabled <bool>} {JitterSinusoidalFrequency <value>} {JitterSinusoidalAmplitude <value>} {JitterFrequencyOffsetEnabled <bool>} {JitterFrequencyOffset <value>} {JitterTransitionRjEnabled <bool>} {JitterTransitionRj <value>} {JitterTransitionDjEnabled <bool>} {JitterTransitionDj <value>} {JitterDCDEnabled <bool>} {JitterDCD <value>} {NoiseSinusoidalEnabled <bool>} {NoiseSinusoidalFrequency <value>} {NoiseSinusoidalAmplitude <value>} {NoiseTransitionEnabled <bool>} {NoiseTransition <value>}] \` `-receivejitternoise [list {Block <block>} {BusGroup <busgroup>} {Signal <signal>} {JitterRandomEnabled <bool>} {JitterRandom <value>} {JitterDeterministicEnabled <bool>} {JitterDeterministic <value>} {NoiseRandomEnabled <bool>} {NoiseRandom <value>} {NoiseDeterministicEnabled <bool>} {NoiseDeterministic <value>}] \` `-check [list {Block <block>} {BusGroup <busgroup>} {Signal <signal>} {<bool>}] \` `-chsimAnalMode <mode> \` `-simulator spdsim|hspice|spectre \` `-circuitSimulatorOption <option> \` `-channelSimulatorControl <option> \` `-spectrePerformanceOption [list <option> … <option>] \` `-spectreSimulatorOption [list <option> … <option>] \` `-spectreTransientOption [list <option> … <option>] \` `-characterizationTimeStep <value> \` `-launchDelay <value> \` `-stepDuration <value> \` `-vmeas <value> \` `-busStimuliConfiguration Random|Even \` `-seedControl <value> \` `-adaptiveTimeStep \` `-restoredefaults`	
Description	Sets the simulation options in the *Analysis Options* panel.	
Arguments	`-autotimestep <boolean>`	
		Selects or deselects the *Auto* check box for time step and time stop. **Note:** This argument applies only to the Topology Explorer workflow.

	`-timestep <step>`	
		Specifies the time step, which is a positive number. **Note:** This argument applies only to the Topology Explorer workflow.
	`-timestop <stop>`	
		Specifies the time stop, which is a positive number. **Note:** This argument applies only to the Topology Explorer workflow.
	`-busType <bus_type>`	
		Specifies the *Bus Type*. The valid `<bus_type>` can be one of the following: `Data`, `Ctrl`, `AddCmd` **Note:** This argument applies only to the Parallel Bus Analysis (PBA) workflow.
	`-datarate <data_rate>`	
		Specifies the *Data Rate*, which is a positive number. **Note:** This argument applies only to the PBA workflow.
	`-datarate [list {Block <block>} {BusGroup <busgroup>} {Signal <signal>} {<data_rate>}]`	
		Sets the *Data Rate* for the specified signal. In this argument's syntax, the following values can be set: ■ `<block>`, a string to specify the block name. ■ `<busgroup>`, a string to specify the bus group name. ■ `<signal>`, a string to specify the signal name. ■ `<data_rate>`, a positive number. **Note:** This argument applies to the Topology Explorer and Serial Link Analysis (SLA) workflows.

	`-baudrate [list {Block <block>} {BusGroup <busgroup>} {Signal <signal>} {<baud_rate>}]`	
		Sets the *Baud Rate* for the specified signal. In this argument's syntax, the following values can be set: ■ `<block>`, a string to specify the block name. ■ `<busgroup>`, a string to specify the bus group name. ■ `<signal>`, a string to specify the signal name. ■ `<baud_rate>`, a positive number. **Note:** This argument applies only to the SLA workflow.
	`-numberofbits [list {Block <block>} {BusGroup <busgroup>} {Signal <signal>} {<bit_num>}]`	
		Sets the *# of Bits* value for the specified signal. Here, `<bit_num>` can be an integer value, [1, 5000] **Note:** This argument applies only to the Topology Explorer workflow.
	`-minnumberofbits <bit_num>`	
		Sets the *Minimum # of Bits* value. **Note:** This argument applies only to the PBA workflow.
	`-corner {<corner_type> <Boolean>}`	
		Select or deselect the corner type check box. Here, `<corner_type>` can be set to one of the following: `Slow`, `Typ`, `Fast`, `Fast/Slow`, `Slow/Fast`
	`-writedirection <Boolean>`	
		Select or deselect the *Write* direction check box. **Note:** This argument applies only to the PBA workflow.

	`-readdirection <Boolean>`	
		Select or deselect the *Read* direction check box. **Note:** This argument applies only to the PBA workflow.
	`-memoryiomodelshared <Boolean>`	
		Select or deselect the *Memory Block Shared IO Models* check box. **Note:** This argument applies only to the PBA workflow.
	`-wloenabled <Boolean>`	
		Select or deselect the *WLO/CktMeasDelay* check box. **Note:** This argument applies only to the PBA workflow.
	`-clockingtype <clocking_type>`	
		Sets the *Signal Clocking* value. Here, `<clocking_type>` can be set to one of the following values: `1T`, `2T`, `3T` **Note:** This argument applies only to the PBA workflow.
	`-memoryrank [list {Rank <mem_name> … <mem_name> <IsActive>} {Rank <mem_name> ... <mem_name> <IsActive>}]`	
		Sets the *Memory Rank*. In this argument's syntax, the following values can be set: ■ `<mem_name>`, a string to specify the memory name. ■ `<IsActive>`, a Boolean value of `true` or `false`. **Note:** This argument applies only to the PBA workflow.

	`-ignoretime [list {Time <time>} {Unit <unit>}]`	
		Sets the *Ignore Time* or *Ignore Bits* value. In this argument's syntax, the following values can be set: ■ `<time>`, a positive number. ■ `<unit>`, the unit as `ns` or `bits`. **Note:** This argument applies only to the PBA channel simulation mode and SLA workflow.
	`-channelminnumberofbits <bit_num>`	
		Sets the *Minimum # of Bits* value. Here, `<bit_num>` is a positive integer. **Note:** This argument applies only to the PBA channel simulation mode and SLA workflow.
	`-bitsamplingrate <bit_sampling_rate>`	
		Sets the *Bit Sampling Rate* value. Here, `<bit_sampling_rate>` is a positive number. **Note:** This argument applies only to the PBA channel simulation mode and SLA workflow.
	`-checkberfloor <Boolean>`	
		Select or deselect the *Log BER Floor* check box. **Note:** This argument applies only to the PBA channel simulation mode and SLA workflow.
	`-lberfloor <lber_floor>`	
		Sets the *Log BER Floor* value. Here, `<ber_floor>` is an integer value, `[-21, -3]`. **Note:** This argument applies only to the PBA channel simulation mode and SLA workflow.

	`-channelnumberofbitsfordisplay <bit_num> [{Option <option>}]`	
		Sets the *# of Bits for Display* value to define the number of bits worth of raw waveforms to be saved to disk and displayed. Here, `<bit_num>` is a positive integer value. If `{Option <option>}` is specified, `<option>` can be set as `first` or `last` for *# of Bits for Display*. This value specifies whether to save and display the last or the first number of bits. For example, if you choose to save the last number of bits, the first several bits from the very beginning are ignored while viewing the eye diagram or generating the report.1 **Note:** This argument applies only to the PBA channel simulation mode and SLA workflow.
	`-timescale <Boolean>`	
		Selects or clears the *Time Scale* check box in the *Channel Simulation* tab of the Analysis Options pane to specify the type of *BER_Eyes* to be generated. **Note:** This argument applies to the PBA channel simulation mode and SLA workflow.
	`-voltagescale <Boolean>`	
		Selects or clears the *Voltage Scale* value. **Note:** This argument applies to the PBA channel simulation mode and SLA workflow.
	`-bothtimeandvoiltage <Boolean>`	
		Selects or clears the *Both time and voltage* value. **Note:** This argument applies to the PBA channel simulation mode and SLA workflow.

	`-lber {<log_ber>}`	
		Sets the *Log BERs* value. `<log_ber>` sets one of the following values: ■ A single `log BER [-12]` ■ Multiple `log BERs [-6,-9,-12]` ■ A range of `log BERS [-12,-15]` **Note:** Ensure that all `<log_ber>` values are negative integer values in the range of `[-21,-3]`.
	`-eyeDistributionMethod <method>`	
		Sets the *Eye Distribution Method.* Here, `<method>` can be *Time Domain Waveform* or *Statistical*.
	`-xtalk <option>`	
		Sets the *xTalk Mode.* For time domain, `<option>` can be one of the following: ■ *Ignore xTalk* ■ *Invert All Aggressor Stimulus* ■ *Use Aggressor Stimulus As Defined* ■ *Random Aggressor Stimulus* For statistical, `<option>` can be one of the following: ■ *Ignore xTalk* ■ *Include xTalk* **Note:** This argument applies to the PBA simulation mode, SLA workflow. The *Ignore xTalk* and *Include xTalk* options are available only for the statistical mode.

	`-signaling [list {Block <block>} {Signal <signal>} {<signal_type>}]`	
		Sets *Signaling* for the specified signal. `<signal_type>` can be one of the following: `NRZ`, `PAM3`, `PAM4`, `PAM6`, `PAM8`, or `MIPI-C`. **Note:** `MIPI-C` is valid for only `MIPI-C` compliance.
	`-leadingbits [list {Block <block>} {Signal <signal>} {<leadingbits_file>}]`	
		Sets *Leading Bits* for the specified signal.
	`-defaultstimuluspattern [list {Block <block>} {BusGroup <busgroup>}]`	
		Resets to default stimulus pattern for the specified signal.
	`-stimuluspattern [list {Block <block>} {BusGroup <busgroup>} {StimulusType <stimulus_type>} {Seed <seed>} {Poly <poly>} {DataFile <datafile>} {BitPattern <bit_pattern>} {DCD <dcd>} {PulseType <pulse_type>} {Init <init>} {SwitchTimes <switch_time>} {DataCodingEnabled <bool>} {DataCoding <data_coding>} {RiseFallTimeEnabled <bool>} {RiseTime <rise_time>} {FallTime <fall_time>}]`	
	Sets *Stimulus Pattern* for the specified signal. Here, `<stimulus_type>` can be one of the following: ■ For SLA workflow: `Random`, `PRBS`, `Data File`, `Sinusoidal Waveform`, `Sawtooth`, or `Clock` ■ For PBA workflow: `Bit Pattern`, `Random`, `PRBS`, `Data File`, or `Worst Case` ■ For Topology Explorer workflow: `Random`, `PRBS`, `Data File`, `Pulse`, `PWL` The other values that can be specified are: ■ `<seed>` is a positive integer for `Random` in PBA simulation mode. ■ `<poly>` specifies `1` integer value for `PRBS`.	

	■ <bit_pattern> specifies the bit pattern, for example "01.." ■ <dcd> is a double value for Pulse in Topology Explorer workflow ■ <pulse_type> can be set for Pulse in Topology Explorer workflow to Pulse, Rise, Fall, QuietHi, or QuietLo. ■ <init> can be set to 0 or 1 for PWL in Topology Explorer workflow ■ <switch_time> specifies the list of time such as 2e-8 4e-8. ■ <data_coding> can be one of the following: 8b10b, 64b66b, 64b67b, 128b130b, 128b132b, 16b18b, or Gray Coding ■ <rise_time> is a positive number and less than Bit Period/2. ■ <fall_time> is a positive number and less than Bit Period/2.	
	-stimulusoffset [list {Block <block>} {BusGroup <busgroup>} {offset}]	
		Sets the *Stimulus Offset* for the specified signal. Here, <offset> is a positive number.
	-transmitiomodel [list {Block <block>} {Bus <bus>} {BusGroup <busgroup>} {Signal <signal>} {<iomodel>}]	
		Sets the *Tx IO Model* for the specified signal. Here, <iomodel> is a ibis model.
	-receiveiomodel [list {Block <block>} {Bus <bus>} {BusGroup <busgroup>} {Signal <signal>} {<iomodel>}]	
		Sets the *Rx IO Model* for the specified signal. Here, <iomodel> is a ibis model.
	-standbyiomodel [list {Block <block>} {Bus <bus>} {BusGroup <busgroup>} {Signal <signal>} {<iomodel>}]	
		Sets the *Standby IO Model* for the specified signal. Here, <iomodel> is a ibis model.

	`-transmitjitternoise [list {Block <block>} {BusGroup <busgroup>} {JitterSinusoidalEnabled <bool>} {JitterSinusoidalFrequency <value>} {JitterSinusoidalAmplitude <value>} {JitterFrequencyOffsetEnabled <bool>} {JitterFrequencyOffset <value>} {JitterTransitionRjEnabled <bool>} {JitterTransitionRj <value>} {JitterTransitionDjEnabled <bool>} {JitterTransitionDj <value>} {JitterDCDEnabled <bool>} {JitterDCD <value>} {NoiseSinusoidalEnabled <bool>} {NoiseSinusoidalFrequency <value>} {NoiseSinusoidalAmplitude <value>} {NoiseTransitionEnabled <bool>} {NoiseTransition <value>}]`	
		Sets the *Tx Jitter & Noise* for the specified signal. Here, `<value>` is a positive number.
	`-receivejitternoise [list {Block <block>} {BusGroup <busgroup>} {Signal <signal>} {JitterRandomEnabled <bool>} {JitterRandom <value>} {JitterDeterministicEnabled <bool>} {JitterDeterministic <value>} {NoiseRandomEnabled <bool>} {NoiseRandom <value>} {NoiseDeterministicEnabled <bool>} {NoiseDeterministic <value>}]`	
		Sets the *Rx Jitter & Noise* for the specified signal. Here, `<value>` is a positive number.
	`-check [list {Block <block>} {BusGroup <busgroup>} {Signal <signal>} {<bool>}]`	
		Selects or deselects the specified Tx signal. Here, `<value>` is a positive number.

	`-chsimAnalMode <mode>`	
		Sets the *Advanced Simulation Controls*. Here, `<mode>` specifies one of the following: ■ Liberal ■ Moderate ■ Conservative **Note:** This argument applies to the PBA simulation mode and SLA workflow.
	`-simulator spdsim | hspice | spectre`	
		Sets the *Circuit Simulator*.
	`-circuitSimulatorOption <option>`	
		Sets the *Circuit Simulator Option*.
	`-channelSimulatorControl <option>`	
		Sets the *Circuit Simulator Controls.*
	`-spectrePerformanceOption [list <option> … <option>]`	
		Sets the *Spectre Performance Option.*
	`-spectreTransientOption [list <option> … <option>]`	
		Sets the *Spectre Transient Option*.
	`-characterizationTimeStep <value>`	
		Sets the *Time Step* for characterization.
	`-launchDelay <value>`	
		Sets the *Launch Delay* for characterization.
	`-stepDuration <value>`	
		Sets the *Step Duration* for characterization.
	`-vmeas <value>`	
		Sets the *Vmeas* for characterization.

	`-busStimuliConfiguration Random | Even`	
		Sets the *Bus Stimuli Seed Control* for characterization when *PRBS* is selected in *Advanced Bus Characterization Options*.
	`-seedControl <value>`	
		Sets the *Seed* for characterization when *PRBS* is selected in *Advanced Bus Characterization Options*.
	`-adaptiveTimeStep`	
		Controls the value of the `TOPXP_ADAPTIVE_TIME_STEP` environment variable.
	`-restoredefaults`	
		Resets all modified values to their default settings.
GUI Equivalent	*Analysis Options* panel that opens when you click the *Setup – Analysis Options...* menu in the Topology Workbench window. Alternatively, you can click *Set Analysis Options* in the Workflow panel to open this panel.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

editBlock

Syntax	`editBlock \`
	`  <block> -newname <new_name> -blockgroup <group>`
	`  <block_type_specific_params>`	

Description	Edits the properties of the specified block.	
Arguments	`<block>`	Specifies the name of the block for which the properties need to be edited.
	`-newname <new_name>`	
		Specifies the new name for the block.
	`-blockgroup <group>`	
		Specifies the name of the group for controller, memory, and EBD blocks.
	`<block_type_specific_params>`	
		This argument can be one of the following types of parameters: ■ `<spice_block_params>` ■ `<via_block_params>` ■ `<sparam_block_params>` ■ `<device_block_params>` ■ `<ebd_block_params>` ■ `<trace_block_params>` ■ `<layout_block_params>` ■ `<descrete_block_params>` ■ `<terminator_block_params>` ■ `<vrm_block_params>` ■ `<clarity_block_params>` ■ `<source_block_params>`

	<spice_block_params>	

		Following is the syntax for SPICE block parameters:
		`-spicefile <spice_file> -subcircuit <subcircuit> -sweepfilemodels <sweepfilemodels>`
		Where,
		■ <spice_file> is the name of the SPICE file.
		■ <subcircuit> is the name of the subcircuit.
		■ <sweepfilemodels> is a list of SPICE files given as following:
		`[list "spice_file1" subcircuit1 "spice_file2" subcircuit2 ...]`

	<via_block_params>	
		Following is the syntax for Via block parameters: `-spicefile <spice_file> -subcircuit <subcircuit>` Where, ■ <spice_file> is the name of the SPICE file. ■ <subcircuit> is the name of the subcircuit.

	<sparam_block_params>	
		Following is the syntax for S Parameter block parameters: `-sparameterfile <sparameterfile> -2n <2n> -sweepfiles <sweepfiles> -removedcblockingcomponents <removedcblockingcomponents> -checkpassivity <checkpassivity> -optimizez0 <optimizez0>` Where, ■ <sparameterfile> is the name of the S Parameter file of .bnp, .s*p, or .ts file format. ■ <2n> is a Boolean, n+1, or 2n model where n is a port number. ■ <sweepfiles> is a list of S Parameter files given as following: [list {sparameterfile1} {sparameterfile2} ...{sparameterfileN}] ■ <removedcblockingcomponents> is a Boolean to remove DC blocking components. ■ <checkpassivity> is a Boolean to enforce passivity. ■ <optimizez0> is a Boolean that is used only in SystemPI.

	<device_block_params>	
		Following is the syntax for device block parameters: `-ibisfile <ibisfile> -component <component> -ignoreVTCurves <ignoreVTCurves> -onDieParasitic <onDieParasitic> -pkgParasitic <pkgParasitic> -shortResistor <shortResistor> -pdsResistor <pdsResistor> -pdsCapacitor <pdsCapacitor> -packageModelFile <packageModelFile> -packageModelName <packageModelName> -interconnectModeling <interconnectModeling> -interconnectModelGroup <interconnectModelGroup> -ioOnly <ioOnly> -singleendedpin <singleendedpin> -diffpin <diffpin> -extModelParam <extModelParam> -extModelConverterParam <extModelConverterParam>` Where, ■ `<ignoreVTCurves>` is a Boolean, that is, `1` or `0`. ■ `<onDieParasitic>` can be `None` or `OnDie RC` ■ `<pkgParasitic>` can be `None`, `Pin RLC`, or `Package Model`. ■ `<shortResistor>` is a number. ■ `<pdsResistor>` is a number. ■ `<pdsCapacitor>` is a number. ■ `<interconnectModeling>` can be `None` or `Interconnect Modeling`. ■ `<ioOnly>` is a Boolean, that is, `1` or `0`. ■ `<singleendedpin>` is the name of a single-ended pin. This argument value supports wildcard characters asterisk (*) and question mark (?). For example: `editBlock {Tx1} -singleendedpin {A*}` selects the single-ended pins with names beginning with A (case-insensitive).

		`editBlock {Tx1} -singleendedpin {*}` selects all the available single-ended pins.
		
		`editBlock {Tx1} -singleendedpin {A?}` selects the single-ended pins with names beginning with A (case-insensitive), and of maximum two characters length.
		
		■ `<diffpin>` is a list of positive and negative-edge pin names specified as:
		`[list <pos_pin_name> <neg_pin_name>]`
		
		■ `<extModelParam>` is a list of external model parameter names and their values specified as:
		`[list <external_model_param_name> <external_model_param_value>]`
		
		■ `<extModelConverterParam>` is a list of external model converter parameter names and their values specified as:
		`[list <external_model_converter_param_name> <external_model_converter_param_value>]`

	`<ebd_block_params>`	

		Following is the syntax for EBD block parameters:
		`-ebdAsPkg <ebdAsPkg> -ebdfile <ebdfile> \`
		`-ebdboard <ebdboard> \`
		`-ebdnodeoption <ebdnodeoption>`
		
		Where,
		
		■ `<ebdAsPkg>` is a Boolean, that is, `1` or `0`.
		
		■ `<ebdnodeoption>` is a list of IBIS files given as following:
		`[list <ibis_node> <ebd_dc_pin>]`

	<trace_block_params>	

		Following is the syntax for Trace block parameters:
		`-trcLayer <trcLayer> \`
		`-trcLayerRemove <trcLayerRemove> \`
		`-trcTrace <trcTrace> \`
		`-trcTraceRemove <trcTraceRemove> \`
		`-trcTraceData <trcTraceData> \`
		`-trcSweepParamSet <trcSweepParamSet> \`
		`-cableData <cableData> \`
		`-cableSweepParamSet <cableSweepParamSet> \`
		`-coaxCableData <coaxCableData> \`
		`-coaxCableSweepParamSet`
		`<coaxCableSweepParamSet> \`
		`-trcGlobalData <trcGlobalData> \`
		`-trcRoughnessDataSet <trcRoughnessDataSet>`
		Where,
		■ <trcLayer> has the following syntax:
		`-roughnessLower <roughness_name>`
		`-roughnessUpper <roughness_name>`
		`-roughnessLeft <roughness_name>`
		`-roughnessRight <roughness_name> -xhatchUsed <xhatchUsed> -xhatchLayer <xhatchLayer> -conductivity <conductivity> -er <er> -loss <loss> -thickness <thickness> -angle <angle> -layerNum <layerNum> -layerName <layerName> -layerType <layerType> -layerMaterial <layerMaterial> -layerMetalMaterial <layerMetalMaterial>`
		Here,
		<xhatchUsed> and <xhatchLayer> are defined as a Boolean, that is, 1 or 0.
		<conductivity>, <er>, <loss>, <thickness>, and <angle> are numbers.
		<layerNum> is an integer.
		■ <trcLayerRemove> is an integer, layer index.
		■ <trcTraceRemove> is an integer, trace index.

		■ <trcTrace> has the following syntax:
		`-traceSpace <traceSpace> -traceWidth <traceWidth> -traceLayer <traceLayer> -traceType <traceType> -traceName <traceName>`
		Here,
		<traceSpace> and <traceWidth> are numbers.
		<traceLayer> is an integer, layer index.
		<traceType> is enum, 0:SE, 1:DIFF, 2:GND.
		■ <trcTraceData> has the following syntax:
		`-length <length> -upperRef <upperRef> -lowerRef <lowerRef> -includepower <includepower> -addvrm <addvrm> -pdsr <pdsr> -pdsc <pdsc> -materialfile <materialfile> -currentunit <currentunit>`
		Here,
		<length> is a number.
		<includepower> and <addvrm> are defined as a Boolean, that is, 1 or 0.
		■ <cableData> has the following syntax:
		`-isShield <isShield> -c2cPitch <c2cPitch> -cableDia <cableDia> -condConn <condConn> -connDia <connDia> -insuEr <insuEr> -insuLoss <insuLoss> -insuThick <insuThick> -shealthFillEr <shealthFillEr> -shealthFillLoss <shealthFillLoss> -shealthLoss <shealthLoss> -shealthEr <shealthEr> -shieldConn <shieldConn> -thickness <thickness> -twistPitch <twistPitch> -condMaterial <condMaterial> -insuMaterial <insuMaterial> -shealthFillMaterial <shealthFillMaterial> -shealthMaterial <shealthMaterial> -shieldlMaterial <shieldlMaterial>`

		■ <coaxCableData> has the following syntax: -condConn <condConn> -dieEr <dieEr> -dieLoss <dieLoss> -innerR <innerR> -outerR <outerR> -thickness <thickness> -shieldConn <shieldConn> -condMaterial <condMaterial> -dieMaterial <dieMaterial> -shieldMaterial <shieldMaterial> ■ <trcGlobalData> has the following syntax: -cutoffFreq <cutoffFreq> -calTemperature <calTemperature> ■ <trcRoughnessDataSet> has the following syntax: -trcRoughData <trcRoughData>
	<layout_block_params>	
		Following is the syntax for Layout block parameters: -extractionengine <extractionengine> \ -layoutfile <layoutfile> \ -commandlineswitches <commandlineswitches> Where, <extractionengine> is POWERSI, SPEEDEM, or Clarity.
	<VRM_block_params>	
		Following is the syntax for VRM block parameters: -powernet <powernet> -groundnet <groundnet> -numberofpowerpins <numberofpowerpins> -numberofgroundpins <numberofgroundpins> -renamepowerpin <renamepowerpin> -renamegroundpin <renamegroundpin> -setpowerpin <setpowerpin> -setgroundpin <setgroundpin> -parameter <parameter> Where, <parameter> is a list of parameter names and their values specified as: [list <param_name> <param_value>] Here, <param_name> can be TypVoltage, MaxVoltage, MinVoltage, IsSpice, or SpiceFile.

	<clarity_block_params>	
		Following is the syntax for Clarity block parameters: `-layoutfile <layoutfile>` Where, <layoutfile> is a file of the following supported file formats: *.3dem, *.spd, *.brd, *.mcm, and *.sip.
	<source_block_params>	

		Following is the syntax for source block parameters:
		`-modeltype <modeltype> -numberofsignals <numberofsignals> -includepower <includepower> -parameter <parameter>`
		Where,
		■ <modeltype> is Gaussian, PWL, Pulse, or Sinesquare.
		■ <parameter> is a list of parameter names and values specified as:
		`[list <param_name> <param_value>]`
		Here,
		<param_name> can be:
		❑ Rs, Cs, SourceImpedance, InitValue, PeakValue, Delay, GaussPulseWidth, or GaussPeriod for Gaussian
		❑ R, repeat, PWLFormType, PWLFile, Time, or Value for PWL
		❑ InitValue, PeakValue, Delay, RiseTime, FallTime, PulsePulseWidth, PulsePeriod for Pulse and Sinesquare

	<terminator_block_params>	

		Following is the syntax for terminator block parameters:
		`-modeltype <modeltype> -numberofsignals <numberofsignals> -includepower <includepower> -parameter <parameter>`
		Where,
		■ <modeltype> is ShuntR, ShuntRC, SeriesR, Thevenin, DualClamp, HiClamp, LowClamp, or VoltageSource.
		■ <numberofsignals> is an integer, greater than zero.
		■ <includepower> is a Boolean.
		■ <parameter> is a list of parameter names and values specified as:
		`[list <param_name> <param_value>]`
		Here,
		<param_name> can be:
		❑ Resistance, Delay, Z0, or Voltage for ShuntR
		❑ Resistance, Capacitance, Delay, Z0, or Voltage for ShuntRC
		❑ Delay, Z0, or SeriesResistance for SeriesR
		❑ ResistanceUp, ResistanceDown, VoltageUp, VoltageDown, Delay, or Z0 for Thevenin
		❑ CutoffVoltageUp, CutoffVoltageDown, VoltageUp, VoltageDown, Delay, or Z0 for DualClamp
		❑ CutoffVoltage, Voltage, Delay, or Z0 for HiClamp and LowClamp
		❑ Voltage for VoltageSource

	<discrete_block_params>	

		Following is the syntax for Discrete block parameters:
		`-ispowergroup <ispowergroup> \`
		`-modeltype <modeltype> \`
		`-numberofsignals <numberofsignals> \`
		`-includepower <includepower> \`
		`-parameter <parameter>`
		Where,
		■ <ispowergroup> and <includepower> are defined as a Boolean, that is, 1 or 0.
		■ <modeltype> is Resistor, Inductor, Capacitor, SeriesRLC, ParallelRLC, or IdealDiode.
		■ <numberofsignals> is an integer, greater than zero.
		■ <parameter> is a list of parameter names and values specified as:
		[list <param_name> <param_value>]
		Here,
		<param_name> can be:
		❑ Z0, Delay, or CutoffVoltage for IdealDiode
		❑ Resistance for Resistor, SeriesRLC, and ParallelRLC
		❑ Inductance for Inductor, SeriesRLC, and ParallelRLC
		❑ Capacitance for Capacitor, SeriesRLC, and ParallelRLC

GUI Equivalent	None	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

editDieModel

Syntax	`topxp::editDieModel`
Description	Opens the Die Model Editor for editing the die model.
Arguments	None.
GUI Equivalent	Die Model Editor dialog box

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

editFrequencyResponse

Syntax	`editFrequencyResponse \`
	`  [ -maxfreq <double> ] \`
	`  [ -freqpoint <int> ] \`
	`  [ -txSignal [list {Mode SingleEnded} {Signal <block> <signal_net> <gnd_net>}] ] \`
	`  [ -txSignal [list {Mode Diff} {Signal <block> <pos_net> <neg_net>}] ]`	

Description	Edits the frequency response.	
Arguments	`-maxfreq <double>`	
		Specifies the maximum frequency.
	`-freqpoint <int>`	
		Sets *# of Frequency Point*.
	`-txSignal [list {Mode SingleEnded} {Signal <block> <signal_net> <gnd_net>}]`	
		Sets the Tx signal for single-ended mode.
	`-txSignal [list {Mode Diff} {Signal <block> <pos_net> <neg_net>}]`	
		Sets the Tx signal for differential mode.
GUI Equivalent	*Frequency Response* panel that opens when you click the *Tools – Frequency Response* menu in the Topology Workbench window.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

editMCPConnection

Syntax	`editMCPConnection \`
	`  -mcp [list {<Cable>} {<to_Left>} {<Tx>} {<to_Right>}] \`
	`  -autoconnect {<type>} [-pttn [list $pttn1 $pttn2]]`	

Description	Edits the MCP connection between the specified blocks.	
Arguments	`-mcp [list {<Cable>} {<to_Left>} {<Tx>} {<to_Right>}]`	
		Defines the MCP connection that needs to be edited.
	`-autoconnect {<type>} [-pttn [list $pttn1 $pttn2]]`	
		Auto-connects blocks with the same pins. Here, ■ `<type>` can be one of the following: `default`, `node`, `pin`, or `net` ■ `$pttn1, $pttn2` can be a regular expression or a wild string.
GUI Equivalent	*Connection Definition* panel in the Parallel Bus Analysis workflow	

Example

```
editMCPConnection \
    -mcp [list {LSI_1} {%LSI1_1%RW_Interposer%fRDL_1%PM0_sbump} {RW_IP}
    {%RW_Interposer%fRDL_1%LSI1_1%TIV_PSM}]
    -autoconnect cktfile -file {map5.csv}
```

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

editOptions

Syntax	`editOptions \`
	`[ -maxcpuforcircuit <int> ] \`
	`[ -enableuseramp <Boolean> ] \`
	`[ -enablerskin <Boolean> ] \`
	`[ -rskin <double> ] \`
	`[ -shortAllGround <Boolean> ] \`
	`[ -maxcpuforcircuit <int> ] \`
	`[ -enableuseramp <Boolean> ] \`
	`[ -enablerskin <Boolean> ] \`
	`[ -rskin <double> ] \`
	`[ -shortAllGround <Boolean> ] \`
	`[ -addhspice <Boolean> ] \`
	`[ -optimizeSpdsim <Boolean> ] \`
	`[ -showwarningmessage <Boolean> ] \`
	`[ -realtimeplots <Boolean> ] \`
	`[ -verifyTimingBudget <Boolean> ] \`
	`[ -runGoldenCheck <Boolean> ] \`
	`[ -result <option> ] \`
	`[ -simsavefiles <Boolean> ] \`
	`[ -showmostrecentcurves <Boolean> ] \`
	`[ -hidepreviouscurves <Boolean> ] \`
	`[ -stimulustype {Cycle} ] \`
	`[ -stimulustype {PRBS} -prbs_option 0 -poly <value>`
	`-numofbits <value> -prbs_seed <value>  ] \`
	`[ -stimulustype {PRBS} -prbs_option 1 ] \`
	`[ -stimulustype {Ramp} ] \`
	`[ -captureXTalk {Boolean} ] \`
	`[ -ignoreamiclk <Boolean> ] \`
	`[ -probealleyes <Boolean> ] \`
	`[ -wavecnt <Boolean> ] \`
	`[ -nuseblkflt <value> ] \`
	`[ -logofile <file> ] \`
	`[ -linktotd <Boolean> ] \`
	`[ -wingcc <Boolean> ] \`
	`[ -vspath <path>  ]`	

Description	Sets the specified general simulation settings for Topology Workbench.	
Arguments	**Simulation > General**	
	`-maxcpuforcircuit <int>`	
		Sets the maximum CPU numbers for each circuit simulation. **Note:** This setting does not apply to Spectre.

	`-enableuseramp <Boolean>`	
		Selects the *Always use [Ramp] data when no VT curves are available or used* option. **Note:** This setting only applies to SPDSIM.
	`-enablerskin <Boolean>`	
		Activates the *Skin Resistance* option.
	`-rskin <double>`	
		Sets the *Skin Resistance* option. **Note:** This setting only applies to the Pin RLC for Package Parasitics.
	`-shortAllGround <Boolean>`	
		Selects the *Short All Ground Pins to GND*.
	`-addhspice <Boolean>`	
		Selects the *Add Hspice,* if it is selected HPSICE can be used as circuit simulator in *Analysis Option*.
	`-optimizeSpdsim <Boolean>`	
		Selects the *Enabling Additional Performance Improvements*.
	Simulation > Messages and Windows	
	`-showwarningmessage <Boolean>`	
		Select the *Show Warning Messages*.
	`-realtimeplots <Boolean>`	
		Selects the *Show Real-Time Plots*.
	`-verifyTimingBudget <Boolean>`	
		Selects the *Run Golden Parser Check Before Simulation.*
	`-runGoldenCheck <Boolean>`	
		Selects the *Verify Timing Specs Before Simulation*.

	Simulation > Result	
	`-result <option>`	
		Here, `<option>` can be one of the following: ■ `movetohistory` to move the previous simulation results under the *result* folder to the *history* folder. ■ `keep` to keep the previous results under the *result* folder. ■ `delete` to delete the previous results under the *result* folder.
	`-simsavefiles <Boolean>`	
		Selects the *Save Simulation Files*.
	`-showmostrecentcurves <Boolean>`	
		Selects the *Show Curves of Most Recent Simulation*.
	`-hidepreviouscurves <Boolean>`	
		Selects the *Hide Curves of Previous Simulation.*
	Simulation > Channel Simulation > Advanced Bus Characterization Options	
	`-stimulustype {Cycle}`	
		Sets the stimulus type to *Cycle*.
	`-stimulustype {RampBothRiseFall}`	
		Sets the characterization stimulus type to *Consider both rising and falling ramp response*.
	`-stimulustype {PRBS} -prbs_option 0 -poly <value> -numofbits <value> -prbs_seed <value>`	
		Sets the characterization stimulus type to *PRBS.*
	`-stimulustype {PRBS} -prbs_option 1`	
		Sets the characterization stimulus type to *Stimulus Pattern* as set in *Analysis Option.*

	`-stimulustype {Ramp}`	
		Sets the characterization stimulus type to *Ramp*.
	`-captureXTalk {Boolean}`	
		Sets characterization stimulus type to *Capture xTalk on individual signal basis.*
	Simulation > Channel Simulation > Advance AMI options	
	`-ignoreamiclk <Boolean>`	
		Checks *Ignore clock ticks from AMI models*.
	`editOptions -probealleyes <Boolean>`	
		Checks *Probe all eyes (Contours at Tx Input, Tx Output and Rx Input).*
	`-wavecnt <Boolean>`	
		Checks *Output last 1000 bits at Tx Input, Tx Output and Rx Input.* **Note:** The Tx waveform is printed only if the Tx AMI model has a GetWave.
	`-nuseblkflt <value>`	
		Sets the bits for *Get Wave* block size.
	Measurement Report > Generate Report	
	`-logofile <file>`	
		Sets the *Logo File* in the HTML report.
	`-linktotd <Boolean>`	
		Selects *Interface* to *Timing Designer*.
	AMI Builder > General	
	`-wingcc <Boolean>`	
		Specifies the compiler for the AMI builder. `true` sets the compiler as Windows GCC and `false` sets it as Microsoft Visual Studio.
	`-vspath <path>`	
		Sets the bin path for Microsoft Visual Studio.

GUI Equivalent	The *Options* dialog box that opens when you click the *Tools – Options* menu in the Topology Workbench window.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

editProbePoints

Syntax	`editProbePoints \`
	`[-addVoltageProbe [list {Block <block>} {Connection <cnt>} {PosNode <pos_node>} {NegNode <neg_node>}]] |`
	`[-updateVoltageProbe [list {Block <block>} {Connection <cnt>} {PosNode <pos_node>} {NegNode <neg_node>} {CustomName <custom_name>} {IsEnabled <Boolean>}]] |`
	`[-deleteVoltageProbe [list {Block <block>} {Connection <cnt>} {PosNode <pos_node>} {NegNode <neg_node>}]] |`
	`[-addCurrentProbe [list {Block1 <block>} {Connection1 <cnt>} {Node1 <node>} {Block2 <block>} {Connection2 <cnt>} {Node2 <node>}]] |`
	`[-updateCurrentProbe [list {Block1 <block>} {Connection1 <cnt>} {Node1 <node>} {Block2 <block>} {Connection2 <cnt>} {Node2 <node>} {IsReversed <Boolean>} {CustomName <custom_name>} {IsEnabled <Boolean>}]] |`
	`[-deleteCurrentProbe [list {Block1 <block>} {Connection1 <cnt>} {Node1 <node>} {Block2 <block>} {Connection2 <cnt>} {Node2 <node>}]]`	

Description	Opens the *Probe Points* panel where you can edit the probes.	
Arguments	`editProbePoints`	
		Opens the *Probe Points* panel when the command is run without any arguments.
	`editProbePoints -addVoltageProbe [list {Block <block>} {Connection <cnt>} {PosNode <pos_node>} {NegNode <neg_node>}]`	
		Adds a voltage probe.
	`editProbePoints -updateVoltageProbe [list {Block <block>} {Connection <cnt>} {PosNode <pos_node>} {NegNode <neg_node>} {CustomName <custom_name>} {IsEnabled <Boolean>}]`	
		Updates voltage probe. Here, `CustomName <custom_name>` specifies a custom name. `IsEnabled <Boolean>` activates a probe point when set to `True` or deactivates a probe point when set to `False`.

	`editProbePoints -deleteVoltageProbe [list {Block <block>} {Connection <cnt>} {PosNode <pos_node>} {NegNode <neg_node>}]`	
		Deletes voltage probe.
	`editProbePoints -addCurrentProbe [list {Block1 <block>} {Connection1 <cnt>} {Node1 <node>} {Block2 <block>} {Connection2 <cnt>} {Node2 <node>}]`	
		Adds the current probe.
	`editProbePoints -updateCurrentProbe [list {Block1 <block>} {Connection1 <cnt>} {Node1 <node>} {Block2 <block>} {Connection2 <cnt>} {Node2 <node>} {IsReversed <Boolean>} {CustomName <custom_name>} {IsEnabled <Boolean>}]`	
		Updates the current probe. Here, `CustomName <custom_name>` specifies a custom name. `IsEnabled <Boolean>` activates a probe point when set to `True` or deactivates a probe point when set to `False`. `IsReversed <Boolean>` reverses the direction of the current probe.
	`editProbePoints -deleteCurrentProbe [list {Block1 <block>} {Connection1 <cnt>} {Node1 <node>} {Block2 <block>} {Connection2 <cnt>} {Node2 <node>}]`	
		Deletes the current probe.
GUI Equivalent	*Probe Points* panel that opens using: ■ shortcut menu on the canvas -or- ■ the *Setup – Probe Points* menu in the Topology Workbench window	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

editSParamExtraction

Syntax	`editSParamExtraction \`
	`[ -add [list {Port <block> <node> ref} {Name <port_name>}]] |`
	`{ -add [list {DiffPort <block> <pos_node> <neg_node> ref} {Name <diff_port_name>}]] |`
	`[ -delete {Port <port_name>}] |`
	`[ -delete {DiffPort <diff_port_name>}] |`
	`[ -checkblock [list {Mode <mode>} {Block <block>}]] |`
	`[ -uncheckblock [list {Mode <mode>} {Block <block>}]] |`
	`[ -freqranges [list {<start_freq>,<end_freq>,<#ofpoints>,<points_decade>,<sweep_mode>} …`
	`{<start_freq>,<end_freq>,<#ofpoints>,<points_decade>,<sweep_mode>}]] |`
	`[ -filename {<file_name>} ] |`
	`[ -fileformat {<format>}]`	

Description	Opens the *S Parameter Extraction* panel that lets you extract the S Parameter definition.	
Arguments	`editSParamExtraction`	
		Opens the *S Parameter Extraction* panel when the command is run without any arguments.
	`editSParamExtraction -add [list {Port <block> <node> ref} {Name <port_name>}]`	
		Adds the specified single-ended port.
	`editSParamExtraction -add [list {DiffPort <block> <pos_node> <neg_node> ref} {Name <diff_port_name>}]`	
		Adds the specified differential port.
	`editSParamExtraction -delete {Port <port_name>}`	
		Deletes the specified single-ended port.
	`editSParamExtraction -delete {DiffPort <diff_port_name>}`	
		Deletes the specified differential port.
	`editSParamExtraction -checkblock [list {Mode <mode>} {Block <block>}]`	
		Includes the specified block in the S Parameter extraction. Here, `<mode>` can be `single-ended` or `diff`.

	`editSParamExtraction -uncheckblock [list {Mode <mode>} {Block <block>}]`	
		Excludes the specified block in the S Parameter extraction.
	`editSParamExtraction -freqranges [list {<start_freq>,<end_freq>,<#ofpoints>,<points_decade>,<sweep_mode>} … {<start_freq>,<end_freq>,<#ofpoints>,<points_decade>,<sweep_mode>}]`	
		Specifies the setting of frequency ranges.
	`editSParamExtraction -filename {<file_name>}`	
		Specifies the S Parameter file.
	`editSParamExtraction -fileformat {<format>}`	
		Specifies the format of the S Parameter file. Here, `<format>` can be `touchstone` or `bnp`.
GUI Equivalent	*S Parameter Extraction* panel that opens when you click the *Tools – S Parameter Extraction* menu in the Topology Workbench window.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

editTimingBudget

Syntax	`editTimingBudget \`
	`  -spec [list {<block_type>} {BusType <bus>} {<spec> <value>}]\`
	`  -default {BusType <bus>}`	

Description	Sets the maximum frequency.	
Arguments	`<block_type>`	Specifies the block type. Use one of the following values: `Controller` or `Memory`
	`<bus>`	Specifies the bus type. Use one of the following values: `Data`, `Ctrl`, or `AddCmd`
	`<spec>`	Specifies the type of specification being defined. The following values are accepted: ■ `MinTransmitSetup` ■ `MinTransmitHold` ■ `MinReceiveSetup` ■ `MinReceiveHold` ■ `MaxReceiveSkewPlus` ■ `MaxReceiveSkewMinus` ■ `MinStrobeSetup` ■ `MinStrobeHold` ■ `MaxStrobeClockSkewPlus` ■ `MaxStrobeClockSkewMinus` ■ `tDQSQ` ■ `tQH`
	`<value>`	Specifies the value of the given specification.
	`-default {BusType <bus>}`	
		Resets the values to default specifications for the specified bus type.
GUI Equivalent	*Timing Budget* panel that opens when you click the *Tools – Timing Budget...* menu in the Topology Workbench window while analyzing a topology in the Parallel Bus Analysis workflow.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

editWriteLeveling

Syntax	`editWriteLeveling \`
	`  [ -resolution <resolution> ] \`
	`  [ -clkmeasdelay [list {Memory <memory>} {BusGroup <busgroup>} {<delay>}]] \`
	`  [ -strobemeasdelay [list {Memory <memory>} {BusGroup <busgroup>} {<delay>}]] \`
	`  [ -wlo [list {Memory <memory>} {BusGroup <busgroup>} {<delay>}]] \`
	`  [ -default ]`	

Description	Opens the *Write Leveling* panel in the Parallel Bus Analysis workflow for specifying and calculating Write Leveling Offset (WLO).	
Arguments	`-resolution <resolution>`	
		Sets the resolution.
	`-clkmeasdelay [list {Memory <memory>} {BusGroup <busgroup>} {<delay>}]`	
		Sets the clock delay (*ClkMeasDelay*).
	`-strobemeasdelay [list {Memory <memory>} {BusGroup <busgroup>} {<delay>}]`	
		Sets strobe delay (*StrobeMeasDelay*).
	`-wlo [list {Memory <memory>} {BusGroup <busgroup>} {<delay>}]`	
		Sets the WLO value.
	`-default`	
		Resets the modified values to the default settings.
GUI Equivalent	*Write Leveling* panel that opens when you click the *Tools – Write Leveling*... menu in the Topology Workbench window while analyzing a topology in the Parallel Bus Analysis workflow.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

makeReceiver

Syntax	`topxp::makeReceiver <block>`	
Description	Makes the selected IO block the receiver.	
Arguments	`<block>`	Name of the block.
GUI Equivalent	*Make Receiver* option clicked from the shortcut menu displayed when an I/O (IBIS) transmitter block is right-clicked.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

makeTransmitter

Syntax	`topxp::makeTransmitter <block>`	
Description	Makes the selected IO block the transmitter.	
Arguments	`<block>`	Name of the block.
GUI Equivalent	*Make Transmitter* option clicked from the shortcut menu displayed when an I/O (IBIS) receiver block is right-clicked.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

renameComponent

Syntax	`topxp::renameComponent \`
	`  -name <old_name> \`
	`  -newname <new_name>`	

Description	Renames the selected component.	
Arguments	`-name <old_name>`	
		Specifies the current name of the block.
	`-newname <new_name>`	
		Specifies the new name of the block.
GUI Equivalent	None	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

restoreDefaultPins

Syntax	`topxp::restoreDefaultPins`
Description	Restores the selected block's pins to default location and names.
Arguments	None
GUI Equivalent	*Restore Default Pins* option from the shortcut menu displayed when an unconnected block is right-clicked.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

setupDefaultPinLocation

Syntax	`topxp::setupDefaultPinLocation \`
	`  -group <group_param> \`
	`  [-groundnodes <gnd_nodes>] \`
	`  [-powernodes <power_nodes>]`	

Description	Sets up the default pin location for the selected block.	
Arguments	`-group <group_param>`	
		Specifies the group parameters in the following format: `[list <group_name> <direction> <nodes>]` Where, ■ `<group_name>` specifies the group's name. ■ `<direction>` specifies the direction as one of the following: `Left`, `Right`, `Up`, or `Down` ■ `<nodes>` specifies the nodes to be used in the following format: `{node1 node2 … nodeN}`
	`-groundnodes <gnd_nodes>`	
		Optional Specifies a list of ground nodes in the following format: `[list node1 node2 ... nodeN]`
	`-powernodes <power_nodes>`	
		Optional Specifies a list of power nodes in the following format: `[list node1 node2 ... nodeN]`
GUI Equivalent	*Set up Default Pin Location* option from the shortcut menu displayed when a block is right-clicked.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

shortToGND

Syntax	`topxp::shortToGND`
Description	Shorts the selected pin to GND.
Arguments	None.
GUI Equivalent	*Short to GND* option from the shortcut menu displayed when a selected pin is right-clicked.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

shortToPower

Syntax	`topxp::shortToPower`
Description	Shorts the selected pin to a power supply.
Arguments	None.
GUI Equivalent	*Short to Power* option from the shortcut menu displayed when a selected pin is right-clicked.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

shortUnConnectedGroundNode2GND

Syntax	`topxp::shortUnConnectedGroundNode2GND`
Description	Shorts all unconnected ground nodes to global GND.
Arguments	None.
GUI Equivalent	*Short Unconnected Ground Pins to GND* option from the shortcut menu displayed when a selected pin is right-clicked.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

Displaying Elements in a Topology

You can use the following Tcl commands in the Command Window panel:

- displayPin
- displayUnConnectedGroundNode
- displayUnConnectedPowerNode
- displayUnConnectedSignalPins

displayPin

Syntax	`topxp::displayPin < net | pin | node | none >`	
Description	Displays information about the specified type of pin.	
Arguments	`< net | pin | node | none >`	
		Specifies the type of pin for which information needs to be displayed.
GUI Equivalent	*Display Pin Info* option from the shortcut menu displayed when the canvas is right-clicked.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

displayUnConnectedGroundNode

Syntax	`topxp::displayUnConnectedGroundNode`
Description	Toggles the visibility of unconnected ground pins.
Arguments	None
GUI Equivalent	*Display Unconnected Ground Pins* option from the shortcut menu displayed when the canvas is right-clicked.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

displayUnConnectedPowerNode

Syntax	`topxp::displayUnConnectedPowerNode`
Description	Toggles the visibility of unconnected power pins.
Arguments	None
GUI Equivalent	*Display Unconnected Power Pins* option from the shortcut menu displayed when the canvas is right-clicked.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

displayUnConnectedSignalPins

Syntax	`topxp::displayUnConnectedSignalPins`
Description	Toggles the visibility of unconnected signal pins for the selected block.
Arguments	None
GUI Equivalent	*Hide Unconnected Pins* option from the shortcut menu displayed when a block is right-clicked.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

Controlling Simulation and Analysis

You can use the following Tcl commands in the Command Window panel:

- checkConnectivity
- deriveGelementCurrent
- deriveTargetImpedance
- runJitterTolerance
- setOption
- showResult
- startACAnalysis
- startBlockSensitivity
- startBusChannelCharacterization
- startBusSimulation
- startChannelCharacterization
- startChannelSimulation
- startCircuitChannelCorrelation
- startConnectivityCheckSimulation
- startDCAnalysis

- startImpedanceSimulation
- startPowerRippleSimulation
- startSParamExtraction
- startSimulation
- startSweepSimulation
- startTransientAnalysis
- startVoltusModelExtraction
- startWriteLevelingSimulation
- stopSimulation
- terminateUnconnectedNodes

checkConnectivity

Syntax	`checkConnectivity`
	`[ -victimRx {RxSignal <signal>} ] \`
	`[ -maxFrequency {<freq>} ] \`
	`[ -numberOfFrequencyPoint {<point>} ] \`
	`[ -couplingThreshold {<threshold>} ] \`
	`[ -couplingThreshold {Apply} ] \`
	`[ -includeinchannelsimulation [list {TxSignal <signal>}`
	`{RxSignal <signal>} {IsChecked <Boolean>}]]`	

Description	Checks the connectivity of the receiver block with its intended transmitter.	
Arguments	`-victimRx {RxSignal <signal>}`	
		Sets the *Victim Rx* signal.
	`checkConnectivity -maxFrequency {<freq>}`	
		Sets the *Max Frequency.*
	`checkConnectivity -numberOfFrequencyPoint {<point>}`	
		Sets the (number) *# of Frequency Points.*
	`checkConnectivity -couplingThreshold {<threshold>}`	
		Sets the coupling threshold.
	`checkConnectivity -couplingThreshold {Apply}`	
		Applies the coupling threshold to set the check status.
	`checkConnectivity -includeinchannelsimulation [list {TxSignal <signal>} {RxSignal <signal>} {IsChecked <Boolean>}]`	
		Selects or deselects the *Include In Channel Simulation* check box in the *Check Connectivity* panel. Here, `<signal>` can be a differential signal such as `RX_LOWER::pos-neg` or a single-ended signal such as `Mem_U1::DQ2`.
GUI Equivalent	*Check Connectivity* panel that opens when you click the *Check Connectivity* option in the *Simulation Setup* schema.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

deriveGelementCurrent

Syntax	`topxp::deriveGelementCurrent \`
	`  -file <file> \`
	`  [-pwr <pwr>] \`
	`  [-gnd <gnd>]`	

Description	Extract the g-element current profile.	
Arguments	`{<file>}`	
		The name of the subcircuit file.
	`{<pwr>}`	
		Optional Specifies the power node whose current profile will be extracted. If not specified, all the power nodes will be lumped.
	`{<gnd>}`	
		Optional Specifies the ground node to be terminated to global ground. If not specified, all the ground nodes will be terminated to global ground
GUI Equivalent	"Extract G Element Current"	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

deriveTargetImpedance

Syntax	`topxp::deriveTargetImpedance \`
	`  -block <block> \`
	`  -maxpowerrippletype <maxpowerrippletype> \`
	`  -maxpowerripple <maxpowerripple> \`
	`  -resistances <resistances> \`
	`  -startfreq <startfreq> \`
	`  -endfreq <endfreq> \`
	`  -timestep <timestep> \`
	`  -timestop <timestop>`	

Description	Derives the target impedance.	
Arguments	`-block <block>`	
		The name of the block.
	`-maxpowerrippletype <maxpowerrippletype>`	
		Type of ripple. The accepted values are `peak-to-peak` or `undershoot`.
	`-maxpowerripple <maxpowerripple>`	
		Specifies the max power ripple in V.
	`-resistances <resistances>`	
		Specifies the resistance values for optimization. This argument accepts double values separated by a comma (,) or a range specified by "`max-min`".
	`-startfreq <startfreq>`	
		Specifies the starting frequency in Hz.
	`-endfreq <endfreq>`	
		Specifies the ending frequency in Hz.
	`-timestep <timestep>`	
		Specifies the time step in seconds. This is automatically extracted from the current profile.
	`-timestop <timestop>`	
		Specifies the simulation time in seconds.
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

runJitterTolerance

Syntax	`topxp::runJitterTolerance`
Description	Starts the Jitter Tolerance simulation.
Arguments	None.
GUI Equivalent	None.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

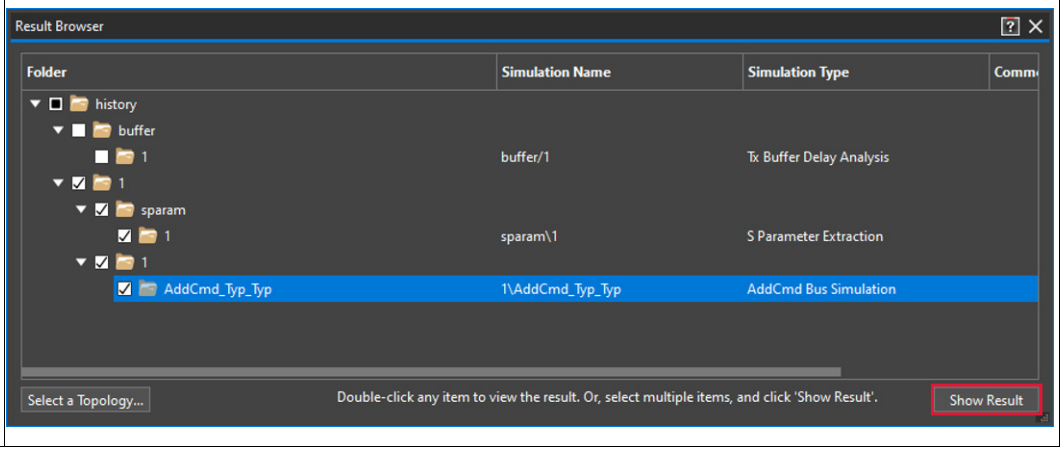
setOption

Syntax	`setOption \` `  -namedsim <boolean>`	
Description	Checks the *Simulation Name* option.	
Arguments	`-namedsim <boolean>`	
		Specifies a Boolean value, `true` or `false`. When set to `true`, the simulation's name is customizable. If `false`, a name is automatically assigned to the simulation.
GUI Equivalent	Analysis Option panel – Circuit Simulation tab	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

showResult

Syntax	`topxp::showResult \` `  [list {$path}]`	
Description	Shows the simulation results that are saved on the specified path.	
Arguments	`list {$path}`	
		Specifies the path from where the identified simulation results need to be shown. `$path` can be the exact path, such as: ■ `history/buffer/1` ■ `1/sparam/1` ■ `1/1/AddCmd_Typ_Typ` `$path` can have wildcards, such as: ■ `*sparam*` ■ `1/sparam/?` The separator for `$path` can be `/` or `\`.
GUI Equivalent	The *Show Result* button in the *Result Browser* panel. 	

Example

```
topxp::showResult [list {history\1\sparam\1} {history\1\1\AddCmd_Typ_Typ}]
```

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

startACAnalysis

Syntax	`topxp::startACAnalysis`
Description	Starts the frequency response simulation. Will work for single-ended or diff-pair connections.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startBlockSensitivity

Syntax	`topxp::startBlockSensitivity`
Description	Starts to run the block sensitivity simulations.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startBusChannelCharacterization

Syntax	`topxp::startBusChannelCharacterization`
Description	Starts channel characterization in Parallel Bus Analysis.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startBusSimulation

Syntax	`topxp::startBusSimulation`
Description	Starts bus simulation.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startChannelCharacterization

Syntax	`topxp::startChannelCharacterization`
Description	Starts channel characterization in Serial Link Analysis.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startChannelSimulation

Syntax	`topxp::startChannelSimulation`
Description	Starts channel simulations in Serial Link Analysis.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startCircuitChannelCorrelation

Syntax	`topxp::startCircuitChannelCorrelation`
Description	Starts circuit channel correlation.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startConnectivityCheckSimulation

Syntax	`topxp::startConnectivityCheckSimulation`
Description	Starts connectivity check simulation.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startDCAnalysis

Syntax	`topxp::startDCAnalysis`
Description	Starts the DC IR Drop Analysis.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startImpedanceSimulation

Syntax	`topxp::startImpedanceSimulation`
Description	Starts the PDN impedance simulation run.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startPowerRippleSimulation

Syntax	`topxp::startPowerRippleSimulation`
Description	Starts the power ripple simulation.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startSParamExtraction

Syntax	`topxp::startSParamExtraction <mode>`	
Description	Starts the S Parameter extraction.	
Arguments	`<mode>`	Specifies the mode of S Parameter extraction, that is, `single-ended` or `diff`.
GUI Equivalent	None.	

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startSimulation

Syntax	`topxp::startSimulation`
Description	Starts the default simulation in the current workflow.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startSweepSimulation

Syntax	`topxp::startSweepSimuation`
Description	Starts the sweep simulation.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startTransientAnalysis

Syntax	`topxp::startTransientAnalyisis`
Description	Starts transient analysis in SLA Tolerance.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startVoltusModelExtraction

Syntax	`topxp::startVoltusModelExtraction`
Description	Starts Voltus model extraction.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

startWriteLevelingSimulation

Syntax	`topxp::startWriteLevelingSimulation`
Description	Starts simulation for calculating write leveling.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

stopSimulation

Syntax	`topxp::stopSimulation`
Description	Stops the current ongoing simulation.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

terminateUnconnectedNodes

Syntax	`terminateUnconnectedNodes \`
	`  -signal <value> \`
	`  -power <value> \`
	`  -ground <value>`	

Description	Sets the termination resistor for un-connected nodes.	
Arguments	`-signal <value>`	
		Specifies the signals that are to be terminated.
	`-power <value>`	
		Specifies the power pins that are to be terminated.
	`-ground <value>`	
		Specifies the ground pins that are to be terminated.
GUI Equivalent	Terminate Unconnected Pins dialog box.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

Exporting from Topology Workbench

You can use the following Tcl commands in the Command Window panel:

- exportBlockSensitivtyResults
- exportReport
- exportToEcsetPage
- extractECSet

exportBlockSensitivtyResults

Syntax	`topxp::exportBlockSensitivityResults \`
	`  <result index> \`
	`  <target file>`	

Description	Exports the selected block sensitivity result to a file.	
Arguments	`{<result index>}`	
		The selected index of result to be exported.
	`{<target file>}`	
		The file name to export the result.
GUI Equivalent	None.	

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

exportReport

Syntax	`topxp::exportReport -path <target file>\`
	`  <-csv | -json> \`
	`  -noparam`
	
	topxp::exportReport -path <target file> -csv|-json -noparam
	
	-csv - the target file format will be csv.
	
	-json - the target file format will be json.
	
	-noparam - sweep parameters will not be exported.	

Description	Exports sweep simulation result to a file.	
Arguments	`<-csv | -json>`	
		Specifies the file format of the target file, that is, CSV or JSON.
	`{<noparam>}`	
		Sweep parameters will not be exported.
GUI Equivalent	None.	

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

exportToEcsetPage

Syntax	`topxp::exportToEcsetPage`
Description	Exports constraint topology.
Arguments	None.
GUI Equivalent	None.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

extractECSet

Syntax	`topxp::extractECSet \`
	`  <tx> \`
	`  <tx pos signal> \`
	`  <tx neg signal> \`
	`  <rx> \`
	`  <rx pos signal> \`
	`  <rx neg signal>`	

Description	Exports the constraint topology from connectivity checkers.	
Arguments	`{<tx>}`	
		Name of transmitter.
	`{<tx pos signal>}`	
		The positive signal name for a diff-pair connectivity, and signal name for single-ended ones.
	`{<tx neg signal>}`	
		Leave this argument empty for a single-ended signal.
	`{<rx>}`	
		Name of receiver.
	`{<rx pos signal>}`	
		The positive receiver signal.
	`{<rx neg signal>}`	
		The negative receiver signal.
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

Managing Projects

You can use the following Tcl commands in the Command Window panel:

- archiveProject

- assignSimName
- checkCompliance
- cleanup
- connectToPowerSupply
- convertToWireBased
- copyProjectAs
- defineBlockGroup
- dumpPropertyWindow
- editPinArg
- getLicensesUsed
- ibisEditor
- launchAbout
- launchAMM
- log
- purgeUnusedModels
- queryLicenseUsage
- saveEcsetPageAsTopFile

archiveProject

Syntax	`archiveProject`
	`[-default]`
	`[-files [list { paths } ...>]]`
	`[-zip {true | false}]`
	`[-o { output_path }]`	

Description	Archives all default project files or only the specified project files. When this Tcl command is run without any arguments, it opens the Archive Topology dialog box.	
Arguments	`-default`	Adds the default project files to archive. These are the files included by default when the Archive Project dialog box is displayed.

	`-files [list] <paths ...>`	

		Adds the project files available at the specified path to the archive list.
		
		The `list` keyword is required, only if there are more than one paths to add.
		
		Each path can be of a folder or a file. A folder is included recursively to add all files that exist in it.
		
		If a path is ended with '/' (or '\' in Windows), the folder itself is not added, but changes the current directory. All the files and folders therein are treated as relative to the current directory.
		
		Each new current directory setting is progressed to the existing setting. The '<' character can be used to regress the current directory such that each less-than sign indicates to regress one level. The level is counted by the specified path items, not physical directories. A single '/' character returns to the root directory at once. For example,
		`-files [list /abc/ x1 dir1/ x2 dir2/dir3/ x3 x4 < x5 / x6]`
		
		is the same as (each `Xn` represents a sub-path name):
		`-files /abc/x1`
		`-files /abc/dir1/x2`
		`-files /abc/dir1/dir2/dir3/x3`
		`-files /abc/dir1/dir2/dir3/x4`
		`-files /abc/dir1/x5`
		`-files x6`

	`-zip {true | false}`	
		Enables or disables zip package of the archived project files. This feature is enabled by default.

	`-o <output_path>`	
		The default output path is the current project's folder name with the postfix `_copy` appended to it. With the `-o` argument you can specify another full path where the archived output should be saved. Otherwise, you can specify only the directory or file name and Topology Workbench uses the default path setting for the omitted part. If `-zip` is set to `false`, the output will always be a directory. However, when `-zip` is set to `true`, the path must be ended with `'/'` if you want to state a directory. The extension `.zip` will be added automatically to the directory name in this case.
GUI Equivalent	In the Topology Workbench window, the *Topology – Archive* menu.	

Related Topic

Archiving a Topology

Using the Topology Workbench Tcl Commands: An Overview

assignSimName

Syntax	`topxp::assignSimName {<name>}`	
Description	Set a custom name for the current simulation.	
Arguments	`<name>`	
		Specifies the name of the simulation.
GUI Equivalent	*Assign Simulation Name* dialog box that opens when the simulation is started after setting the *Simulation Name* option in the *Circuit Simulation* tab of the Analysis Option panel to *Custom*.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

checkCompliance

Syntax	`topxp::checkCompliance`
Description	Starts the check compliance simulations in the Serial Link Analysis workflow.
Arguments	None.
GUI Equivalent	*Check Compliance* in the *Simulation* schema of the *Workflow* panel displayed when using *Custom Compliance Kit*.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

cleanup

Syntax	`topxp::cleanup`
Description	Replaces all the blocks in the canvas and automatically routes all the connections.
Arguments	None
GUI Equivalent	The *Cleanup Canvas* option in the shortcut menu displayed on a right-click on the canvas.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

connectToPowerSupply

Syntax	`topxp::connectToPowerSupply`
Description	Connects the selected pins to power supply.
Arguments	None
GUI Equivalent	Context Menu - "Connect to Power Supply"

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

convertToWireBased

Syntax	`topxp::convertToWireBased`
Description	Converts legacy SystemSI (old canvas) topologies having MCP-based connectivity to wire-based connectivity. All legacy SystemSI topologies use MCP-based connectivity. Therefore, when you import them into Topology Workbench (new canvas), the legacy topologies continue to hold their MCP-based connectivity property. In addition, for large topologies, like the coupled buses with many signals, this is a great way to represent it. However, some topologies, for aspects like a single signal, or a diff pair, or other cases with a small number of signals, it is easier to work with a wire-based connectivity instead. In such scenarios, you can use this Tcl command that automatically converts all the MCP-based connectivity into wire-based connectivity and conversely.
Arguments	None
GUI Equivalent	None

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

copyProjectAs

Syntax	`topxp::copyProjectAs <file>`	
Description	Save a copy of the current topology project.	
Arguments	`<file>`	
		Specifies the name to be assigned to the saved copy.
GUI Equivalent	In the Topology Workbench window, the *Copy Topology As…* menu.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

defineBlockGroup

Syntax	`topxp::defineBlockGroup \`
	`  {<block_type>} \`
	`  [list {<group_name>} … {<group_name>}]`	

Description	Defines the block groups of Memory or EBD in PBA.	
Arguments	`{<block_type>}`	
		Can be from Memory or MemSPICEBlock or EBD
	`{<group_name>}`	
		Specifies the name of a group. This is in list format in case of multiple groups.
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

dumpPropertyWindow

Syntax	`topxp::dumpPropertyWindow {`*`output_file_path`*`}`	
Description	Prints the properties of the block to the specified file. Running the simulation and comparing the result is one way to make sure that the setups and connections are correct. To get quick data for comparison, you can use this command to print the properties of various blocks in the topology to a text file.	
Arguments	{*output_file_path*}	
		Specifies the path where the output file needs to be saved.
GUI Equivalent	None	

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

editPinArg

Syntax	`topxp::editPinArg \`
	`  -block <block name> \`
	`  -pin <pin name> \`
	`  -value <value>`	

Description	Edit the specified pin property's value.	
Arguments	`{<block name>}`	
		Specifies the name of the block.
	`{<pin name>}`	
		Specifies the name of the pin.
	`{<value>}`	
		Value of voltage for shorted to power pins / Value of resistance for terminated with resistor pins / Values of voltages for connected to power supply pins.
GUI Equivalent	None.	

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

getLicensesUsed

Syntax	`topxp::getlicensesUsed`
Description	Returns current checked out license.
Arguments	None.
GUI Equivalent	In the Topology Workbench window, the *Help – License Used...* menu.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

ibisEditor

Syntax	`topxp::ibisEditor`
Description	Opens the IBIS editor.
Arguments	None.
GUI Equivalent	In the Topology Workbench window, the *Tools – IBIS Editor...* menu.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

launchAbout

Syntax	`topxp::launchAbout`
Description	Shows the *About* dialog box.
Arguments	None.
GUI Equivalent	In the Topology Workbench window, the *Help – About* menu.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

launchAMM

Syntax	`topxp::launchAMM \`
	`  -libraries | -browsing \ \`
	`  -save <file> | -load <file>`	

Description	Displays the Sigrity™ Analysis Model Manager (AMM) main window.	
Arguments	`-libraries`	Displays the AMM Library Management window.
	`-browsing`	Displays the AMM Browse Models window.
	`-save <file>`	Specifies the file name to be used to save the AMM library.
	`-load <file>`	Imports AMM library from the specified file.
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

log

Syntax	`topxp::log <message>`	
Description	Adds a message to the log file.	
Arguments	`<message>`	Message to be appended to the file
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

purgeUnusedModels

Syntax	`topxp::purgeUnusedModels`
Description	Purges unused models in asi_models.
Arguments	None.
GUI Equivalent	In the Topology Workbench window, the *Topology – Purge Unused Models...* menu.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

queryLicenseUsage

Syntax	`topxp::queryLicenseUsage <feature>`	
Description	Queries the status of the licenses.	
Arguments	`<feature>`	Specifies the license string for which status needs to be retrieved. The result is displayed in the following format: `<feature>: <total> available, <used> used` Where, ■ `<feature>` is the specified license string. ■ `<total>` displays the total number of available licenses. ■ `<used>` displays the number of licenses that are currently checked out.
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

saveEcsetPageAsTopFile

Syntax	`topxp::saveEcsetPageAsTopFile <file>`	
Description	Save the constraint topology as a SigXplorer `.top` file.	
Arguments	`<file>`	Name of the file
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

terminateWithR

Syntax	`topxp::terminateWithR`
Description	Exits the selected pin with resistor.
Arguments	None.
GUI Equivalent	None.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

Generating Via Models

You can use the following Tcl commands in the Command Window panel:

- auroraViaWizard
- sigrityViaWizard

auroraViaWizard

Syntax	`topxp::auroraViaWizard <block_name>`	
Description	Opens the Aurora Via Wizard that helps to generate via models in the pre-layout phase for the specified block.	
Arguments	`<block_name>`	Specifies the name of the block for which the via model needs to be generated
GUI Equivalent	The *Aurora Via Wizard* button in the *Edit Properties* panel displayed when a Via block is double-clicked in the Topology Workbench canvas.	

Related Topic

Aurora Via Wizard

Using the Topology Workbench Tcl Commands: An Overview

sigrityViaWizard

Syntax	`topxp::sigrityViaWizard <block_name>`
Description	Opens the Sigrity Via Wizard that helps to generate via models in the pre-layout phase for the specified block.
GUI Equivalent	The *Sigrity Via Wizard* button in the *Edit Properties* panel displayed when a Via block is double-clicked in the Topology Workbench canvas.

Related Topic

Sigrity Via Wizard

Using the Topology Workbench Tcl Commands: An Overview

Managing AMI Models

You can use the following Tcl commands in the Command Window panel:

- deleteAMI
- modelEnable

deleteAMI

Syntax	`topxp::deleteAMI \`
	`  {<block>} \`
	`  {<index>}`	

Description	Delete the AMI of index {index} from the block.	
Arguments	`{<block>}`	
		Specifies the block that contains the AMI to be deleted.
	`{<index>}`	
		Specifies the index of the AMI in the block.
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

modelEnable

Syntax	`topxp::modelEnable \` `  <true|false>`	
Description	Enable/disable the currently selected AMI model.	
Arguments	`{<true|false>}`	
		If input is true, then the selected AMI model is enabled. If input is false, then the selected model is disabled.
GUI Equivalent	None.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

Handling Constraint Manager

You can use the following Tcl commands in the Command Window panel:

- setDesignPrecision
- setOptionalPins
- updateCM

setDesignPrecision

Syntax	`topxp::setDesignPrecision <precision>:<unit>`	
Description	Sets the precision and unit for a constraint design.	
Arguments	`<precision>:<unit>`	
		Specifies the precision and unit. The valid values for `<precision>` can be: ■ an integer range in `[2,4]` for `cm`, `inch` ■ `[0,4]` for `um` ■ `[1-4]` for `mm` ■ `[0,2]` for `mil` For `<unit>`, valid values can be one of the following: `cm`, `in`, `um`, `mm`, or `mil`
GUI Equivalent	*Design Units and Precision* dialog box that opens when *Set Precision* option is clicked in the *Workflow* panel of the exported constraint topology.	

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

setOptionalPins

Syntax	`topxp::setOptionalPins`
Description	Marks the pins on a selected component as optional when Ecset is updated in Constraint Manager.
Arguments	None.
GUI Equivalent	*Optional Pins* from the shortcut menu displayed on right-clicking a selected block in the ECSet page.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

updateCM

Syntax	`topxp::updateCM`
Description	Updates the constraint manager for the selected object.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

Toggling in Topology Workbench

You can use the following Tcl commands in the Command Window panel:

- toggleAutoShort
- toggleBackChannel
- togglePauseBeforeSim
- toggleShowNewMCP
- toggleTxRxJitterNoise

toggleAutoShort

Syntax	`topxp::toggleAutoShort`
Description	Toggles auto-short for the selected block.
Arguments	None.
GUI Equivalent	None.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

toggleBackChannel

Syntax	`topxp::toggleBackChannel`
Description	Toggles the back channel in SLA.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

togglePauseBeforeSim

Syntax	`topxp::togglePauseBeforeSim`
Description	Toggles “Pause before Simulation”.
Arguments	None.
GUI Equivalent	In the Topology Workbench window, the *Setup – Pause before Simulation* menu.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview

toggleShowNewMCP

Syntax	`topxp::toggleShowNewMCP`
Description	Toggles the *Show new connection definition* option.
Arguments	None.
GUI Equivalent	None.

Related Topics

Using the Topology Workbench Tcl Commands: An Overview

toggleTxRxJitterNoise

Syntax	`topxp::toggleTxRxJitterNoise`
Description	Toggles transmitter (Tx) and receiver (Rx) jitter noise.
Arguments	None.
GUI Equivalent	None.

Related Topic

Using the Topology Workbench Tcl Commands: An Overview